IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received February 9, 2022, accepted February 21, 2022, date of publication March 4, 2022, date of current version March 16, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3156944

Addressing Evolutionary-Based Dynamic Problems: A New Methodology for Evaluating Immigrants Strategies in MOGAs

ANGEL PANIZO-LLEDOT[1], MARTÍN PEDEMONTE[2], (Member, IEEE), GEMA BELLO-ORGAZ[1], AND DAVID CAMACHO[1], (Senior Member, IEEE)

[1]Computer Systems Engineering Department, Technical University of Madrid, 28040 Madrid, Spain
[2]Instituto de Computación, Universidad de la República, Montevideo 11200, Uruguay

Corresponding author: Angel Panizo-Lledot (angel.panizo@upm.es)

This work was supported in part by the XAI-DisInfodemics under Grant PLEC2021-007681, in part by the FightDIS under Grant PID2020-117263GB-100, in part by Grant MCIN/AEI/10.13039/501100011033, in part by ERDF (a way of making Europe), in part by the European Union, in part by the European Union NextGenerationEU/PRTR, in part by the Research Project CIVIC: Intelligent Characterization of the Veracity of the Information Related to COVID-19, in part by BBVA Foundation Grants for Scientific Research Teams SARS-CoV-2 and COVID-19, and in part by the European Commission through the IBERIFIER—Iberian Digital Media Research and Fact-Checking Hub under Grant 2020-EU-IA-0252. The work of Martín Pedemonte was supported in part by ANII under Grant MOV_CA_2019_1_156657, and in part by CSIC, UDELAR, at the Universidad Politécnica de Madrid. The work of David Camacho was supported by the Comunidad Autónoma de Madrid through Convenio Plurianual with the Universidad Politécnica de Madrid in the actuation line of the Programa de Excelencia para el Profesorado Universitario.

ABSTRACT Multi-Objective Genetic Algorithms (MOGAs) have been successfully used to address dynamic problems in a wide variety of domains. In these domains, data changes over time, so a non-static analysis is required to obtain feasible solutions. In this type of environments, MOGAs are often time-consuming and require special adaptation to work properly. A number of different techniques have been proposed to adapt MOGAs to dynamic environments for tackling the previous problems such as hypermutation, memory and immigrant schemes or multi-population methods, among others. In particular, immigrant strategies are one of the most commonly used methods, for that reason, this work proposes a new methodology that allows to make a detailed evaluation of their performance when these strategies are used. The proposed methodology works on two levels, a coarse-grain one and a fine-grain one. In the former, an overall evaluation of the different immigrant strategies is made based on three different dimensions: *Quality*, *Stability* and *Speed*. In the latter, a detailed study of the status of the immigrant individuals during the evolution of the algorithm is carried out. This is a very relevant aspect to take into account in order to evaluate whether an immigrant strategy is working properly or not. To deploy this methodology, a new visualization technique for population mixing analysis is proposed in this work. In order to validate the proposed methodology, a test case in the context of the Dynamic Community Detection problem (DCD) has been selected using a MOGA that applies several different immigrant schemes, showing both how the methodology works and how it could be employed in a particular dynamic problem.

INDEX TERMS Dynamic problems, immigrant strategies, multi-objective genetic algorithms, dynamic community detection, social network analysis.

I. INTRODUCTION

In areas such as computational complexity theory, dynamic problems are defined as a kind of problems where input data changes over time [1]. This simply fact critically affects how the information is represented and how the algorithms handle data. These algorithms, usually known as *dynamic algorithms*, need to manage two main issues, the first one related to the memory space required to store data, and the second one related to a set of time-based features (initialization, insertion, deletion, query, etc…) and how the algorithm uses data in different time stamps [2]. Many real-world problems are dynamic in nature, in particular within the optimization field. Dynamic Multi-objective Optimization

The associate editor coordinating the review of this manuscript and approving it for publication was Qingli Li.

