Heterogeneous Parallel Island Models

Lucas A. da Silveira[†], José L. Soncco-Álvarez[*], Thaynara A. de Lima[‡] and Mauricio Ayala-Rincón[†]
[†]Department of Computer Science, Universidade de Brasília, 70900–010 Brasília D.F., Brazil
[*]Departamento Académico de Informática, Universidad Nacional de San Antonio Abad del Cusco, Cusco, Peru
[‡]Institute of Mathematics and Statistics, Universidade Federal de Goiás — Campus II, 74690–900 Goiânia, Brazil

***Abstract*—Homogeneous Parallel Island Models (HoPIMs) run the same bio-inspired algorithm (BA) in all islands. Several communication topologies and migration policies have been fine-tuned in such models, speeding up and providing better quality solutions than sequential BAs for different case studies. This work selects four HoPIMs that successfully ran a genetic algorithm (GA) in all their islands. Furthermore, it proposes and studies the performance of *heterogeneous* versions of such models (HePIMs) that run four different BAs in their islands, namely, GA, double-point crossover GA, Differential Evolution, and Particle Swarm Optimization. HePIMs aim to maintain population diversity covering the space of solutions and reducing the overlap between islands. The $\mathcal{NP}$-hard evolutionary reversal distance problem is addressed with HePIMs verifying their ability to compute accurate solutions and outperforming HoPIMs.**

I. Introduction

Evolutionary computing relies on the evolution of candidate solutions over a finite number of temporal steps to obtain accurate solutions for complex problems. Bio-inspired algorithms (BAs) have been applied to solve search and optimization problems in different domains where finding exact solutions is difficult or impossible, applying traditional approaches such as mathematical programming. BAs' fundamental principle uses a constructive method to obtain an initial feasible population and a local search technique to improve the population individuals (solutions). In BAs, the population evolves according to specified rules that promote information exchange between individuals. Such approaches may require a prohibitive amount of computing resources, and thus, a variety of issues are studied to design efficient algorithms. With this aim in mind, advances are continuously being achieved by developing hybrid algorithms. This work focus on improvements using parallel island models (PIM) that partition the population among their islands (processors) and simultaneously run BAs in them. Each island evolves its local population running its BA. Furthermore, interactions between islands are organized through a communication topology, and a *migration* strategy applied periodically, aiming to improve solutions by sharing individuals.

This paper extends previous work on homogeneous PIMs (HoPIMs) (that apply the same BA in all their islands) [1] allowing running different BAs in their islands, known as *heterogeneous* PIMs (HePIMs). Islands of the proposed architectures run four BAs: simple Genetic Algorithm (`GA`), double-point crossover GA (`GAD`), Differential Evolution (`DE`), and self-adjusting Particle Swarm Optimization (`PSO`) (see e.g., [2], [3], and [4]). Different aspects influence the performance provided by the PIMs. Since we focus on BAs with structured panmictic populations running in parallel, we need to address decisive implementation issues such as communication synchronism, communication frequency, communication topology, and selection of migrants. The communication synchronism defines whether the migration process occurs synchronous or asynchronous; communication frequency establishes the interval between migrations; communication topology is responsible for organizing the neighborhood of each island, and; selection of migrants defines the class of individuals to be shared between islands.

Our case study is the unsigned reversal distance problem (URD), a well-known $\mathcal{NP}$-hard problem [5], [6]. In comparative genomics, rearrangement by reversals is a relevant operation to estimate the evolutionary distance between genomes. Computing the minimum number of reversals to obtain a genome from another is known as the reversal distance (RD) problem [7]. The complexity of the RD problem depends on how the genes are abstracted into the genome. When genes are not orientated, the problem corresponds to URD.

Main contributions.

- Synchronous and asynchronous HoPIMs are designed from a static and a dynamic topology running the four BAs in their islands: `GA`, `GAD`, `DE`, and `PSO`. The experiments ratify that, independently of the bio-inspired algorithm, HoPIMs with breeding-cycle and migration parameters accurately calibrated always outperform their sequential versions. Furthermore, the HoPIM running `DE` provides the most accurate results.
- Synchronous and asynchronous HePIMs are designed, which run the four BAs in their islands and have identical communication topologies than the HoPIMs. After calibration of the migration parameters, experiments with these HePIMs demonstrate that the successful experiences of the best adapted bio-inspired algorithms in each island are propagated towards all other islands. Such propagation allows for providing very competitive outputs. Furthermore, the solutions obtained with each HePIM outperforms the average results obtained by the group of islands in the model running the same BA. Also, HePIMs outperform the results of HoPIMs running `GA`, `GAD`, and `PSO`. In addition, the HePIMs provide competitive results regarding HoPIMs applying `DE`. These observations make it evident that HePIMs may not necessarily deliver better results than HoPIMs, in contrast to recent works (e.g., [8], [9]). However, they show that the diversity and migration policy of HePIMs assure the propagation of the best evolutionary experiences among all islands in such architectures.

Design and experiments are performed over readily available parallel hardware (as multiprocessors). The significance of the designed models is validated through statistical tests.

Organization. Sec. II presents the background on PIMs, the case study, and discusses the four applied BAs and related work. Sec. III introduces the HePIMs explaining how different BAs are distributed on their islands and their communication topologies. Then, Sec. IV presents experiments and discusses accuracy results and statistical analysis. Finally, Sec. V concludes and proposes future work. Source and data used in the experiments are available at http://genoma.cic.unb.br.

II. Background

A. Parallel island model (PIM)

PIMs, initially proposed for GAs [10], are highly suited to run other BAs to increase population diversity. In addition to improving run-time, it is always expected such models improve the quality of sequential GAs' solutions.

The population in a PIM is partitioned into its islands that evolve, executing their BAs in parallel. The islands are connected through a given topology, over which individuals are exchanged through migration. The exchange follows a *migration policy* that is an essential strategy for evolution in the global context and positively impacts the quality of solutions when well-adjusted. PIMs that run the same BA in all their islands are homogeneous, while those that run different BAs in their islands are heterogeneous.

Aspects under the developer's responsibility that determine a PIM's architecture include the number of islands, the communication topology, and the BAs applied in each island, [11]. The topology is responsible for establishing islands' neighborhoods. It is static if the connections remain unchanged throughout the PIM execution, whereas it is dynamic if there is a change during the evolutionary process. In addition, it is interesting to define whether individuals exchange will happens uni- or bi-directionally. There is a variety of schemes available to implement a topology (e.g., [9], [1], [12]). Fig. 2 (a) shows an example of a static bi-directional tree topology. The vertices represent the islands with their respective BAs, while the edges the connections between them.

