# On Reconfiguring Heterogeneous Parallel Island Models*

Lucas A. da Silveira[a], Thaynara A. de Lima[b], Mauricio Ayala-Rincón[a,c,*]

[a]*Department of Computer Science, Universidade de Brasília, DF, 70900–010 Brazil*
[b]*Department of Mathematics, Universidade Federal de Goiás — Campus II, Goiânia, Goiás, 74690–900 Brazil*
[c]*Department of Mathematics, Universidade de Brasília, DF, 70900–010 Brazil*

**Abstract**

This work introduces a new Parallel Island Model (PIM) that encompasses the benefits of heterogeneity and algorithmic reconfigurability. The former feature, heterogeneity, allows the execution of different evolutionary algorithms on the islands, increasing the usual diversity obtained by the communication topologies and migration policies by homogeneous PIMs (HoPIMs). Previous heterogeneous PIMs (HePIMs) were able to provide competitive solutions regarding the HoPIMs. The latter feature, the capability of reconfiguration, empowers PIMs to change dynamically from the execution of one evolutionary algorithm to another. In this manner, the required diversity and flexibility to outperform HoPIMs and HePIMs is achieved. This paper discusses policies to profit from the feature of reconfigurability on HePIM models and provides an innovative and successful stagnation-based reconfiguration policy. The benefits of the new reconfigurable model are verified using the unsigned reversal distance optimization problem as a case study.

*Keywords:* Island-based Bioinspired Algorithms, Parallel Island Models, Heterogeneous Models, Reconfigurability, Evolutionary Algorithms, Unsigned Reversal Distance
*2000 MSC:* 65Y05, 68W25, 03D15, 90C27, 92D15

*Research funded by the Brazilian agencies CNPq (Universal 409003/21-2); FAPDF (DE 00193-00001175/2021-11); and FAPEG (CAP 2022061000121).

*Corresponding author. Partly supported by CNPq productivity grant 313290/21-0.
*Email addresses:* lucasangelosilveira2018@gmail.com (Lucas A. da Silveira), thaynaradelima@ufg.br (Thaynara A. de Lima), ayala@unb.br (Mauricio Ayala-Rincón)

## 1. Introduction

Many complex optimization problems in engineering, biology, and social sciences cannot be solved by exact optimization methods such as function extreme methods or mathematical programming. In such contexts, approximation algorithms and heuristics that provide near-optimal solutions in reasonable time and allow parallelism are of great interest. Among such techniques, we are interested in parallelizing bio-inspired algorithms (BAs) to solve $\mathcal{NP}$-hard problems. BAs have adaptive search capabilities mimicking the evolutionary process of biological systems and can compute good-quality and satisfactory solutions for practical applications. However, optimization through such BA approaches demands exhaustive computations and efficient use of resources. Parallelization is one of the natural mechanisms applied to speed up and improve the accuracy of solutions obtained by BAs. In this context, parallel island models (PIMs) are an efficient and easily adaptable approach to dealing with $\mathcal{NP}$-hard problems. A PIM subdivides the population among their islands (processors) and simultaneously runs a BA on each island. The exchange of individuals promotes migration between islands, improving the global population. The result is the most adapted individual among all the islands, regarding a fitness function defined from the characteristics of the problem.

### *1.1. Homogeneous and Heterogeneous Parallel Island Models*

PIMs in which all islands run the same BA are called homogeneous PIMs (HoPIMs). We proposed different HoPIM-based approaches to solve different complex combinatorial problems, obtaining highly competitive results regarding performance and accuracy [1, 2, 3, 4].

Heterogeneous PIMs (HePIMs) are PIMs in which each island may execute a different BA [5]. Since this feature promotes higher diversity (than the one obtained by HoPIMs), such models encourage the exploration of migration policies and island topologies that outperform the solutions obtained by homogeneous models. Indeed, we have explored HePIMs that provide competitive solutions to different $\mathcal{NP}$-hard problems regarding those computed by HoPIMs. Despite this fact, no HePIM was able to outperform all homogeneous models [6].

### *1.2. Contribution*

To improve the diversity and flexibility of HePIMs, to outperform the accuracy of results computed by the best-adapted HoPIMs (regarding [4, 6]), the feature of reconfigurability was first added to HePIMs in [7], obtaining encouraging and competitive results. In such models, each island may run a

different BA and update it to the BA being executed by the island with the best performance, and the reconfiguration process was periodically performed after a fixed number of generations during the whole evolutionary process. However, the periodic reconfiguration policy proposed in [7] does not provide the required diversity to outperform the best-adapted HoPIMs.

This paper introduces reconfigurable HePIMs with a so-called stagnation-based reconfiguration policy providing the required diversity. The algorithmic reconfiguration is applied continuously if island stagnation is detected. Such improvement in the dynamicity and flexibility of the reconfiguration phase is possible by maintaining a record of the algorithm executed by the most evolved island in a previous period of generations of the evolutionary process. The early reconfigurable PIMs proposed in [7] were refined after exhaustive experiments. Maintaining such a record eliminates the need to exchange information between the islands to decide how each island should reconfigure its BA.

The problem addressed in this paper is the unsigned reversal distance problem (URD), an $\mathcal{NP}$-hard problem highly applied in comparative genomics. For comparison matters, we select the best-adapted HoPIMs to URD from [4] and the HePIMs that performed better from [6]. The HePIMs run three different BAs: simple genetic algorithm `GA` [8], double-point crossover genetic algorithm `GAD` [9], and `DE` [10], and the best-adapted HoPIMs run `DE` in all its islands. Experiments are performed using two different topologies, a static three topology and a dynamic complete graph topology.

The design decisions used by the stagnation-based reconfigurable HePIMs, introduced in this paper, show that empowering HePIMs with the reconfiguration feature opens an exciting space for investigation since the accuracy of the computed solutions outperforms the results obtained by all previous PIMs.

In addition, this paper analyzes the algorithm convergence of the stagnation-based reconfigurable PIMs implementing mechanisms to track island reconfiguration. They identify in which phase of the evolutionary process the model becomes homogeneous. Curiously, not in all cases, the convergence is towards the best-adapted HoPIM reported in [4, 6].

### *1.3. Organization*

Sec. 2 presents the unsigned reversal distance problem, the three selected BAs, and PIMs. Sec. 3 presents related work, and Sec. 4.1 introduces the new reconfigurable HePIMs and explaining how the reconfiguration works. Then, Sec. 5 presents experiments, discusses accuracy results, performance, and statistical analysis. Finally, Sec. 6 concludes and discussed future work.

To assure reproducibility, source and data used in the experiments are available at http://genoma.cic.unb.br.

## 2. Background

### *2.1. Case study*

The evolutionary distance between two organisms can be computed as the number of rearrangements needed to transform a genome into another using some evolutionary measure. This work considers the minimum number of reversals to compute the distance between unichromosomal organisms.

Permutations on $\{1, \cdots, n\}$ represent a genome containing $n$ genes. Given a genome $\pi = (\pi_1, \pi_2, ..., \pi_n)$, where $1 \leq i, \pi_i \leq n$, a reversal $\rho^{j,k}$, for $1 \leq j \leq k \leq n$, transforms $\pi$ into $\pi' = (\cdots, \pi_{j-1}, \pi_k, \cdots, \pi_j, \pi_{k+1}, \cdots)$, that is, it inverts the elements between $\pi_j$ and $\pi_k$. Genes may have forward or reverse orientation on a genome; if known, genes receive a positive or negative sign, respectively, and the genome is a signed permutation.

There are two evolutionary problems related to computing the distance by reversals. The signed reversal distance (SRD) problem asks for the minimum number of reversals needed to transform a signed permutation into another. Moreover, the unsigned reversal distance (URD) problem, addressed in this paper, consists of computing such a number between unsigned permutations in which the orientation of genes is unknown.

It is well-known that SRD belongs to class $\mathcal{P}$ [11], whereas URD is an $\mathcal{NP}$-hard problem [12]. Approximated solutions to solve an URD instance can be obtained by computing exact solutions to SDR instances resulting from attributing signs to the genes in the original URD instance [13, 14]. The fitness used by our models is linearly computed over signed permutations by applying Bader's exact algorithm to solve SRD [15].

### *2.2. Local Evolutionary Engines — bio-inspired Algorithms*

The reconfigurable models, introduced in this paper, use three BAs, widely applied to solve optimization problems. These BAs present distinct adaptability characteristics.