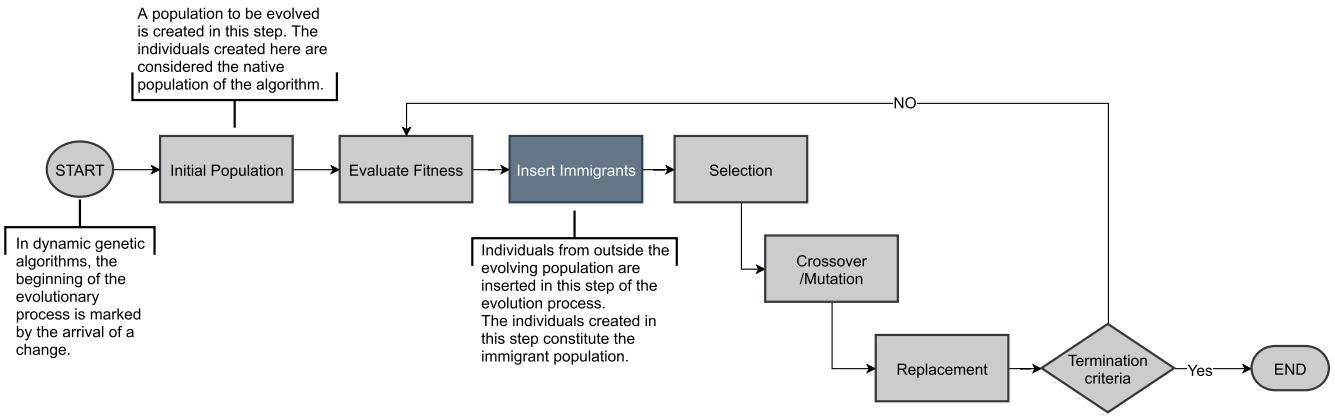


FIGURE 1. Flow chart of a genetic algorithm that integrates an immigrant strategy in its algorithmic scheme. The steps involved in the vanilla version of the algorithm are shown in light (grey) color. Contrary, the step added for the immigrants variant of the algorithm is shown in a dark one.

Problems (DMOPs) are widely present in real world [3]. This type of problem is characterized by time-varying objective functions in which at least two goals can be pursued. Traditionally, solving DMOPs implies not only evolving the solutions close to the optimal Pareto Front (PF) in any given timestamp but also doing it quickly before a change occurs. In recent years, another goal has also been proposed, which implies finding solutions robust to changes over time (ROOT) [4].

DMOP is a very active area of research in evolutionary computation [5], [6], especially in relation to Multi-Objective Genetic Algorithms (MOGAs). To address these problems, MOGAs require that the optimization algorithm can trace the Pareto-optimal over time, increasing its computational complexity. Therefore, MOGAs should be able not only to evolve a near-optimal and diverse PF, but also to continually track a time-changing environment. However, it is unreasonable to assume no priori knowledge whenever a change occurs in the environment. Hence, some extent of knowledge transfer [7] is mandatory when designing MOGAs for dynamic problems. Although more complex techniques exist, such as those proposed in the emerging field of Evolutionary Multi-Task Optimization (EMTO) [8], the reuse of individuals from previous generations is the simplest and most popular technique for sharing knowledge in genetic algorithms for dynamic problems. Researchers in this area have proposed many methods to address this issue [9]. Some of the techniques presented for this purpose are *hypermutation* [10] (increases the mutation rate of the population when a change occurs in the environment), *memory schemes* [11] (maintains a memory of individuals during all the MOGA execution, and when a change occurs, this memory is used to influence the current population), and *immigrant's* [12] (inserts new individuals into the population when a change occurs in order to improve the population's diversity). Among the different approaches mentioned, the *immigrant's* technique is one of the most widespread and popular [10], [13]–[15]. However, tuning this method for optimal performance is not a straightforward task. Setting the number of immigrants to be inserted, the replacement strategy of those immigrants, or boosting their survival probability is a challenging task [16]. These parameters critically affect the performance of the MOGA, and it is difficult to get insights about what could be malfunctioning. For this reason, this work focuses on proposing a methodology that allows an in-depth analysis of the effectiveness of the application of this method to solve dynamic problems.

Summarizing previous algorithms and methods, any immigrant technique is based on the idea of inserting new individuals from outside the current evolving population to replace a predefined part of that population. The underlying hypothesis is that inserting new individuals will help to improve the convergence speed and/or increase the population diversity (two of the most fundamental and critical features for any evolutionary algorithm). Figure 1, shows the flow chart of a generic genetic algorithm (GA) that incorporates a basic immigrant strategy. Depending on the immigrant technique selected, the *Initial Population* and *Insert Immigrants* steps in the GA will vary. There are several strategies proposed in the literature for this type of technique, some of the most commonly applied are the following:

1) **Random Immigrant Genetic Algorithm** (RIGA) [17]: Algorithm 1 contains a pseudo-code that implements the RIGA method. In this immigrants scheme, the initial population is composed by the solutions of the population that was being evolved previously to the new change (line 1). In the *Insert Immigrants* step (line 11), newly created random individuals are inserted into the population replacing the worse individuals. This technique aims to enhance the exploration capabilities of the algorithm by making the population more diverse.
2) **Elite Immigrants Genetic Algorithm** (EIGA) [18]: Algorithm 2 contains a pseudo-code that implements

Algorithm 1 RIGA

Require: P^{t-1}
1: $P^t \leftarrow updatePrevious(P^{t-1})$
2: $P^t \leftarrow evaluate(P^t)$
3: **while** !$termination\_criteria()$ **do**
4: $parents \leftarrow selection(P^t)$
5: $offspring \leftarrow \emptyset$
6: **for all** $p_1, p_2 \in parents$ **do**
7: $child \leftarrow crossover\&mutation(p_1, p_2)$
8: $offspring \leftarrow offspring \cup child$
9: **end for**
10: $P^t \leftarrow replacement(P^t, offspring)$
{N random individuals replace the worse of P^t}
11: $P^t \leftarrow insertRandom(P^t, N)$
12: **end while**
13:
14: **return** P^t

Algorithm 2 EIGA

Require: P^{t-1}
1: $P^t \leftarrow createRandom(POP\_SIZE)$
2: $P^t \leftarrow evaluate(P^t)$
3: **while** !$termination\_criteria()$ **do**
4: $parents \leftarrow selection(P^t)$
5: $offspring \leftarrow \emptyset$
6: **for all** $p_1, p_2 \in parents$ **do**
7: $child \leftarrow crossover\&mutation(p_1, p_2)$
8: $offspring \leftarrow offspring \cup child$
9: **end for**
10: $P^t \leftarrow replacement(P^t, offspring)$
{N previous individuals replace the worse of P^t}
11: $P^t \leftarrow insertPrevious(P^t, P^{t-1}, N)$
12: **end while**
13:
14: **return** P^t

the EIGA method. In this immigrants scheme, the initial population is a completely new population that is generated randomly (line 1). In addition, the population of the MOGA before the change happens is stored in an external memory to be used as immigrants. In the *Insert Immigrants* step (line 11), individuals from the external memory are inserted into the population, replacing the worse individuals. This technique aims to enhance the convergence speed of the method by reusing past information (assuming smooth changes in the environment).

3) **Hybrid Immigrants Genetic Algorithm** (HIGA) [19]: Algortihm 3 contains a pseudo-code that implements the HIGA method. This immigrants scheme combines previous techniques depending on the population needs, improving convergence speed or improving the exploration necessities by inserting a random immigrant or an elite one. In this scheme, the initial population is

Algorithm 3 HIGA

Require: P^{t-1}
1: $P^t \leftarrow createRandom(POP\_SIZE)$
2: $P^t \leftarrow evaluate(P^t)$
3: **while** !$termination\_criteria()$ **do**
4: $parents \leftarrow selection(P^t)$
5: $offspring \leftarrow \emptyset$
6: **for all** $p_1, p_2 \in parents$ **do**
7: **if** $\{p_1, p_2\} \cap nonDominate(P^t) == \emptyset$ **then**
{Exploitation is needed add from elite}
8: $child \leftarrow selectInd(P^{t-1})$
9: **else**
10: **if** $genDistance(p_1, p_2) \leq 0.1$ **then**
{Exploration is needed add a random}
11: $child \leftarrow makeRandomInd()$
12: **else**
13: $child \leftarrow crossover\&mutation(p_1, p_2)$
14: **end if**
15: **end if**
16: $offspring \leftarrow offspring \cup child$
17: **end for**
18: $P^t \leftarrow replacement(P^t, offspring)$
19: **end while**
20:
21: **return** P^t

a completely new random population (line 1) and also stores the population before the change in an external memory (as in EIGA). However, contrary to EIGA, this method combines the insertion of immigrants with the *Crossover/Mutation* phase. The method works as follows. If both parents are not members of the non-dominated set of solutions, then an individual from the external memory is inserted (lines 7-8). If at least one of the parents is a member of the non-dominated set and the distance between both parents is less than 10% of genes, then a completely random individual is inserted (lines 10-11). Otherwise, a classical crossover is used (lines 12-13).