In addition to breeding-cycle parameters of their BAs, PIMs include migration parameters briefly described below.

- Migration Interval (*MI*): percentage of iterations of the evolutionary process (generations) after which the migration process is repeated. This percentage defines the migration interval. Before each generation in which the migration process occurs, each island separately evolves its population by *MI* $\times$ *maxIt* generations, where *maxIt* is the total number of iterations performed by each BA.
- Individuals number (*IN*): number of individuals emigrating from each island.
- Emigration Policy (*EP*): defines whether individuals are **cloned** or **removed** in the local island when they emigrate to the target island.
- Emigrant Individuals (*EMI*): determines the type of individuals selected for emigration among: 1. **best**, 2. **worst**, and 3. **random**.
- Immigrant Individuals (*IMI*): defines the type of individuals in the target island replaced by immigrants among: 1. **worst**, 2. **random**, and 3. **similar**. Similar individuals are individuals with the same fitness rank as the immigrants.

B. Case-study

Computing the evolutionary distance between organisms by comparing their genomes has become possible through global rearrangements where the chromosome genes are rearranged using some evolutionary operation. Here, the evolutionary distance between organisms containing a single chromosome is computed as the minimum number of reversals necessary to convert a genome into the other.

A genome containing n genes is represented as a permutation $\pi = (\pi_1, \pi_2, ..., \pi_n)$, where $1 \leq i, \pi_i \leq n$; i.e., a bijection on $\{1, \cdots, n\}$. The reversal operation, denoted as $\rho^{j,k}$, for $1 \leq j \leq k \leq n$, inverts contiguous elements between π_j and π_k transforming π into $\pi' = (\cdots, \pi_{j-1}, \pi_k, \cdots, \pi_j, \pi_{k+1}, \cdots)$. There are two types of permutations: signed and unsigned. When the orientation of the genes inside the genome is taken, the genome is seen as a signed permutation. Thus, each gene has a positive or negative sign according to its orientation within the genome. Computing the minimum number of reversals between two signed permutations is called the signed reversal distance (SRD) problem, and it belongs to $\mathcal{P}$ [13]. On the other hand, if the genes have no orientation, the genome is represented as an unsigned permutation. Computing the minimum reversal distance between two unsigned permutations is called the unsigned reversal distance (URD) problem, which is a well-known $\mathcal{NP}$-hard problem [5]. In the BAs used here, the fitness is computed as the reversal distance of signed genomes obtained by randomly orienting the genes in the genome.

C. Local Evolutionary Engines — bio-inspired Algorithms

The four BAs were chosen because they are optimization benchmarks that offer evolutionary mechanisms that provide different adaptability features.

- Simple Genetic Algorithm (`GA`): is inspired by the principles of natural evolution and was developed by J. H. Holland in the 1970s [2]. The `GA` evolves the local population through the breeding cycle parameters consisting of the percentages of *selection* and *replacement*, and the probability of application of *mutation* and *crossover*. For the breeding cycle, the `GA` selects the best parents and applies one point crossover (Fig. 1 (a)) to produce offspring. Then, the descendants replace the worst individuals in the current population.
- Double-point Crossover Genetic Algorithm (`GAD`): is a variant from the `GA` in which the *crossover* uses the technique presented in Fig. 1 (b), and the *replacement* of individuals by descendants in the current population randomly selects the individuals to be replaced.
- Differential Evolution (`DE`): is a optimization method for multidimensional real value functions that use a population

of individual solutions, proposed by Storn and Price [3]. The *mutation factor* F_M, and the *probability of crossover* P_C guide the evolutionary process. In the mutation, three distinct individuals I_α, I_β, and I_γ are randomly selected from the population. Then, I_α suffers a disturbance resulting from the vector difference between I_β and I_γ multiplied by F_M, giving rise to a new mutated individual $I_n = I_\alpha + F_M \times (I_\beta - I_\gamma)$. At each iteration, a new population is generated, replacing individuals with the worst fitness by mutants.

- Self-adjusting Particle Swarm Optimization (PSO): PSO is inspired by the behavior of social organisms in groups and was introduced to address continuous domain problems [4]. We use the self-adaptive PSO proposed in [14], where the weight of inertia (momentum), w, and the individual and global acceleration coefficients, c_1 and c_2 are self-tuning during the search process. Particles in PSO are points, $x_i = (x_{i1}, \cdots, x_{in})$, in an n-dimensional space. Particles motion is given by velocity vectors $v_i = (v_{i1}, \cdots, v_{in})$. Each particle remembers its best position, $pbest_i$. The best of all particle positions, *gbest*, is maintained by the algorithm. $pbest_i$ and *gbest* are used to evolve particles. The velocity vector, v_i^{k+1} for the i^{th}-particle in the $(k+1)^{th}$ iteration is computed as:

$$v_{ij}^{k+1} = w_i^k \cdot v_{ij}^k + c_{1i}^k \cdot rand_{1ij} \cdot (pbest_{ij}^k - x_{ij}^k) + c_{2i}^k \cdot rand_{2ij} \cdot (gbest_j^k - x_{ij}^k) \quad (1)$$

In Eq. (1), w introduces friction into the particles motion, reducing inertial velocity; c_1 and c_2 are the individual and global acceleration coefficients that influence the maximum step size a particle can take and, $rand_{1i}$ and $rand_{2i}$ are random number vectors generated at each iteration, $(0 \leq rand \leq 1)$. Particle next positions are computed as: $x_i^{k+1} = x_i^k + v_i^{k+1}$.

PSO and DE are known to be suitable for solving real value function problems. For adapting the BAs to the URD problem, each particle/individual in the swarm/population is associated with a signed permutation randomly constructed from the unsigned permutation provided as input. For GA and GAD, the orientation of the genes in each individual is randomly generated as ± 1; for PSO and DE, the orientation is randomly generated as a value in the closed interval $[0, 1]$, being values smaller than and greater than or equal to 0.5 interpreted as negative and positive orientations, respectively. For applying PSO and DE, if this continuous representation of the orientation of an individual's gene gets outside the interval $[0, 1]$, a correct orientation is randomly generated. The fitness of each particle/individual is computed using Bader *et al.* [15] linear approach to solve the SRD problem.

D. Related Work

Bianchini and Brown [16] proposed a HePIM that connects islands by ring and torus topologies. HePIMs and HoPIMs were applied to the task map scheduling problem, for which the former PIMs found the best solutions. In addition, they observed that adding more islands is more advantageous than increasing population size. Different HePIMs were proposed by Lin *et al.* in [17], considering several migration strategies and topologies to avoid premature convergence and addressing the graph partitioning problem. Lin *et al.* showed that the sequential GAs do not equate to 25-island PIMs, where good individuals replace the worst individuals on target islands. Furthermore, exchanging individuals based on fitness-based population similarity instead of a static connection topology get better results without any speed degradation.