- Simple Genetic Algorithm (`GA`). `GA` was introduced by J. H. Holland [8]. This algorithm evolves a population by considering a breeding cycle where the best individuals are selected and produce offspring by applying one-point crossover (Fig. 1 (a)). Then, the descendants replace the worst individuals in the current population. The breeding cycle relies on four parameters; namely, the percentages of *selection* and *replacement*, and the probability of application of *mutation* and *crossover*.

- Double-point Crossover Genetic Algorithm (`GAD`). This algorithm has a similar behavior than `GA` except by the technique to promote *crossover*, illustrated in Fig. 1 (b), and how the population evolves: in contrast to `GA`, `GAD` offspring replace individuals randomly selected.
- Differential Evolution (`DE`). `DE` was roposed by Storn and Price [10]. It is a method to optimize functions over the real multidimensional space $\mathbb{R}^n$. We adapt the algorithm by restricting the domain of the function as the set of permutations. The evolutionary process is guided by the *mutation factor* $F_M$, applied to individuals randomly selected (from the population) to generate mutants, and the *probability of crossover* $P_C$. The local population evolves by replacing individuals having the worst fitness with mutants.

To adapt `DE` to URD, each $n$-dimensional individual $v$ is represented by a numerical vector with values in the range $[0, 1]$, which is associated with the permutation $\pi = (\pi_1, \ldots, \pi_n)$ given as input. If the $i-$th entry of $v$ belongs to the interval $[0, 0.5)$ then $\pi_i$ receives a negative orientation; otherwise, if it belongs to the interval $[0.5, 1]$, $\pi_i$ is assigned positively. In the end, a signed permutation is built from $v$ and $\pi$.

For `GA` and `GAD`, the orientation of the genes in each individual is randomly generated as $\pm 1$. After the transformation of an unsigned to a signed permutations, the linear exact algorithm to solve the SRD problem, proposed by Bader *et al.* [15], computes the fitness of each individual.

Although `GA` and `GAD` are variants of the genetic algorithm, it is well-known that they have distinct combinatorial behaviours [9].

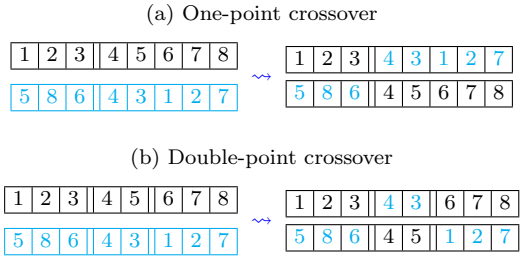


Figure 1: One-point and double-point crossing operators.

### *2.3. Parallel island model (PIM)*

PIMs have been widely used to improve the performance of Evolutionary Algorithms [16]. In PIMs, the population is distributed into islands, running their BAs in parallel. The connection between the islands establishes the model's topology. PIM's topologies can be classified as static or dynamic. *Static* PIMs maintain the connections fixed during the execution, whereas

*dynamic* models admit changes during the process. Analysis of $\mathcal{NP}$-hard problems from PIMs based on different topologies is available (e.g. [17, 4, 18]).

*Homogeneous* PIMs execute the same BA in all islands, whereas *heterogeneous* models admit different BAs running in their islands.

Connected islands exchange individuals to evolve, and a *migration policy* guides individuals exchange between islands during the evolutionary process. PIMs have tuned breeding cycles and migration parameters to improve the solutions' quality. The following briefly presents the migration parameters used in the PIMs considered in this work. Some of them take into account the classification of individuals as **best**, **worst**, and **random**, based on a rank established according to their fitness, in increasing order. The first half of the rank corresponds to the best, whereas the second half to the worst individuals; random individuals are selected randomly.

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

PIMs are also classified according to the synchroneity in which islands evolve their population. In *Synchronous* PIMs, islands evolve performing each generation simultaneously, whereas, in *asynchronous* PIMs, islands' evolution occurs independently, even during migration. The latest mimics the behavior found in nature.

## 3. Related Work

The discussion in this section is restricted to previous work that boosted the evolution of parallel island models.

Duarte *et al.* [19] proposed a migration policy for PIMs called DIM-1 with target islands defined by attractiveness. The migrations are synchronous to occur *point to point*, where links between islands are unidirectional and weighted dynamically according to the local's attractiveness to the target island. The weights represent the probability of each communication being used for migration. Afterward, they presented a new evaluation strategy in [20] called DIM-2 that changes how to define the island's attractiveness so that islands become more or less attractive according to their solutions' quality. The authors in [21] proposed HePIMs based on strategies DIM-1 and DIM-2 using as an evolutionary engine only variations of `DE`, which had a successful history in competitions held by the CEC. The results presented demonstrated that the HePIMs from the DIM-1 and DIM-2 can produce better results than the HoPIM versions. Duarte *et al.* also propose tweaks to the migration policy in [17] by adjusting how the attractiveness and weights of island connections are calculated. The adjustments were inspired by the natural phenomenon known as stigmergy [22]. The dynamic HePIMs, presented in the current work add to the heterogeneity and the dynamism of the migration policy, the algorithmic flexibility obtained by dynamic reconfiguration.

Qinxue *et al.* [23] implemented dynamism in PIM using GA through spectral clustering. The authors do not have a fixed number of islands but limit the number to ten islands. The model is initialized with an island and starts to evolve. During the evolutionary process, whenever there is a migration "epoch" (migration phase), individuals are grouped by similarity using spectral clustering, giving rise to new islands. Similar individuals are assigned to the same island. The standard model is compared with traditional models showing its satisfactory performance. Among the benefits, the authors highlight a reduction of the workload, in contrast to other methods in a parallel environment that usually implement each island allocated to a processor and exchange individuals in the migration phases via messages. Once again, the migration policy of this approach is dynamic as ours but does not add the power of algorithmic reconfiguration. Moreover, our models maintain a regularity regarding the population size on each island and the distribution of the processors over the islands aiming to explore parallelism as efficiently as possible.

Hashimoto *et al.* [24] proposed a HePIM to solve multi-task problems, where each island evaluates an objective. Migrants are selected at high migration frequency and removed randomly on each local island, replacing the worst individuals in the target islands. Since emigrants went to islands responsible for different objectives, their fitness values are the worst, assuming they have fewer chances of being suitable for the target island objective. The

current work applies migration policies shared by Hashimoto *et al.* focusing on consolidating reconfigurability as a new influential parameter to be considered in the design of HePIMs. Despite this restriction, it is clear that the proposed model's heterogeneity is relevant to multi-objective optimization since different BAs adapt better to various optimization tasks.

Lardeux *et al.* [25] study dynamic island models with a focus on migration policies at the level of individuals. To control migrations, the authors combine migration and gain matrices that are updated during the search, allowing the simultaneous use of different migration policies within the same model. Thus, individuals cooperate and share their information throughout the evolutionary process. The choices of the best migrations are improved using the QLearning approach, which aims to learn the best migration options by each user. The results show that the QLeaning significantly improves the accuracy of the results. A parallel study addresses the context of population size versus the number of generations, where results show that increasing the number of individuals and reducing the number of generations does not benefit the quality of the results. As in Qinxue *et al.* clustering-based approach, in contrast to ours, the Lardeux *et al.* method allows variation of island population size through the selection of individual migration choices, but it does not apply the flexibility of reconfiguration.

From our side, Silveira *et al.* [6] proposed a variety of static HePIMs running four different BAs on their islands. HePIMs maintained population diversity by covering the solution space and reducing overlap between islands compared to HoPIMs. In [7], the authors go a step forward, maintaining the population diversity provided by HePIMs, and increasing their flexibility, allowing BA reconfiguration on islands during execution according to their performance, where the islands may substitute their BAs periodically during the evolutionary process. Results were competitive regarding the best-adapted HoPIMs, demonstrating the potential of adding such (periodic) reconfiguration capability to HePIMs. After a series of exhaustive experiments, the current work shows how refining the dynamism of reconfiguration, reaching the stagnation-based reconfiguration approach, such innovative ability adds the required flexibility and diversity to HePIMs to outperform the best-adapted HoPIMs.

Nssibi et al. [26] survey nature-inspired metaheuristic methods in the field of machine learning, more specifically for feature selection. In this perspective, the authors include a section highlighting the strength of island models in recent works to solve the feature selection problem. The authors discuss works related to homogeneous island models, such as the Harris Hawk Optimization algorithm with islands organized over master-slave communication topology. The slave processors have a swarm of hawks, which send

the best solutions to the master island when the results of their local optimization are available. Also, they survey models with a moth flame optimization algorithm using random ring topology with an elitist policy: the best immigrants replacing the worst natives. A multi-objective evolutionary procedure with an evolutionary engine based on the non-dominated sorting GA is also considered, where the islands are organized through a dynamic complete graph and elitist migration policy. Moreover, they survey parallel PSO models where a ring topology organizes the islands, and the migration policy consists of sending the best individual and only replacing the worst native if the immigrant has better fitness. The survey does not address island-based algorithm-adaptative or algorithmic reconfigurable mechanisms as those introduced in the reconfigurable island models mentioned in this paper. Hence, we believe that the innovative reconfigurable PIMs here introduced constitute a significant way for further investigations to improve island-based bio-inspired algorithms.

## 4. Reconfigurable HePIMs with stagnation policy

### *4.1. Communication Topologies*

The introduced stagnation-based reconfigurable models select one static, and one dynamic topology that successfully addressed URD in [4, 7].

The static topology is a 12-island bi-directional binary tree, and the dynamic topology is the 12-island complete graph (see Fig. 2).

In the complete graph topology all pairs of islands may exchange individuals. The island communication dynamism is acquired by exploring the diversity and quality of 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. The 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*).

### *4.2. Reconfigurable islands*

Reconfigurable HePIMs were proposed initially in [7]. The dynamic complete graph model classifies islands according to their fitness average and variance and uses it to build neighborhoods during the evolutionary process. A master island is responsible for receiving data from all islands; it ranks and notifies the worst island. The worst island reconfigures (i.e., replaces) its algorithm with the algorithm of the best-ranked island. Furthermore, the reconfiguration process is controlled by a parameter called reconfiguration

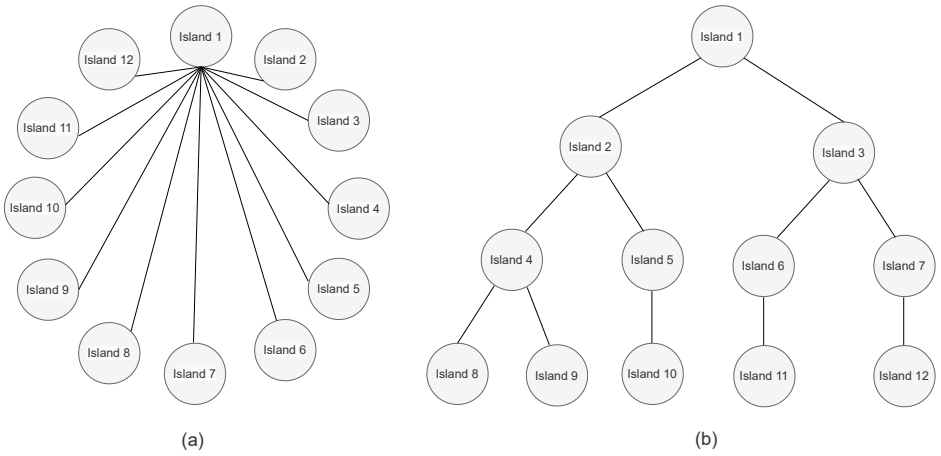


Figure 2: (a) A dynamic complete graph topology. (b) binary tree topology. In (a,) for simplicity, only a subset of the edges between all islands are included.

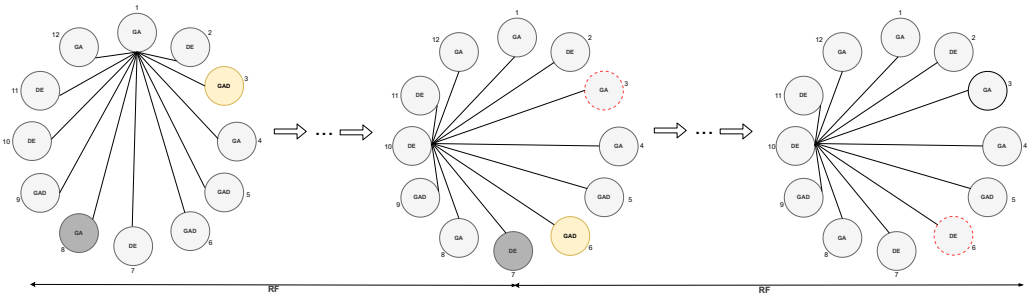


Figure 3: Example of periodic reconfiguration in the dynamic complete graph topology. For simplicity, only a subset of the edges between all islands are included. In each reconfiguration cycle, according to the number of generations defined by the parameter RF, the gray island has the best performance, and the red-dotted island, with the worst performance, has its BA updated to the BA being executed by the gray island.

frequency *(RF)*. The *RF* establishes the number of evolutionary generations to (periodically) perform a reconfiguration step. The dynamic reconfiguration allows updating the BAs executed in their islands. In [7], authors implemented two HePIMs: $\mathcal{P}^{\text{recHet}}_{\text{Tr12A}}$ and $\mathcal{P}^{\text{recHet}}_{\text{gbmm12A}}$. The former uses the static binary tree topology, and the latter uses the dynamic complete graph topology; both models are asynchronous and evolve through a refined migration policy that allows the exchange of individuals, maintaining diversity. Fig. 3 shows reconfiguration cycles, taking the $\mathcal{P}^{\text{recHet}}_{\text{gbmm12A}}$ model as an example.

*4.3. Stagnation-based Reconfigurable HePIMs*

This paper refines the models proposed initially in [7]. Instead of performing a reconfiguration phase after a fixed number of generations (given by the parameter RF), the new model may reconfigure the current BA in each island to the best BA in all evolutionary generations (See Figure 4). The best BA is the BA used by the island showing the highest progress in a fixed interval of generations. This interval is given by a BA classification interval (BACI) parameter. Exhaustive experiments showed that a good metric to evaluate the progress of the islands is granted by the difference between the best individual fitness at the beginning and the end of the interval.

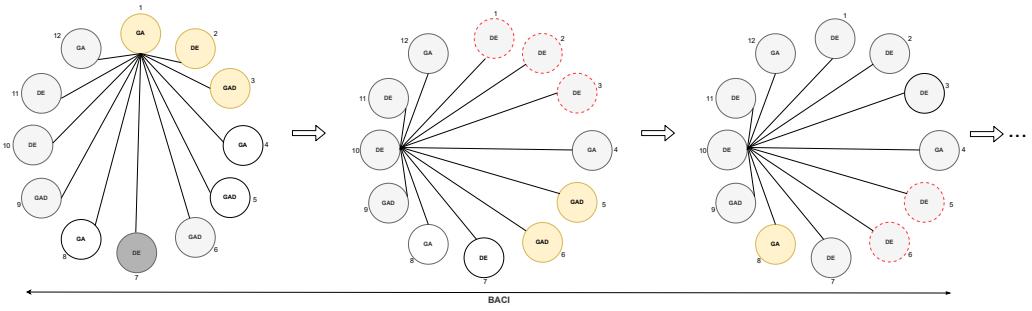


Figure 4: Example of stagnation-based reconfiguration on the complete graph topology. For simplicity, only a subset of the edges between all islands are included. The yellow islands represent stagnated islands in each generation that have undergone reconfiguration to the BA algorithm executed by the best performance island, in gray, computed at the beginning of each cycle of evolutionary generations defined by the parameter BACI.

Furthermore, since the best BA is not known during the initial BA classification interval, reconfigurations only start after it. Once the first phase of the classification of BAs is performed, stagnated islands will continuously update their current BA to the best BA. Island stagnation is understood as not improving the best individual fitness in the last three generations.

The new stagnation-based reconfigurable HePIMs are denoted according to their topology as $\mathcal{P}^{\text{recHetStag}}_{\text{Tr12A}}$ and $\mathcal{P}^{\text{recHetStag}}_{\text{gbmm12A}}$.

## 5. Experiments and analysis of accuracy

As in [7], all PIMs, including the new reconfigurable models, were implemented using the `MPI` library of `C` in Linux, and for the sake of comparison, experiments were executed on a computational platform using two Xeon E5-2620 2.4 GHz six-core processors with hyper-threading.

The basis for comparing the performance of PIMs, are sequential versions of `GA`, `GAD`, and `DE` with populations of size $24n \log n$, for inputs of length $n$,

and 200 generations. For a fair comparison, all PIMs deal with populations of the same size. Also, we select three static binary three and three dynamic complete graph 12-island asynchronous HoPIMs, designed in [4], each running one of the BAs: `GA`, `GAD`, and `DE`. In addition, two asynchronous 12-island HePIMs, with the topologies of the HoPIMs above, presented in [6], were adjusted, running in their islands the BAs `GA`, `GAD`, and `DE`. Finally, two asynchronous 12-island reconfigurable HePIMs, with identical topologies as above, and applying (periodic) reconfiguration at fixed generation frequencies were adapted from [7].