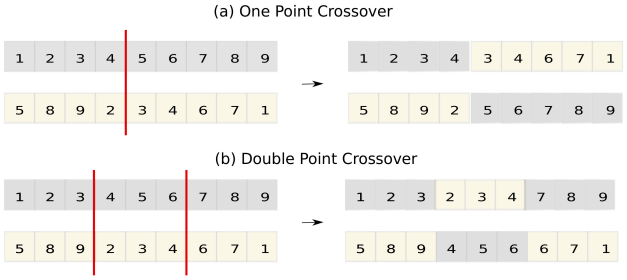

Fig. 1. Variants of crossing operators.

Izzo *et al.* [18] proposed a HePIM from variations of the differential evolution algorithm using asynchronous migration. Such migration was justified because it is more intuitive and suitable for distributed computing over TCP/IP, where resources might become available/unavailable at any time. The proposed PIMs obtained better performance and accuracy than their sequential versions. Gong and Fukunaga [19] proposed a GA based PIM that randomly selects different parameters for each processor. By having a sufficient number of processors, it is expected that some of the processors will end up being assigned parameters that perform well on a given problem.

Duarte *et al.* [8] proposed a migration policy for five-island HePIMs with target island defined by attractiveness. Islands' attractiveness is based on their solutions' quality. This model was improved in [9], inspired by the natural phenomenon known as stigmergy [20], adjusting how attractiveness and the weights of the islands' connections are computed. Experiments were conducted over the fifteen problems given in [21].

In [22], we proposed HoPIMs for a sequential GA introduced in [23] to solve unsigned translocation distance problem. These HoPIMs used ring and complete graph topologies migrating at each generation but reusing parameters calibrated for the sequential GA. Such PIMs' accuracy outperformed the GA after careful calibration of the migration and breeding cycle parameters and exploration of other static topologies such as torus, complete graph, tree, and net and migration dynamics that consider individuals' characteristics [24], [25]. Further, in [1], we analyzed synchronous HoPIMs for three BAs: GA, PSO and Social Spider Algorithm (SSA). Experiments for smaller populations (than those used in this paper) showed that static HoPIMs from the PSO output the best solutions in general, and PIMs from the GA are competitive. The HoPIMs for SSA presented the worst solutions but the best speed-ups.

Similar algorithmic approaches appear in the context of solving multiobjective optimization problems (MOP). For instance, in [26], Zang *et al.*, introduced a multi-swarm optimizer that handles each objective function of a MOP with a different slave swarm. Simultaneously, a master swarm

covers gaps among non-dominated optima by using a local multiobjective PSO algorithm. More recently, in [27], Gong *et al.* proposed a framework to tackle dynamic interval MOPs that handles a cooperative co-evolutionary optimization based on interval similarity. Dynamic interval MOPs are MOPs with changing ranges of interval parameters in at least one objective or constraint over time. The framework divides decision variables into two groups according to the interval similarity and interval parameters. Then, two subpopulations are used to optimize decision variables in the two groups cooperatively. Furthermore, Xu *et al.* proposed a model with EAs that uses two subpopulations to solve dynamic interval multiobjective problems [28]. In [29], Hashimoto *et al.* proposed a PIM to solve multi-task problems. The number of islands and tasks is the same, and each island is responsible for evaluating a single objective. Migration is performed periodically where a pre-specified number of migrant individuals are selected and removed at random on each local island. The worst individuals are replaced, and immigrants have their fitness values set as the worst, assuming that feasible solutions for a task have great chances of being unsuitable for another objective. Experiments were carried out with two objective problems obtaining similar or better results than the sequential model, mainly when applying small migration intervals. An interesting potential application of our HePIMs is the treatment of multiobjective problems. For such applications, each island may take care of the optimization of a different objective function.

III. Communication Topologies

A static and a dynamic topology that successfully addressed URD in [1] are selected. The static topology is the bi-directional binary tree topology, and the dynamic topology is the potentially complete graph. The island communication dynamism is acquired by exploring diversity and quality into each island, given by fitness variance and average metrics. Variance measures islands' diversity: high variance represents high individuals' diversity, improving the chances of evolution into islands. Fitness average measures the quality of island populations. According to such metrics, the islands are ranked as **good**, **bad**, and **medium**. Migrations exchange individuals between good and bad islands, and medium and medium islands only (for short, *gbmm*).

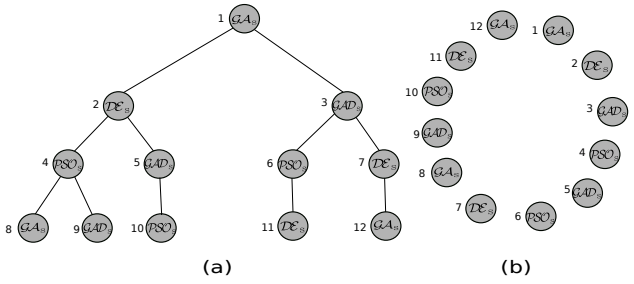

Fig. 2. Communication topologies and distribution of BAs among the islands for heterogeneous PIMs: (a) static binarytree, (b) dynamic complete graph.

IV. Experiments and Analysis of Accuracy

All PIMs were implemented using the `MPI` library of the `C` language in the Linux operating system. The experiments were executed on a computer using two Xeon E5-2620 2.4 GHz six core processors with hyper-threading.

TABLE I
Estimated Values for the Parameters

	Parameter	Estimated values
GA and GAD	*crossover*	$0.02, 0.04, \cdots, 0.98, 1.0$
	mutation	$0.01, 0.011, \cdots, 0.019, 0.02$
	selection	$2\%, 4\%, \cdots, 98\%, 100\%$
	replacement	$2\%, 4\%, \cdots, 98\%, 100\%$
DE	P_C	$0.02, 0.04, \cdots, 0.98, 1.0$
	F_M	$1\%, 1.1\%, \cdots, 1.9\%, 2\%$
Migration	*IN*	1,2,3,4,5,6,7,8,9,10,11,12,13
	EMI	1=Best, 2=Worst, 3=Random
	EP	1=Clone, 2=Remove
	IMI	1=Worst, 2=Random, 3=Similar
	MI	$2\%, 4\%, \cdots, 98\%, 100\%$

To compare the performance of the models, sequential versions of `GA`, `GAD`, `DE` and `PSO` with population/ swarm of size $24n \log n$ and breeding cycles fixed as n were implemented. Posteriorly, 12-island synchronous and asynchronous homogeneous PIMs evolving native individuals through the BAs: `GA`, `GAD`, `DE` and `PSO` from the communication topologies showed in Fig. 2 were designed. The notation for these models is exemplified by $\mathcal{P}^{\mathtt{GA}}_{\text{gbmm12S}}$ and $\mathcal{P}^{\mathtt{DE}}_{\text{Tr12A}}$. The superscripts indicate that these models use the BAs `GA` and `DE`, respectively; the subscripts prefixes *gbmm* and *tree* that they use the static tree and dynamic complete graph topology (*gbmm*), respectively; and, the subscript suffixes 12S and 12A, the number of islands and whether the model is synchronous or asynchronous.