The homogeneous models are $\mathcal{P}^{\texttt{GA}}_{\text{Tr12A}}$, $\mathcal{P}^{\texttt{GAD}}_{\text{Tr12A}}$, $\mathcal{P}^{\texttt{DE}}_{\text{Tr12A}}$, $\mathcal{P}^{\texttt{GA}}_{\text{gbmm12A}}$, $\mathcal{P}^{\texttt{GAD}}_{\text{gbmm12A}}$ and $\mathcal{P}^{\texttt{DE}}_{\text{gbmm12A}}$. The superscripts denote the BA used by the homogeneous model. The subscript prefixes indicate whether the model uses the static tree (`Tr`) or the dynamic complete graph topology (`gbmm`), and the subscript suffix `12A` indicates the number of islands and that the model is asynchronous. Furthermore, it is essential to point out that the homogeneous model $\mathcal{P}^{\texttt{DE}}_{\text{gbmm12A}}$ provides the best solutions for the URD problem.

The heterogeneous models are $\mathcal{P}^{\text{Het}}_{\text{Tr12A}}$ and $\mathcal{P}^{\text{Het}}_{\text{gbmm12A}}$, and the reconfigurable HePIMs with fixed (periodic) reconfiguration frequency are $\mathcal{P}^{\text{recHet}}_{\text{Tr12A}}$ and $\mathcal{P}^{\text{recHet}}_{\text{gbmm12A}}$. The new HePIMs with stagnation-based reconfiguration are $\mathcal{P}^{\text{recHetStag}}_{\text{Tr12A}}$ and $\mathcal{P}^{\text{recHetStag}}_{\text{gbmm12A}}$. All PIMs have the following configuration:

- Each island has $2n \log n$ individuals, for inputs of length $n$;
- Initially, islands 1, 4, 8 and 12 run `GA`, islands 2, 6, 7 and 11 `GAD`, and islands 3, 5, 9 and 10 runs `DE`;
- Generation number is fixed at 200.

### *5.1. Parameter Setup*

We use the parameters for BAs, HoPIMs, and HePIMs obtained in [6] and [7]. The *parameter tuning* adopted the "taxonomy T1" in [27]. Table 1 presents the parameter ranges. For percentages, the tested values range between 2% and 100%. For probabilities, the values range from 0.02 to 1.0, and for the mutation parameter, from 0.01 to 0.02. For `DE`, the $F_M$ parameter is set in the range of 1% to 2%. The upper bound of 2% was defined based on the analysis of our parameter adjustment process, in which solutions with $F_M$ greater than 2% substantially degrade the quality of the solutions.

Table 2 presents the parameter configuration for sequential versions and HoPIMs, while Table 3 shows the parameters for HePIMs.

Table 1: Parameter Value Ranges

| | Parameter | Parameter 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* | $2\%, 4\%, \cdots, 98\%, 100\%$ |

Table 2: Parameter Settings for `GA`, `GAD`, `DE` and associated HoPIMs.

| Parameter | `GA` | $\mathcal{P}^{\texttt{GA}}_{\text{Tr12A}}$ | $\mathcal{P}^{\texttt{GA}}_{\text{gbmm12A}}$ | `GAD` | $\mathcal{P}^{\texttt{GAD}}_{\text{Tr12A}}$ | $\mathcal{P}^{\texttt{GAD}}_{\text{gbmm12A}}$ | `DE` | $\mathcal{P}^{\texttt{DE}}_{\text{Tr12A}}$ | $\mathcal{P}^{\texttt{DE}}_{\text{gbmm12A}}$ |
|---|---|---|---|---|---|---|---|---|---|
| *crossover* | 0.90 | 0.98 | 0.96 | 0.92 | 0.98 | 0.98 | | | |
| *mutation* | 0.02 | 0.015 | 0.011 | 0.01 | 0.01 | 0.01 | | | |
| *selection* | 60% | 92% | 94% | 98% | 98% | 94% | | | |
| *replacement* | 60% | 70% | 70% | 90% | 80% | 90% | | | |
| $P_C$ | | | | | | | 0.74 | 0.72 | 0.78 |
| $F_M$ | | | | | | | 1% | 1.4% | 1% |
| *IN* | | 9 | 5 | | 12 | 5 | | 3 | 5 |
| *EMI* | | 1 | 1 | | 1 | 1 | | 1 | 1 |
| *EP* | | 2 | 2 | | 2 | 1 | | 1 | 2 |
| *IMI* | | 1 | 1 | | 1 | 1 | | 1 | 1 |
| *MI* | | 30% | 30% | | 14% | 12% | | 14% | 12% |

Table 3: Parameter Settings for HePIMs.

| Parameter | $\mathcal{P}^{\text{Het}}_{\text{Tr12A}}$ | $\mathcal{P}^{\text{Het}}_{\text{gbmm12A}}$ | $\mathcal{P}^{\text{recHet}}_{\text{Tr12A}}$ | $\mathcal{P}^{\text{recHet}}_{\text{gbmm12A}}$ | $\mathcal{P}^{\text{recHetStag}}_{\text{Tr12A}}$ | $\mathcal{P}^{\text{recHetStag}}_{\text{gbmm12A}}$ |
|---|---|---|---|---|---|---|
| *IN* | 3 | 6 | 3 | 6 | 3 | 6 |
| *EMI* | 1 | 3 | 1 | 3 | 1 | 3 |
| *EP* | 2 | 2 | 2 | 1 | 2 | 1 |
| *IMI* | 3 | 3 | 3 | 3 | 3 | 3 |
| *MI* | 10% | 10% | 10% | 14% | 10% | 14% |
| *RF* | | | 14% | 24% | | |
| *BACI* | | | | | 10% | 10% |

### *5.2. Analysis of Accuracy*

The experiments were conducted as described below, considering preliminary results on reconfigurable models obtained in [7].

- The evolutionary process was extended to 200 generations for all samples to guarantee the desired effect from the reconfigurable approach. In previous works, the number of evolutionary generations was fixed as the length of the permutation inputs reaching the maximum number in samples of length 150 evolved during 150 generations (cf [6, 7]).

- The PSO algorithm, used by HePIMs in previous work, is no longer used as an evolutionary engine because it always performs much worse than algorithms: `GA`, `GAD` and `DE`. When considering heterogeneous versions, the `PSO` maintained a slice of three islands, and due to its inferior performance, it negatively impacted the solutions of heterogeneous PIMs (see [7]).

- We diversified the length of the input instances:

  - For each permutation length, $n \in \{100, 110, \ldots, 150\}$, one package of one hundred unsigned permutations with $n$ genes was randomly generated;
  - All PIMs were executed ten times (a total of one thousand executions) using each one of the permutations of length $n$ and the average of these executions for each permutation was taken as the result. The average gives the computed number of reversals for each unsigned permutation.

The radar chart in Figure 5 compares the sequential algorithms `GA`, `GAD`, and `DE`. Since URD is a minimization problem, the smaller the output, the higher the accuracy. The radar chart shows that `DE` presents the best and `GA` the worst solutions. The algorithm `GAD` only computes better solutions than `DE` for inputs of length 130. The genetic algorithm variants `GA` and `GAD` behave differently as the search space increases: for permutations of length 100 and 110, `GA` provides the best solutions, but the scenario is reversed for longer inputs.

The radar charts in Figure 6 and the left side radar chart in Figure 7 show results from the HoPIMs. They show how sequential versions are easily overcome by HoPIMs, and that the accuracy of the outputs computed by dynamic HoPIMs is better than the outputs computed by static versions. The three radar charts have different scales, and for comparison, the right side radar chart in Figure 7 presents all static and dynamic homogeneous PIMs. The best-adapted model is the dynamic HoPIM with BA `DE`, $\mathcal{P}^{\texttt{DE}}_{\text{gbmm12A}}$. The strength of dynamic models has already been presented in previous work. The property of not having a fixed neighborhood allows faster dissemination of genetic material with high evolutionary potential among the islands. A

less dominant scenario is obtained with experiments involving HoPIM from the GA as seen in the left side radar chart in Figure 6, where only for inputs of size $100, 110$, it is possible to see a substantial improvement of the dynamic HoPIM.

The left and right side radar charts in Figure 8 compares the results of the static and dynamic HePIMs versus the best static and dynamic HoPIMs, $\mathcal{P}^{\texttt{DE}}_{\text{Tr12A}}$ and $\mathcal{P}^{\texttt{DE}}_{\text{gbmm12A}}$, respectively. Regarding static models (see left side chart), the HoPIM presents the worst solutions. The static neighborhood scenario imposed by the binary tree topology benefits HePIMs because islands always send and receive individuals with different evolutionary characteristics, which favors not being trapped in local optima. The static HePIMs show very similar behavior, with the proposed stagnation-based reconfigurable model $\mathcal{P}^{\text{recHetStag}}_{\text{Tr12A}}$ not performing better than the competing periodic reconfigurable model $\mathcal{P}^{\text{recHet}}_{\text{Tr12A}}$, and the heterogeneous model, $\mathcal{P}^{\text{Het}}_{\text{Tr12A}}$. Considering longer instances of the URD problem, the periodic reconfigurable model, $\mathcal{P}^{\text{recHet}}_{\text{Tr12A}}$, is the best static HePIM. On the other hand, the stagnation-based reconfiguration method succeeded for the dynamic topology (see the right side chart); indeed, $\mathcal{P}^{\text{recHetStag}}_{\text{gbmm12A}}$ presented the best quality solutions, and the distance to the second best model ($\mathcal{P}^{\texttt{DE}}_{\text{gbmm12A}}$) becomes even more significant when considering large instances of the URD problem.

The radar chart in Figure 9 compiles all results in both radar charts in Figure 8 using the same scale. In this radar chart, it is clear how the migration policy on the dynamic topology of the complete graph model jointly with stagnation-based reconfiguration approach provided by the model $\mathcal{P}^{\text{recHetStag}}_{\text{gbmm12A}}$ give the required diversity to outperform all other heterogeneous and reconfigurable heterogeneous, and the best adapted homogeneous models.

A point that will arouse the reader's curiosity about the reconfiguration method is how the islands end up when all generations are executed. After the evolutionary cycle, both $\mathcal{P}^{\text{recHetStag}}_{\text{Tr12A}}$ and $\mathcal{P}^{\text{recHetStag}}_{\text{gbmm12A}}$ have all the islands running the same BA, and both become homogeneous. The average number of generations required for the models to become homogeneous is shown in Figure 10. The $\mathcal{P}^{\text{recHetStag}}_{\text{Tr12A}}$ needs more generations until it becomes homogeneous compared with the $\mathcal{P}^{\text{recHetStag}}_{\text{gbmm12A}}$, except for the sample size 120. Also, regardless of the model, the reconfiguration process is complete before reaching 100 generations.

Another question is which BA is dominant? Investigating the input samples for both stagnation-based reconfigurable models, $\mathcal{P}^{\text{recHetStag}}_{\text{Tr12A}}$ and $\mathcal{P}^{\text{recHetStag}}_{\text{gbmm12A}}$ we have the scenario presented in Table 4. Only DE and GA are being shown because there were no samples where the islands ended running the algorithm GAD. The values in the table are the percentage of inputs ending with all islands executing either the algorithm DE or GA. The algorithm DE is dom-

inant for all samples. In addition, the fact that the algorithm GAD never has dominance is unusual since, as shown in Figure 5, it delivers better quality solutions than the algorithm GA. We investigated it and realized that the HoPIMs from the GA present better solutions than GAD if we consider a maximum of 150 generations (see Figure 11). Furthermore, as seen in Table 4, before 100 generations, all islands have already become homogeneous. We can attribute the success of model $\mathcal{P}^{\text{recHetStag}}_{\text{gbmm12A}}$ to the aptitude of the dynamic topology and the fact that it becomes homogeneous earlier. Hence, the evolutionary engine DE has more time to evolve a population that already has individuals shaken by GA and GAD, which makes it difficult to stay stuck in great locations.

Finally, partial accuracy results of $\mathcal{P}^{\texttt{DE}}_{\text{gbmm12A}}$ and $\mathcal{P}^{\text{recHetStag}}_{\text{gbmm12A}}$ during the evolutionary cycle are compared in Figure 12. We use the population average to measure island development at intervals of 20 generations. The dynamic stagnation-based reconfigurable model $\mathcal{P}^{\text{recHetStag}}_{\text{gbmm12A}}$ produces populations better adapted to the URD problem regardless of the analyzed interval. Although the partial accuracy evolution between both models is always very close, it is noticeable that the advantage caused by heterogeneity and algorithmic stagnation-based reconfiguration (regarding the best adapted homogeneous model $\mathcal{P}^{\texttt{DE}}_{\text{gbmm12A}}$) at the beginning of the evolutionary cycle is maintained after the model becomes homogeneous.

Table 4: Configuration of islands of the reconfiguralbe HePIMs $\mathcal{P}^{\text{recHetStag}}_{\text{Tr12A}}$ and $\mathcal{P}^{\text{recHetStag}}_{\text{gbmm12A}}$ at the end of the evolutionary cycle (200 generations).

| Length | $\mathcal{P}^{\text{recHetStag}}_{\text{Tr12A}}$ | | $\mathcal{P}^{\text{recHetStag}}_{\text{gbmm12A}}$ | |
|---|---|---|---|---|
| | DE | GA | DE | GA |
| 100 | 90% | 10% | 100% | |
| 110 | 100% | | 80% | 20% |
| 120 | 100% | | 100% | |
| 130 | 90% | 10% | 90% | 10% |
| 140 | 100% | | 70% | 30% |
| 150 | 90% | 10% | 90% | 10% |

### *5.3. Performance*

The speed-up of the homogeneous models $\mathcal{P}^{\texttt{DE}}_{\text{Tr12A}}$ and $\mathcal{P}^{\texttt{DE}}_{\text{gbmm12A}}$ was evaluated regarding the runtime of the sequential algorithm DE. The input dataset with permutations of length 150 was chosen because such permutations represent more challenging, inherently complex problems than shorter permutations. The algorithms $\mathcal{P}^{\texttt{DE}}_{\text{gbmm12A}}$, $\mathcal{P}^{\texttt{DE}}_{\text{Tr12A}}$, and DE were executed ten times for

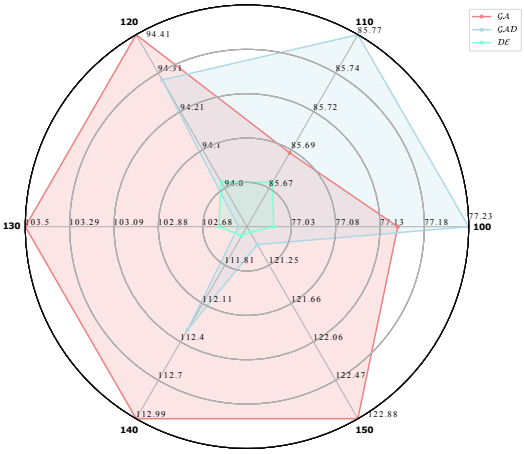


Figure 5: Accuracy of the sequential BAs: `GA`, `GAD` and `DE`. The radar chart is scaled according to the worst performance for each input size. Since the target optimization problem URD is a minimization problem, the smaller the radius the better the result.

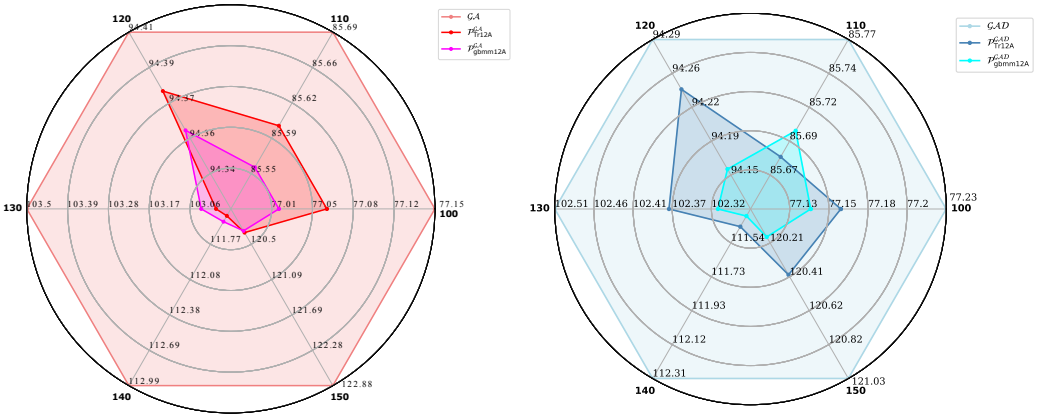


Figure 6: Left side radar chart: accuracy of the HoPIMs from `GA`. Right side radar chart: Accuracy of the HoPIMs from `GAD`. Each radar chart is scaled according to the performance of the associated sequential algorithm since it provides the worst performance.

each permutation in the dataset, and the meantime is the average runtime. The dynamic homogeneous model $\mathcal{P}^{\texttt{DE}}_{\text{gbmm12A}}$ reached a speed-up of 7.18 while the static homogeneous model $\mathcal{P}^{\texttt{DE}}_{\text{Tr12A}}$ a speed-up of 8.37.