The four HePIMs, denoted as $\mathcal{P}^{\text{Het}}_{\text{Tr12S}}$ and $\mathcal{P}^{\text{Het}}_{\text{Tr12S}}$, and $\mathcal{P}^{\text{Het}}_{\text{gbmm12S}}$ and $\mathcal{P}^{\text{Het}}_{\text{gbmm12A}}$, were designed over the communication topologies with the distribution of BAs given in Fig. 2. The connections between the islands in Fig. 2 (b) are generated at run-time before each migration (see Section III).

A. Parameter Setup

The sequential BAs and PIMs were submitted to *parameter tuning* adopting the "taxonomy T1" in [30]. The calibration approach gives suitable parameters fixed during the experiment's evolutionary process. Table I presents the ranges of parameter values. For those involving percentages, the tested values range between 2% and 100%. For those involving probability, the rate was set from 0.02 to 1.0, and for the mutation parameter from 0.01 to 0.02. For the `DE` F_M parameter, the range from 1% to 2% was tested because values above 2% degrade the quality of solutions. Remember that in `PSO`, the parameters to guide the particles in the search space are self-adjusting.

In the setup, we used groups of twenty n-gene permutations for $n \in \{50, 60, \ldots, 140, 150\}$. Experiments were separately performed for sequential BAs, HoPIMs and HePIMs. Firstly, we calibrate parameters for the sequential BAs and HoPIMs; each parameter reference value was submitted to evaluation and those values that provided the best solutions were selected, see Tables II and III. The HePIMs inherited the HoPIMs parameter values related to the evolutionary process, and only the migration parameters were calibrated, see Table IV.

TABLE II
PARAMETER SETTINGS FOR GA, GAD, AND ASSOCIATED HOPIMS.

		$\mathcal{P}^{\mathrm{GA}}$					$\mathcal{P}^{\mathrm{GAD}}$			
Parameter	GA	Tr12S	Tr12A	gbmm12S	gbmm12A	GAD	Tr12S	Tr12A	gbmm12S	gbmm12A
crossover	.90	.96	.98	.98	.96	.92	.94	.98	.98	.98
mutation	.02	.01	.015	.01	.011	.01	.01	.01	.01	.01
selection	60%	98%	92%	78%	94%	98%	96%	98%	94%	94%
replacement	60%	50%	70%	50%	70%	90%	80%	80%	90%	90%
IN		3	9	2	5		2	12	9	5
EMI		1	1	3	1		1	1	1	1
EP		2	2	1	2		2	2	1	1
IMI		2	1	2	1		1	1	1	1
MI		56%	30%	30%	30%		16%	14%	14%	12%

TABLE III
PARAMETER SETTINGS FOR DE, PSO, AND ASSOCIATED HOPIMS.

		$\mathcal{P}^{\mathrm{DE}}$				$\mathcal{P}^{\mathrm{PSO}}$			
Parameter	DE	Tr12S	Tr12A	gbmm12S	gbmm12A	Tr12S	Tr12A	gbmm12S	gbmm12A
P_C	.74	.68	.72	.70	.78				
F_M	1%	1%	1.4%	1%	1%				
IN		1	3	2	5	5	6	22	5
EMI		1	1	1	1	3	3	1	3
EP		2	1	1	2	2	2	1	2
IMI		1	1	1	1	1	1	1	2
MI		26%	14%	10%	12%	10%	12%	10%	22%

TABLE IV
PARAMETER SETTINGS FOR HEPIMS.

Parameter	$\mathcal{P}^{\mathrm{Het}}_{\mathrm{Tr12S}}$	$\mathcal{P}^{\mathrm{Het}}_{\mathrm{Tr12A}}$	$\mathcal{P}^{\mathrm{Het}}_{\mathrm{gbmm12S}}$	$\mathcal{P}^{\mathrm{Het}}_{\mathrm{gbmm12A}}$
IN	2	3	2	6
EMI	2	1	1	3
EP	2	2	2	1
IMI	3	3	3	3
MI	26%	10%	32%	14%

B. Analysis of Accuracy

Packages of one hundred unsigned permutations were randomly generated containing n genes of lengths $n \in \{100, 110, \ldots, 150\}$. Parameters were taken from Tables II, III and IV. For each permutation, the PIMs and the sequential versions were executed ten times starting from the same population for all permutations of the same length. Then, the averages of the ten results for each permutation and algorithm were calculated. These averages represent the number of reversals for each unsigned permutation.

The improvement of PIMs' accuracy is analyzed relative to the associated sequential BAs. The radar chart in Fig. 3 shows the accuracy of DE, GA, GAD and PSO. DE obtained the most accurate results while PSO provided the worst results. GAD and GA output competitive results, being GAD the second-placed BAs. The six radii of the radar chart represent the accuracy for inputs of size $100, 110$ to 150. Also, notice that the ranges and scales in each radius of the radar chart are different.

C. Accuracy for HoPIMs

Radar charts in Figs. 4 and 5 show how in all cases, the four HoPIMs implemented with each BA improve the accuracy of their sequential versions (see the smaller radar charts). Also, as expected the best and the worst accuracies were obtained by HoPIMs applying DE and PSO, respectively, as happens for the sequential BAs (see the ranges and scales in Fig. 4). This is not surprising, since PSO does not adapts well to discrete optimization problems as GA and GAD do, and DE adapts well to both continuous and discrete optimization problems and explores efficiently large search spaces as the one of URD. In Fig. 5, the ranges and scales in the radii of the charts are the same. An interesting observation is that the improvement of accuracy obtained by HoPIMs with GA is so expressive that the accuracy results for HoPIMs using GA surpassed substantially the results of HoPIMs using GAD. This contrasts with the fact that GAD gave better accuracy than GA (Fig. 3).