Due to the nature of HePIMs, where groups of islands run different BAs, it is not easy to decide which sequential BA would be the reference to compute the speed-up; thus, the Table 5 shows the speed-up of heterogeneous models concerning the BAs `GA`, `GAD`, and `DE`. The methodology to compute speed-ups for HePIMs is the same applied to HoPIMs. The simple versions of HePIMs provide the best speed-ups. Static HePIMs deliver bet-

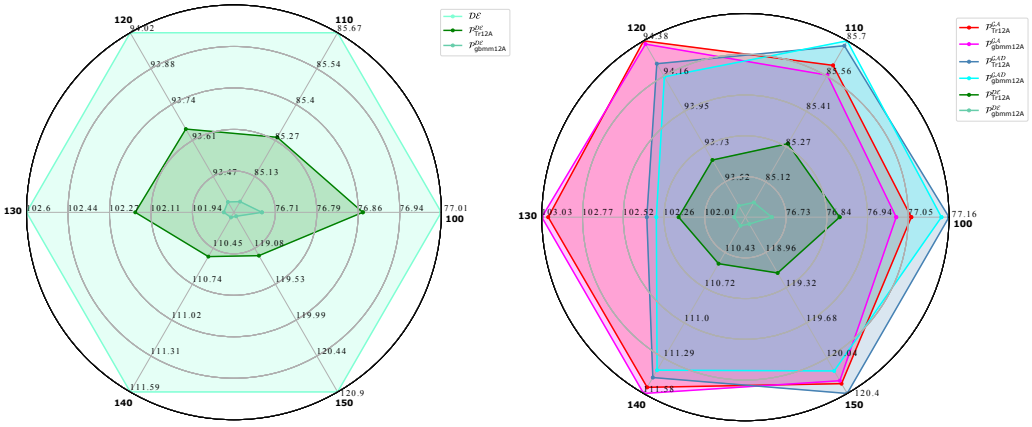

Figure 7: Left side radar chart: accuracy of the HoPIMs from DE. Right side radar chart: Comparison of the accuracy of all HoPIMs. The first radar chart is scaled according to the accuracy of the sequential model, while the second radar chart is scaled according to the model with the worst accuracy for each input length.

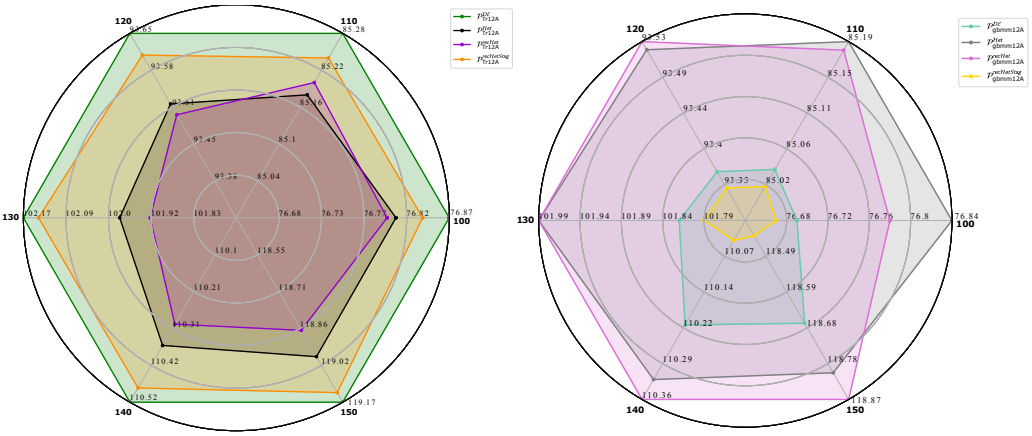

Figure 8: Comparing accuracy of (left side radar chart) static models: HoPIM from DE and HePIM, and reconfigurable HePIMs, and (right side radar chart) dynamic models: HoPIM from DE and HePIM, and reconfigurable HePIMs. The charts are scaled according to the model with the worst performance. For the static tree topology model, the homogeneous model, $\mathcal{P}^{\mathtt{DE}}_{\mathrm{Tr12A}}$ has the worst accuracy; for the dynamic complete graph topology model, the stagnation-based reconfigurable HePIM presented the bests accuracy.

ter performances than dynamic HePIMs since dynamic topology PIMs build neighborhoods for the islands at each migratory cycle. The proposed reconfiguration process involves sophisticated techniques that improve the quality of the solutions; however, it impacts performance. The stagnation-based reconfigurable HePIMs $\mathcal{P}^{\mathrm{recHetStag}}_{\mathrm{Tr12A}}$ and $\mathcal{P}^{\mathrm{recHetStag}}_{\mathrm{gbmm12A}}$ are slower than the periodic reconfigurable models $\mathcal{P}^{\mathrm{recHet}}_{\mathrm{Tr12A}}$ and $\mathcal{P}^{\mathrm{recHet}}_{\mathrm{gbmm12A}}$ since they make use of an effortless

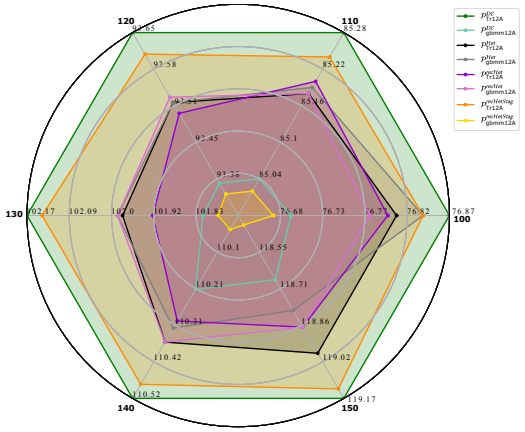


Figure 9: Radar chart compiling all accuracies in Figure 8. The chart is scaled according to the accuracy of the static homogeneous model $\mathcal{P}^{\mathtt{DE}}_{\mathrm{Tr12A}}$ providing the worst performance. The best accuracy is obtained in all cases by the dynamic stagnation-based reconfigurable heterogeneous model, $\mathcal{P}^{\mathrm{recHetStag}}_{\mathrm{gbmm12A}}$.

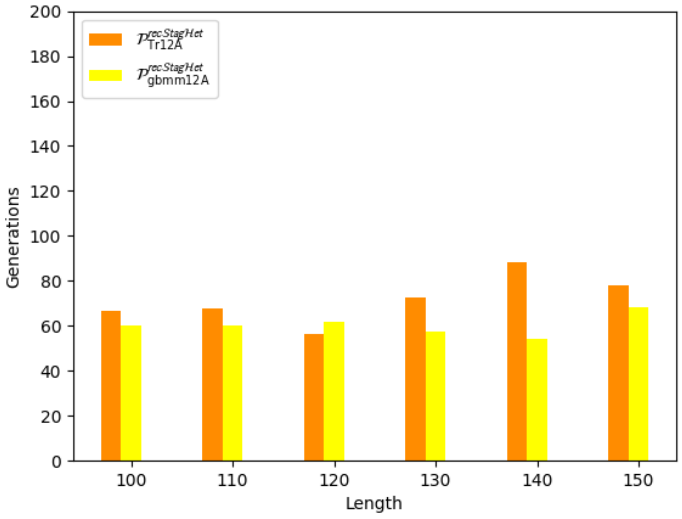


Figure 10: The diagram shows the number of generations required for the reconfigurable HePIMs $\mathcal{P}^{\mathrm{recHetStag}}_{\mathrm{Tr12A}}$ and $\mathcal{P}^{\mathrm{recHetStag}}_{\mathrm{gbmm12A}}$ to become HoPIMs; i.e., to homogeneously run the same BA in all their islands.

reconfiguration process.