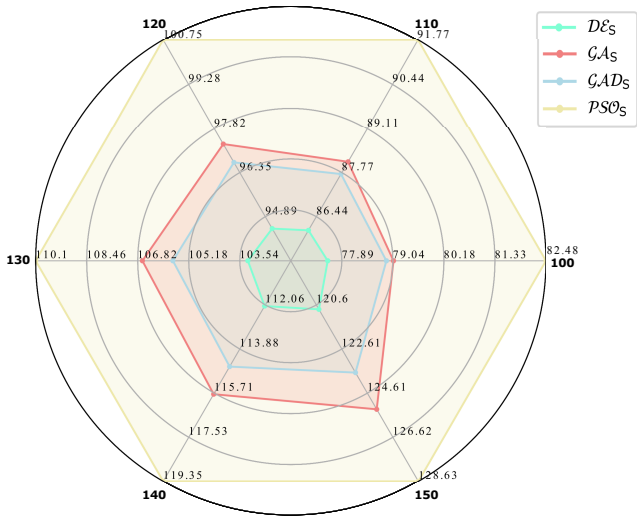

Fig. 3. Accuracy of the sequential BAs DE, GA, GAD and PSO.

D. Accuracy for HePIMs

Fig. 6 shows the accuracy for the asynchronous HePIM $\mathcal{P}^{\mathrm{Het}}_{\mathrm{Tr12A}}$. The chart on the left shows that the accuracy results are very competitive regarding the four HoPIMs with the same architecture ($\mathcal{P}^{\mathrm{DE}}_{\mathrm{Tr12A}}$, $\mathcal{P}^{\mathrm{GA}}_{\mathrm{Tr12A}}$, $\mathcal{P}^{\mathrm{GAD}}_{\mathrm{Tr12A}}$, and $\mathcal{P}^{\mathrm{PSO}}_{\mathrm{Tr12A}}$) being only surpassed by $\mathcal{P}^{\mathrm{DE}}_{\mathrm{Tr12A}}$. The chart on the right shows (dashed lines) the best final average results obtained by each set of three islands in $\mathcal{P}^{\mathrm{Het}}_{\mathrm{Tr12A}}$ that perform the same BA. From this chart, it is clear that the migration policy of the $\mathcal{P}^{\mathrm{Het}}_{\mathrm{Tr12A}}$ architecture successfully propagates the results obtained in all islands. Indeed, the average results in islands running the BAs GA, GAD and PSO outperform the accuracy obtained with their respective HoPIMs. These final average results are very close to those given by the best HoPIM, $\mathcal{P}^{\mathrm{DE}}_{\mathrm{Tr12A}}$, and by $\mathcal{P}^{\mathrm{Het}}_{\mathrm{Tr12A}}$.

The radar charts in Fig. 7 show the accuracy for the asynchronous HePIM $\mathcal{P}^{\mathrm{Het}}_{\mathrm{gbmm12A}}$. The chart on the left shows that the accuracy results of $\mathcal{P}^{\mathrm{Het}}_{\mathrm{gbmm12A}}$ outperform those of all HoPIMs with the same architecture except $\mathcal{P}^{\mathrm{DE}}_{\mathrm{gbmm12A}}$ but computing results very close to $\mathcal{P}^{\mathrm{DE}}_{\mathrm{gbmm12A}}$. The chart on the right, which uses different ranges in its radii, compares the quality of results obtained by $\mathcal{P}^{\mathrm{Het}}_{\mathrm{gbmm12A}}$ and the final average results of the four sets of islands running the same BA inside $\mathcal{P}^{\mathrm{Het}}_{\mathrm{gbmm12A}}$. Once again, it is clear that the migration policy combined with the diversity promoted by the application of different BAs in the islands of the $\mathcal{P}^{\mathrm{Het}}_{\mathrm{gbmm12A}}$ architecture succeeds in sharing good quality information among all islands giving to this HePIM the ability to find good quality solutions. In the right chart, one observes that the set of three islands running PSO finished with average results that have better quality than those computed by the

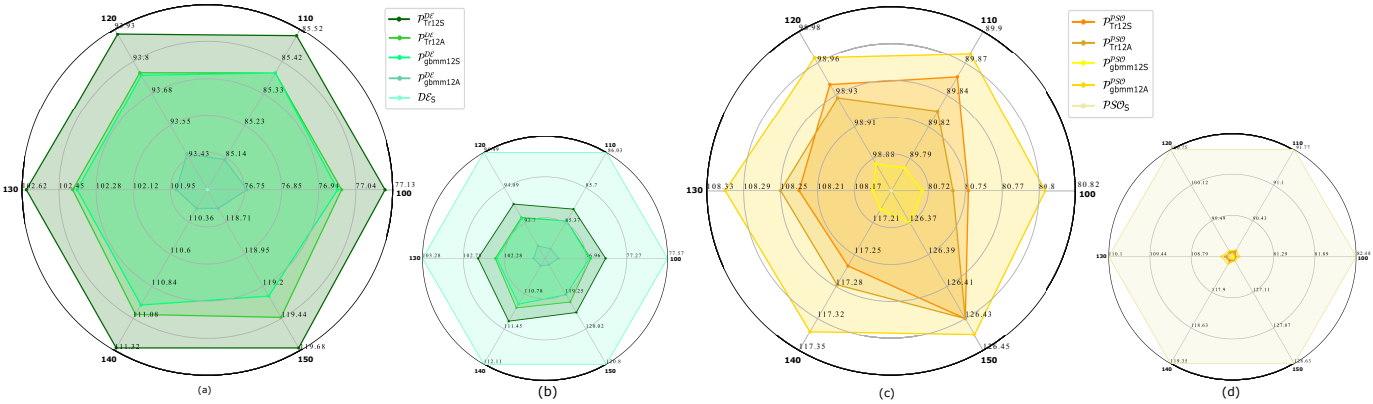

Fig. 4. Accuracy of (a) DE HoPIMs and (b) relative to DE, (c) PSO HoPIMs and (d) relative to PSO.

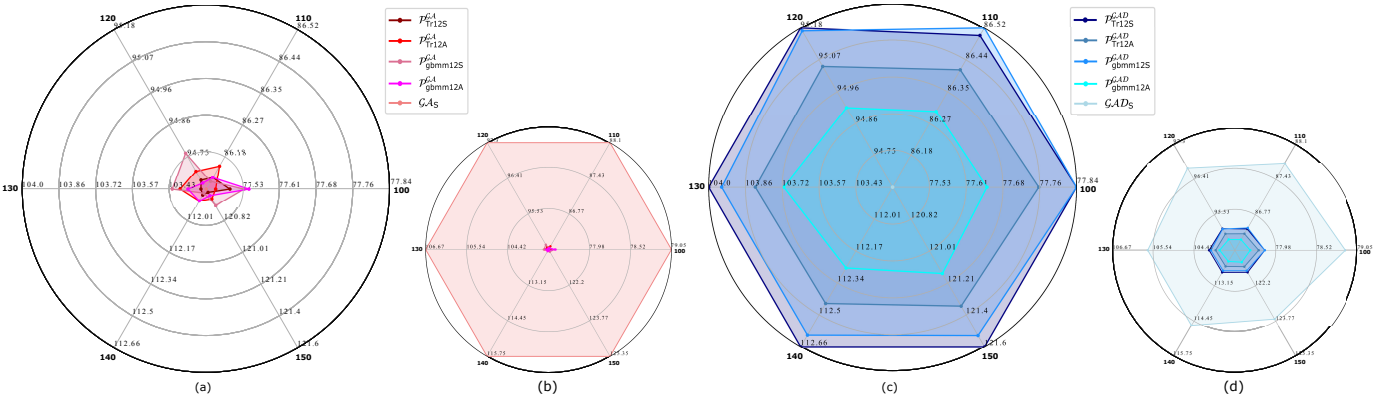

Fig. 5. Accuracy of: (a) GA HoPIMs and (b) relative to GA, (c) GAD HoPIMs and (d) relative to GAD.

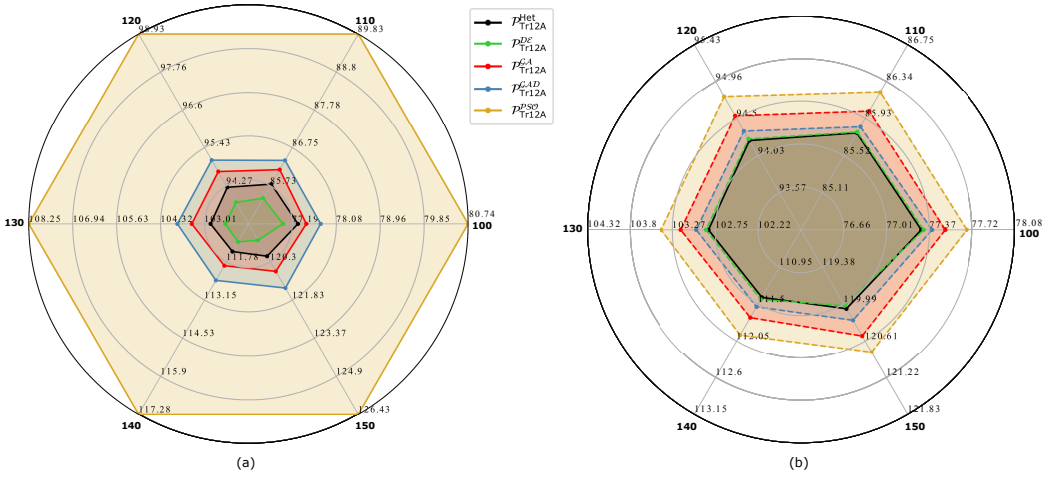

Fig. 6. (a) Accuracy of $\mathcal{P}^{\text{Het}}_{\text{Tr12A}}$ and related HoPIMs; (b) Accuracy of $\mathcal{P}^{\text{Het}}_{\text{Tr12A}}$ and average results of each set of islands in $\mathcal{P}^{\text{Het}}_{\text{Tr12A}}$ running the same BA.

sets of islands executing GA and GAD, swapping the quality performance observed in the respective HoPIMs.

Radar charts in Fig. 8 compare the quality of the results obtained by the synchronous architectures $\mathcal{P}^{\text{Het}}_{\text{Tr12S}}$ and $\mathcal{P}^{\text{Het}}_{\text{gbmm12S}}$ with HoPIMs with the same architecture. In both cases, the accuracy of the heterogeneous models is better than those of the corresponding homogeneous models using BAs GA, GAD and PSO (in this order) and very close to the results obtained by the best homogeneous models that are $\mathcal{P}^{\text{DE}}_{\text{Tr12S}}$ and $\mathcal{P}^{\text{DE}}_{\text{gbmm12S}}$, respectively. Also, the results provided by $\mathcal{P}^{\text{Het}}_{\text{Tr12S}}$ and $\mathcal{P}^{\text{Het}}_{\text{gbmm12S}}$ are indistinguishable (the ranges and scales in the radii of the charts in Fig. 8 are the same). For these synchronous models, the average best accuracy of the four sets of islands running the same BA are equal to the output of the models. This is explained because in the synchronous process, even with the same migration interval as in asynchronous models, the

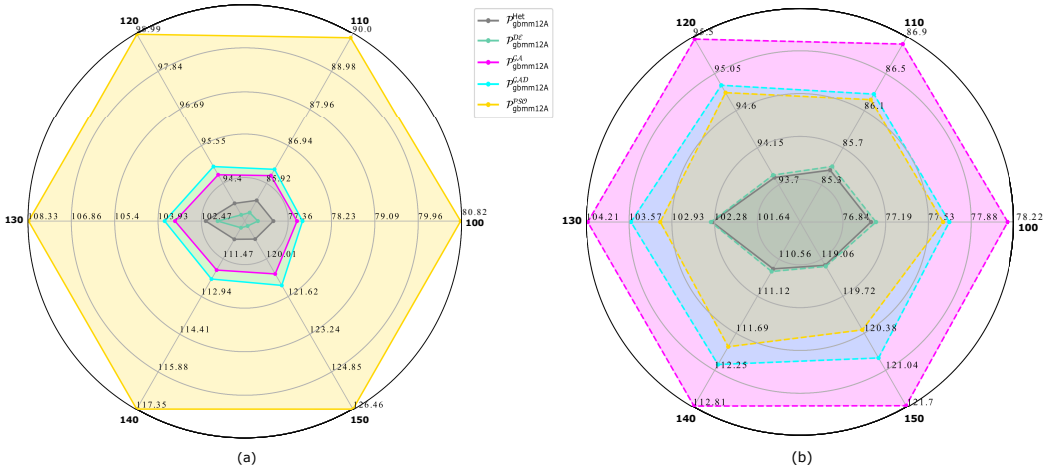

Fig. 7. (a) Accuracy of $\mathcal{P}^{\text{Het}}_{\text{gbmm12A}}$ and related HoPIMs; (b) Accuracy of $\mathcal{P}^{\text{Het}}_{\text{gbmm12A}}$ and average results of each set of islands in $\mathcal{P}^{\text{Het}}_{\text{gbmm12A}}$ running the same BA.

synchrony of evolution (i.e., all islands perform synchronously one generation of their BA) makes that island evolve with less diversity than in the asynchronous models.