### *5.4. Statistical Analysis*

Statistical tests validated experiments with a significance level $\alpha = 0.05$, i.e., 5% chance of incorrectly rejecting the null hypothesis in the long term.

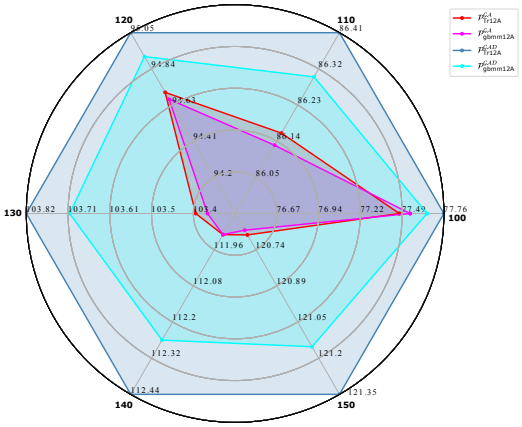


Figure 11: The radar chart shows the accuracy obtained by static and dynamic HoPIMs from `GA` and `GAD` with an evolutionary history of 150 generations. Compare with the radar chart on the right of Figure 7 showing that in experiments with 200 generations not necessarily the `GA` based homogeneous models provide the best performance. Also, observe in Table 4 that no reconfigurable model finish running `GAD` in all its islands.

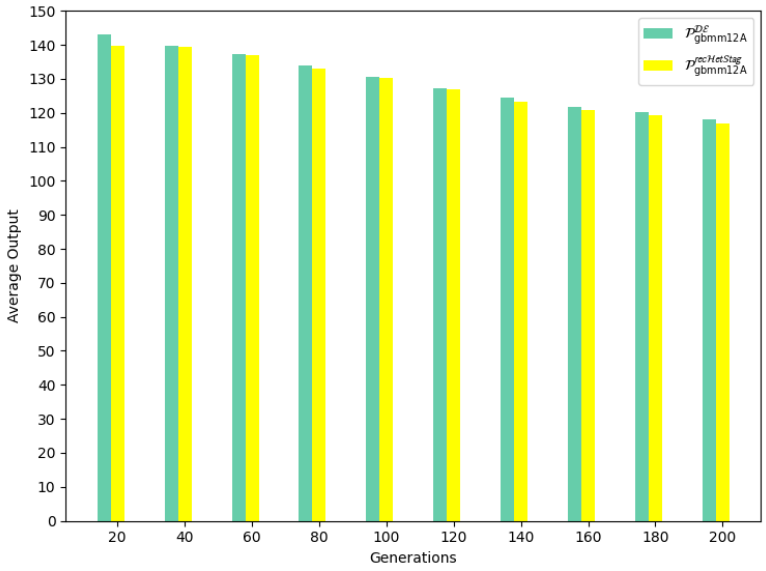


Figure 12: Comparing the partial evolution during 200 generations of the best models: HoPIM $\mathcal{P}^{\mathtt{DE}}_{\mathrm{gbmm12A}}$ and reconfigurable HePIM $\mathcal{P}^{\mathrm{recHetStag}}_{\mathrm{gbmm12A}}$.

The samples are the sets of one hundred outputs considered in Section 5.2. The first step is applying the Friedman non-parametric statistical test to define the control algorithm. Then, the multiple hypothesis testing Holm's method was used to check the null hypothesis that the performance of the control algorithm is the same concerning the remaining algorithms. The

Table 5: Speed-up for HePIMs regarding the sequential version of `GA`, `GAD`, and `DE` for the dataset with genomes of length 150

| | $\mathcal{P}^{\text{Het}}_{\text{Tr12A}}$ | $\mathcal{P}^{\text{recHet}}_{\text{Tr12A}}$ | $\mathcal{P}^{\text{recHetStag}}_{\text{Tr12A}}$ | $\mathcal{P}^{\text{Het}}_{\text{gbmm12A}}$ | $\mathcal{P}^{\text{recHet}}_{\text{gbmm12A}}$ | $\mathcal{P}^{\text{recHetStag}}_{\text{gbmm12A}}$ |
|---|---|---|---|---|---|---|
| `GA` | 6.69 | 6.40 | 5.33 | 6.42 | 5.86 | 5.14 |
| `GAD` | 6.40 | 6.12 | 5.09 | 6.14 | 5.61 | 4.92 |
| `DE` | 7.68 | 7.35 | 6.12 | 7.37 | 6.73 | 5.9 |

application of Friedman's test and Holm's method was motivated by the discussion [28].

Table 6 presents the statistical results for the HePIMs and the best HoPIM ($\mathcal{P}^{\texttt{DE}}_{\text{gbmm12A}}$). The Friedman test selects $\mathcal{P}^{\text{recHetStag}}_{\text{gbmm12A}}$ as the control algorithm. Holm's method rejects the null hypotheses (for $p$-value $\leq 0.05$). Hence, the model $\mathcal{P}^{\text{recHetStag}}_{\text{gbmm12A}}$ has statistical significance for all other HePIMs regardless of the input. On the other hand, when compared with the algorithm $\mathcal{P}^{\texttt{DE}}_{\text{gbmm12A}}$, the results showed significance only for inputs with a length greater than 130. In Table 6, an algorithm has statistical significance, whenever $p$-value $\leq \alpha/i$.

## 6. Conclusions and future work

This work consolidates the reconfigurable heterogeneous Parallel Island Models, introduced in [7], presenting the innovative approach of stagnation-based reconfiguration. Through stagnation-based reconfiguration, islands may dynamically update their local bioinspired algorithm at each generation of the evolutionary process. In contrast to the periodic reconfiguration mechanism, proposed initially in [7], where island reconfiguration was parsimoniously performed accordingly to a fixed generational frequency, and only updating the algorithm being executed in the island with the worst by the algorithm applied by the island with best performance, this exhaustive stagnation-based reconfiguration mechanism provides a reconfigurable dynamic heterogeneous PIM, $\mathcal{P}^{\text{recHetStag}}_{\text{gbmm12A}}$, that outperforms all other parallel island models, even the best-adapted homogeneous PIMs (that use `DE` in all its islands) and other heterogeneous (reconfigurable) PIMs.

Experiments evaluated statistically show that the proposed reconfiguration technique is practical and provides competitive solutions with the reference models for the unsigned reversal distance $\mathcal{NP}$-hard optimization problem.

This work offers attractive further investigation opportunities. On the one side, from a practical perspective, it is natural to infer that, after enough evolutionary generations, the reconfiguration feature on HePIMs provides a natural mechanism to converge to the best-adapted homogeneous PIM to

any optimization problem. Nevertheless, it is relevant to investigate how dynamic reconfigurable heterogeneity provides advantages at the beginning of evolution that the model can preserve during all generations of the evolutionary process (without the necessity to perform the best-adapted bioinspired algorithm). Hence, the interest in performing experiments with a greater diversity of recent bioinspired algorithms (e.g., elephant herding behavior [29], social spider algorithm [30]). Regarding other target optimization problems, it is required to investigate how the flexibility of the new reconfigurable HePIMs adapts to solve different optimization problems. In particular, algorithmic heterogeneity appears adequate to address multi-objective optimization problems (e.g., [31, 24, 32]). Indeed, heterogeneity and the flexibility of reconfiguration will facilitate the application of the best-adapted BA to each objective optimization problem being addressed by each island.

## References


[1] 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.

[2] L. A. da Silveira, J. L. Soncco-Álvarez, and M. Ayala-Rincón, "Parallel genetic algorithms with sharing of individuals for sorting unsigned genomes by reversals," in *IEEE Cong. on Evol. Comp. (CEC)*, pp. 741–748, 2017.

[3] 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.

[4] 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.

[5] E. Alba, A. J. Nebro, and J. M. Troya, "Heterogeneous computing and parallel genetic algorithms," *Journal of Parallel and Distributed Computing*, vol. 62, no. 9, pp. 1362–1385, 2002.

[6] L. A. da Silveira, J. L. Soncco-Álvarez, T. A. de Lima, and M. Ayala-Rincón, "Heterogeneous Parallel Island Models," in *IEEE Symposium Series on Computational Intelligence (SSCI)*, 2021.

[7] L. A. Da Silveira, T. A. De Lima, and M. Ayala-Rincón, "Reconfigurable heterogeneous parallel island models," in *2022 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1618–1625, 2022.

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

[9] K. A. D. Jong and W. M. Spears, "A formal analysis of the role of multi-point crossover in genetic algorithms," *Annals of Mathematics and Artificial Intelligence*, vol. 5, p. 1–26, 1992.