The chart in Fig. 9 shows the accuracy of the four HePIMs: $\mathcal{P}^{\text{Het}}_{\text{gbmm12A}}$ gives the best while $\mathcal{P}^{\text{Het}}_{\text{Tr12A}}$ the worst accuracy. Models $\mathcal{P}^{\text{Het}}_{\text{Tr12S}}$ and $\mathcal{P}^{\text{Het}}_{\text{gbmm12S}}$ compute very similar quality outputs.

E. Statistical Analysis

Statistical tests were performed to validate the PIMs experiments with the significance level used in the tests of 95% represented in the tests as $\alpha = 0.05$. The sample of each PIM is the set of 100 outputs from the inputs used in Section IV-B. First, the Friedman test was applied to find the control algorithm. Subsequently, Holm's test (Post-hoc test) is applied to test the null hypothesis that the performance of the control algorithm is the same in relation to the remaining algorithms. This technique is proposed in [31] (also see [32] and [33]). The tests use $\mathcal{P}^{\text{DE}}_{\text{gbmm12A}}$ as the control algorithm for all inputs.

V. Conclusions and Future Work

Synchronous and asynchronous homogeneous and heterogeneous models were proposed from four BAs: `GA`, `GAD`, `PSO` and `DE` using a static binary tree topology and a dynamic topology based on diversity and population quality.

Our experiments, evaluated statistically, showed that in HePIMs, island diversity and migration policy are effective enough to reach quality results that, in general, outperform the quality of HoPIMs. However, they also show an exception: the HoPIM applying `DE`, which offered better efficiency, delivering better solutions for most input sets. Our experiments showed that the general superiority of HePIMs over HoPIMs could be guaranteed through careful calibration of the migration parameters to assure the efficient exploration and dissemination of the variety of individuals among islands that run the different BAs. These observations let us infer that the groups of islands using `GA`, `GAD` and `PSO` significantly improve the quality of their individuals driven by the genetic material disseminated by immigrants proceeding from islands using `DE`.

The diversity inherent to the asynchronous heterogeneous dynamic model was relevant in obtaining the best results among all HePIMs. Like other PIMs, it promotes diversity through a dynamic communication topology between islands and the heterogeneous application of different BAs in their islands. But in addition to these two factors, the asynchronous evolution, guided by different BAs in groups of islands, increments diversity improving its native individuals. In contrast, synchronous evolution restricts diversity and, consequently, impoverishes the results' accuracy.

An important observation is a contrast regarding the evolutionary process. Indeed, the experiments showed that synchronous models achieved the best results with HoPIMs, whereas asynchronous HePIMs showed better exploration of the search space and consequently provided better solutions.

Future work will implement flexible HePIMs promoting an island-localized migration process; each island could have its migration parameters, including the BA, to be executed. Current work explores the design of HePIMs that detect the BA that adapts better to the optimization objectives. Then each island runs the best-adapted BA to deal with its objective.

References

[1] L. A. da Silveira, J. L. Soncco-Álvarez, T. A. de Lima, and M. Ayala-Rincón, "Behavior of Bioinspired Algorithms in Parallel Island Models," in *IEEE Cong. on Evol. Comp. (CEC)*, 2020.

[2] J. H. Holland, "Genetic Algorithms and the Optimal Allocation of Trials," *SIAM Journal on Computing*, vol. 2, no. 2, pp. 88–105, 1973.

[3] R. Storn and K. Price, "Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces," *J. of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[4] R. Eberhart and J. Kennedy, "Particle swarm optimization," in *IEEE Int. Conf. on neural networks*, vol. 4, pp. 1942–1948, 1995.

[5] A. Caprara, "Sorting by reversals is difficult," in *Proc. of the first ACM annual Int. Conf. on Comp. molecular Biology*, pp. 75–83, 1997.

[6] T. A. de Lima and M. Ayala-Rincón, "On the average number of reversals needed to sort signed permutations," *Discrete Applied Mathematics*, vol. 235, no. Supplement C, pp. 59 – 80, 2018.

[7] J. Kececioglu and D. Sankoff, "Exact and approximation algorithms for the inversion distance between two chromosomes," in *Proc. Combinatorial Pattern Matching (CPM)*, pp. 87–105, Springer, 1993.

[8] G. Duarte, A. Lemonge, and L. Goliatt, "A New Strategy to Evaluate the Attractiveness in a Dynamic Island Model," in *IEEE Cong. on Evol. Comp. (CEC)*, 2018.

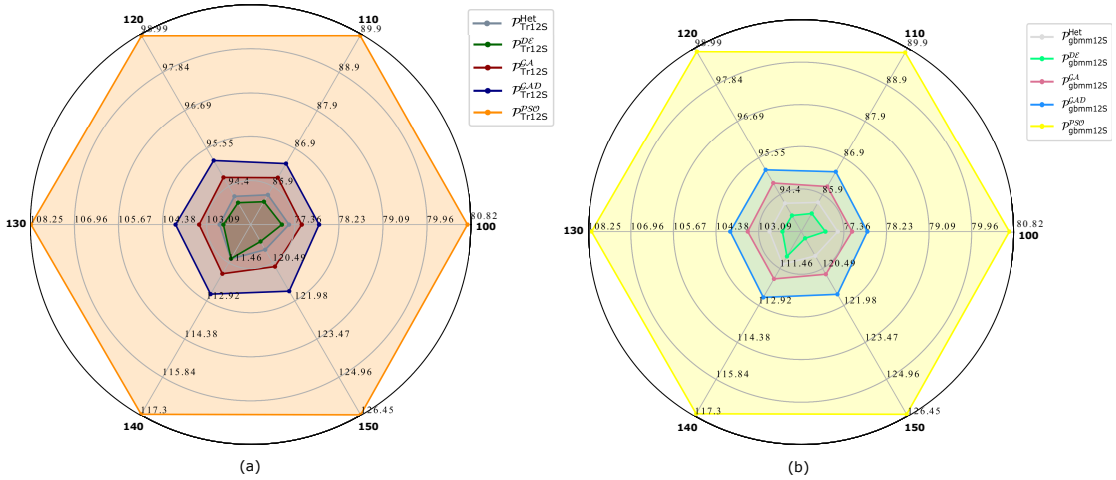

Fig. 8. Accuracy of $\mathcal{P}^{\text{Het}}_{\text{Tr12S}}$ (a) and $\mathcal{P}^{\text{Het}}_{\text{gbmm12S}}$ (b) and HoPIMs with the same architecture.

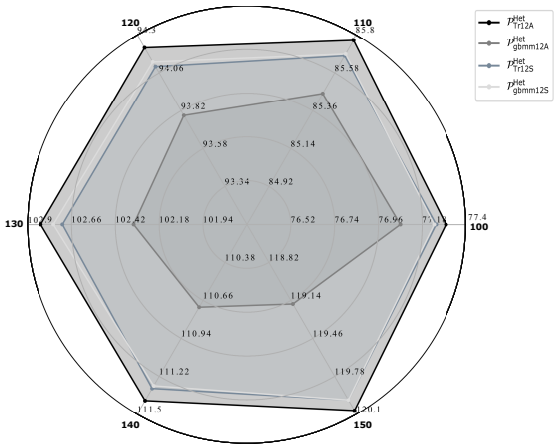

Fig. 9. Accuracy of the HePIMs

[9] G. R. d. C. L. Duarte, A. C. da Fonseca, L. G. de Lima, and B. S. L. Pires, "An Island Model based on Stigmergy to solve optimization problems," *Natural Computing*, pp. 1–29, 2020.

[10] T. G. Crainic and M. Toulouse, *Parallel Strategies for Meta-Heuristics*, pp. 475–513. Boston, MA: Springer US, 2003.

[11] M. Ruciński, D. Izzo, and F. Biscani, "On the impact of the migration topology on the island model," *Parallel Computing*, vol. 36, no. 10, pp. 555–571, 2010. Parallel Architectures and Bioinspired Algorithms.

[12] D. Sudholt, *Springer Handbook of Computational Intelligence*, ch. Parallel Evolutionary Algorithms, pp. 929–959. Springer, 2015.

[13] S. Hannenhalli and P. A. Pevzner, "Transforming cabbage into turnip: Polynomial algorithm for sorting signed permutations by reversals," *J. of the ACM*, vol. 46, no. 1, pp. 1–27, 1999.

[14] K. Yasuda and K. Yazawa, "Parameter self-adjusting strategy for particle swarm optimization," in *11th Int. Conf. on Intelligent Systems Design and Applications*, pp. 265–270, IEEE, 2011.

[15] D. Bader, B. Moret, and M. Yan, "A linear-time algorithm for computing inversion distance between signed permutations with an experimental study," *J. of Comp. Biology*, vol. 8, no. 5, pp. 483–491, 2001.

[16] R. Bianchini and M. C. Brown, "Parallel Genetic Algorithms on Distributed-Memory Architectures," in *NATUG-6: Proc. of the Sixth Conf. of the North American Transputer Users Group on Transputer Research and Applications 6*, IOS Press, 1993.

[17] S. C. Lin, W. F. Punch, and E. D. Goodman, "Coarse-grain parallel genetic algorithms: categorization and new approach," in *Proc. 6th IEEE Symp. on Parallel and Distributed Processing*, pp. 28–37, 1994.

[18] D. Izzo, M. Rucinski, and C. Ampatzis, "Parallel global optimisation meta-heuristics using an asynchronous island-model," in *IEEE Cong. on Evol. Comp. (CEC)*, pp. 2301–2308, 2009.

[19] Y. Gong and A. Fukunaga, "Distributed island-model genetic algorithms using heterogeneous parameter settings," in *IEEE Cong. on Evol. Comp. (CEC)*, pp. 820–827, 2011.

[20] P. V. S. Z. Capriles, L. G. Fonseca, H. J. C. Barbosa, and A. C. C. Lemonge, "Rank-based ant colony algorithms for truss weight minimization with discrete variables," *Communications in Numerical Methods in Engineering*, vol. 23, no. 6, pp. 553–575, 2007.

[21] J. Liang, B. Qu, P. Suganthan, and Q. Chen, "Problem definitions and evaluation criteria for the CEC 2015 competition on learning-based real-parameter single objective optimization," tech. rep., Zhengzhou University, 2014.

[22] L. A. da Silveira, J. L. Soncco-Álvarez, and M. Ayala-Rincón, "Parallel memetic genetic algorithms for sorting unsigned genomes by translocations," in *IEEE Cong. on Evol. Comp. (CEC)*, pp. 185–192, 2016.

[23] L. A. da Silveira, J. L. Soncco-Álvarez, T. A. de Lima, and M. Ayala-Rincón, "Computing translocation distance by a genetic algorithm," in *2015 Latin American Computing Conf. (CLEI)*, pp. 1–12, 2015.

[24] L. A. da Silveira, J. L. Soncco-Álvarez, T. A. de Lima, and M. Ayala-Rincón, "Parallel multi-island genetic algorithm for sorting unsigned genomes by reversals," in *IEEE Cong. on Evol. Comp. (CEC)*, 2018.

[25] L. A. da Silveira, J. L. Soncco-Álvarez, T. A. de Lima, and M. Ayala-Rincón, "Parallel Island Model Genetic Algorithms applied in NP-Hard problems," in *IEEE Cong. on Evol. Comp. (CEC)*, 2019.

[26] Y. Zhang, D. Gong, and Z. Ding, "Handling multi-objective optimization problems with a multi-swarm cooperative particle swarm optimizer," *Expert Systems with Applications*, vol. 38, pp. 13933–13941, 2011.

[27] D. Gong, B. Xu, Y. Zhang, Y. Guo, and S. Yang, "A Similarity-Based Cooperative Co-Evolutionary Algorithm for Dynamic Interval Multiobjective Optimization Problems," *IEEE Trans. on Evol. Comp.*, vol. 24, no. 1, pp. 142–156, 2020.

[28] B. Xu, Y. Zhang, D. Gong, Y. Guo, and M. Rong, "Environment Sensitivity-Based Cooperative Co-Evolutionary Algorithms for Dynamic Multi-Objective Optimization," *IEEE/ACM Trans. on Comp. Biology and Bioinformatics*, vol. 15, no. 6, pp. 1877–1890, 2018.

[29] R. Hashimoto, H. Ishibuchi, N. Masuyama, and Y. Nojima, "Analysis of Evolutionary Multi-Tasking as an Island Model," in *Proc. of the Genetic and Evol. Comp. Conf. (GECCO)*, pp. 1894–1897, ACM, 2018.

[30] A. Eiben and S. Smit, "Parameter tuning for configuring and analyzing evolutionary algorithms," *Swarm and Evol. Comp.*, vol. 1, no. 1, pp. 19–31, 2011.

[31] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. of Machine Learning Research*, vol. 7, pp. 1–30, 2006.

[32] S. García and F. Herrera, "An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons," *J. of Machine Learning Research*, vol. 9, pp. 2677–2694, 2008.

[33] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evol. and swarm intelligence algorithms," *Swarm and Evol. Comp.*, vol. 1, no. 1, pp. 3–18, 2011.