[10] 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.

[11] 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.

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

[13] J. L. Soncco-Álvarez, D. M. Muñoz, and M. Ayala-Rincón, "Opposition-based memetic algorithm and hybrid approach for sorting permutations by reversals," *Evol. Comput.*, vol. 27, no. 2, pp. 229–265, 2019.

[14] 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.

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

[17] 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.

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

[19] G. Duarte, A. Lemonge, and L. Goliatt, "A dynamic migration policy to the island model," in *IEEE Cong. on Evol. Comp. (CEC)*, pp. 1135–1142, 2017.

[20] 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.

[21] G. R. Duarte and B. S. L. P. d. Lima, "Differential evolution variants combined in a hybrid dynamic island model," in *2020 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–8, 2020.

[22] 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.

[23] Q. Meng, J. Wu, J. Ellis, and P. J. Kennedy, "Dynamic island model based on spectral clustering in genetic algorithm," in *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 1724–1731, 2017.

[24] 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.

[25] F. Lardeux, J. Maturana, E. Rodriguez-Tello, and F. Saubion, "Migration policies in dynamic island models," *Natural Computing: An International Journal*, vol. 18, p. 163–179, mar 2019.

[26] M. Nssibi, G. Manita, and O. Korbaa, "Advances in nature-inspired metaheuristic optimization for feature selection problem: A comprehensive survey," *Computer Science Review*, vol. 49, no. 100559, 2023.

[27] 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.

[28] D. G. Pereira, A. Afonso, and F. M. Medeiros, "Overview of friedman's test and post-hoc analysis," *Communications in Statistics - Simulation and Computation*, vol. 44, no. 10, pp. 2636–2653, 2015.

[29] G. Wang, S. Deb, X. Gao, and L. D. S. Coelho, "A new metaheuristic optimisation algorithm motivated by elephant herding behaviour," *International Journal of Bio-Inspired Computation*, vol. 8, no. 6, pp. 394–409, 2017.

[30] J. James and V. O. Li, "A social spider algorithm for global optimization," *Applied Soft Computing*, vol. 30, pp. 614–627, 2015.

[31] 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.

[32] 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.

Table 6: Holm test for $\mathcal{P}^{\texttt{DE}}_{\text{gbmm12A}}$ x HePIMs.

| **Length** | **Control** | $i$ | **Algorithm** | $p$**-value** | $\alpha/i$ |
|---|---|---|---|---|---|
| 100 | $\mathcal{P}^{\text{recHetStag}}_{\text{gbmm12A}}$ | 6 | $\mathcal{P}^{\text{recHetStag}}_{\text{Tr12A}}$ | 1.2258954459292176E-13 | 0.008 |
| | | 5 | $\mathcal{P}^{\text{Het}}_{\text{gbmm12A}}$ | 1.7734111951013715E-13 | 0.010 |
| | | 4 | $\mathcal{P}^{\text{Het}}_{\text{Tr12A}}$ | 1.980967305825471E-8 | 0.0125 |
| | | 3 | $\mathcal{P}^{\text{recHet}}_{\text{Tr12A}}$ | 2.761583790983588E-7 | 0.016 |
| | | 2 | $\mathcal{P}^{\text{recHet}}_{\text{gbmm12A}}$ | 8.381464733500173E-7 | 0.025 |
| | | 1 | $\mathcal{P}^{\texttt{DE}}_{\text{gbmm12A}}$ | 0.7188026245529647 | 0.05 |
| 110 | $\mathcal{P}^{\text{recHetStag}}_{\text{gbmm12A}}$ | 6 | $\mathcal{P}^{\text{recHetStag}}_{\text{Tr12A}}$ | 3.6506733156072896E-20 | 0.008 |
| | | 5 | $\mathcal{P}^{\text{recHet}}_{\text{Tr12A}}$ | 9.571441858241067E-14 | 0.01 |
| | | 4 | $\mathcal{P}^{\text{Het}}_{\text{gbmm12A}}$ | 9.62783195797665E-13 | 0.012 |
| | | 3 | $\mathcal{P}^{\text{recHet}}_{\text{gbmm12A}}$ | 9.107359384790177E-11 | 0.016 |
| | | 2 | $\mathcal{P}^{\text{Het}}_{\text{Tr12A}}$ | 4.996646058399122E-10 | 0.025 |
| | | 1 | $\mathcal{P}^{\texttt{DE}}_{\text{gbmm12A}}$ | 0.4515390224098334 | 0.05 |
| 120 | $\mathcal{P}^{\text{recHetStag}}_{\text{gbmm12A}}$ | 6 | $\mathcal{P}^{\text{recHetStag}}_{\text{Tr12A}}$ | 1.927919232582314E-21 | 0.008 |
| | | 5 | $\mathcal{P}^{\text{recHet}}_{\text{gbmm12A}}$ | 2.200327289088137E-12 | 0.01 |
| | | 4 | $\mathcal{P}^{\text{Het}}_{\text{gbmm12A}}$ | 8.169951697851222E-11 | 0.012 |
| | | 3 | $\mathcal{P}^{\text{Het}}_{\text{Tr12A}}$ | 1.933815449552944E-10 | 0.016 |
| | | 2 | $\mathcal{P}^{\text{recHet}}_{\text{Tr12A}}$ | 3.4787444807522945E-8 | 0.025 |
| | | 1 | $\mathcal{P}^{\texttt{DE}}_{\text{gbmm12A}}$ | 0.47145170683892584 | 0.05 |
| 130 | $\mathcal{P}^{\text{recHetStag}}_{\text{gbmm12A}}$ | 6 | $\mathcal{P}^{\text{recHetStag}}_{\text{Tr12A}}$ | 1.6510072254753912E-27 | 0.008 |
| | | 5 | $\mathcal{P}^{\text{Het}}_{\text{gbmm12A}}$ | 4.422946724480415E-12 | 0.01 |
| | | 4 | $\mathcal{P}^{\text{Het}}_{\text{Tr12A}}$ | 1.2595868719961186E-10 | 0.012 |
| | | 3 | $\mathcal{P}^{\text{recHet}}_{\text{gbmm12A}}$ | 2.9565205977131737E-10 | 0.016 |
| | | 2 | $\mathcal{P}^{\text{recHet}}_{\text{Tr12A}}$ | 6.824213461036749E-10 | 0.025 |
| | | 1 | $\mathcal{P}^{\texttt{DE}}_{\text{gbmm12A}}$ | 0.5891354643621368 | 0.05 |
| 140 | $\mathcal{P}^{\text{recHetStag}}_{\text{gbmm12A}}$ | 6 | $\mathcal{P}^{\text{recHetStag}}_{\text{Tr12A}}$ | 1.027115868211875E-31 | 0.008 |
| | | 5 | $\mathcal{P}^{\text{Het}}_{\text{Tr12A}}$ | 5.757742850114732E-20 | 0.01 |
| | | 4 | $\mathcal{P}^{\text{recHet}}_{\text{gbmm12A}}$ | 1.5126612148387156E-18 | 0.012 |
| | | 3 | $\mathcal{P}^{\text{recHet}}_{\text{Tr12A}}$ | 1.749612437192936E-18 | 0.016 |
| | | 2 | $\mathcal{P}^{\text{Het}}_{\text{gbmm12A}}$ | 2.413141855941992E-16 | 0.025 |
| | | 1 | $\mathcal{P}^{\texttt{DE}}_{\text{gbmm12A}}$ | 8.925709019458667E-4 | 0.05 |
| 150 | $\mathcal{P}^{\text{recHetStag}}_{\text{gbmm12A}}$ | 6 | $\mathcal{P}^{\text{recHetStag}}_{\text{Tr12A}}$ | 4.288491503871603E-45 | 0.008 |
| | | 5 | $\mathcal{P}^{\text{Het}}_{\text{Tr12A}}$ | 3.257446176124881E-31 | 0.010 |
| | | 4 | $\mathcal{P}^{\text{recHet}}_{\text{Tr12A}}$ | 9.286667219420533E-24 | 0.012 |
| | | 3 | $\mathcal{P}^{\text{recHet}}_{\text{gbmm12A}}$ | 4.629733664111373E-22 | 0.016 |
| | | 2 | $\mathcal{P}^{\text{Het}}_{\text{gbmm12A}}$ | 8.052342507094246E-17 | 0.025 |
| | | 1 | $\mathcal{P}^{\texttt{DE}}_{\text{gbmm12A}}$ | 2.4159005413133192E-5 | 0.05 |