The implementation of any particular immigrant strategy and its inclusion in a particular MOGA, in order to enable the algorithm to solve DMOPs, is often quite straightforward. However, when it comes to the empirical analysis to study how the immigrant strategy actually affects the evolution of the population and the numerical performance of the algorithm, the task becomes complex and hard. Therefore, and with the goal of making a step further in this direction, in this work we propose a new methodology that allows studying in detail the efficiency of the application of the immigrant strategies to MOGAs from a quantitative perspective. Our methodology makes possible to analyze if the optimization algorithm is finding sets of non-dominated solutions close to the PFs, as well as, well-distributed through the front and adapted to changes in the environment over time.

When the algorithm fails to fulfill such goals, our methodology allows identifying the underlying causes via a new visualization technique. This technique is inspired by the works proposed to analyze the performance of cellular genetic algorithms [20]. Specifically, the *takeover* time [21] and the solution's flow presented in [22]. On the one hand, the *takeover* time, which measures the time it takes a good solution to dominate a population, serves as a theoretical basis for the visualization technique since its main utility is detecting if the native or immigrant population is dominating the population too quickly. On the other hand, the flow analysis done in [22] inspired the categorization of the individuals depending on their ancestors.

This work is not the first of its kind. During the past two decades, several methods have been proposed to evaluate and compare dynamic genetic algorithms, including the ones using immigrant strategies. It is common to show graphs comparing the mean value of the objective function over time or use adaptations of accuracy metrics to dynamic environments. For example, Yu *et al.* [16] evaluate dynamic algorithms based on three metrics: the best/average performance, the best/average robustness, and the diversity. Or Morrison [23] who propose the Mean Fitness metric. Without forgetting the metrics that require knowing the best solution to a problem like the Accuracy and the Adaptability presented in [24] and [25]. It is worth nothing to mention that all the immigrant strategies work on top of genetic algorithms designed for static problems. However, all the methods proposed in the state of the art analyze the immigrant strategies in isolation, without taking into account the base genetic algorithm. The main novelty in our methodology is that it gives priority to the base genetic algorithm. With this methodology is not only possible to assess if an immigrant strategy works better than another but also assess if it improves o hinders the base algorithm in some factor.

In addition, the visualization technique proposed allows getting an insight into the evolution process that other methods cannot. It is common knowledge that random immigrants work best when the changes are ''big'' and elite immigrants work best if the changes are ''medium or small''. However, the terms ''big'', ''medium'' or ''small'' are open to debate and highly dependent on the problem. It is difficult to know when a change is big or medium for a particular problem and the visualization technique allows checking if the immigrants are working correctly whatever the size of the changes are. It is true that there are metrics that try to give understanding about the immigrant dynamics during the evolutionary process, like the ''immigrant effect'' proposed in [26] or the aforementioned ''diversity'' [16]. But it is also true, that those metrics were not designed with that goal in mind, notice that the goal of the ''immigrant effect'' is to adapt the immigrant insertion. Which, in our opinion, makes our technique more flexible and powerful.

Therefore, the main contributions of this work can be briefly summarised as follows:

1) A **new methodology** for conducting in-depth and quantitative evaluations of the performance of immigrant strategies is proposed on the basis of three different dimensions: the *quality* of the solutions, the convergence *speed*, and *stability* of the results obtained.
2) A **new visualization analysis technique** for studying how the solutions of the populations are mixed, as well as, the status of the immigrant individuals during the evolution of the MOGA, is presented.
3) An **empirical validation** of our proposed methodology using a real-world test case is conducted using a MOGA that employ different immigrant schemes in a Dynamic Community Detection (DCD) problem.

The reminder of the article is organized as follows. A detailed description of the methodology proposed in this paper is presented in Section II. Section III, introduces the dynamic community detection problem as a case study for validating the methodology presented. This section shows how our proposal can be used for an in-depth analysis of different immigrant schemes while addressing a real-world problem. Finally, Section IV outlines the main conclusions of this work and the future lines of work.

II. METHODOLOGY

The methodology proposed for evaluating the performance of immigrant strategies is based on the usage of three different metrics or dimensions: *Quality*, *Speed*, and *Stability*. The next subsections describe these dimensions, and the workflow followed to assess them. A requirement to apply our methodology is that two or more different immigrant strategies are under study. Since our methodology is proposed for analyzing the performance over several instances, we assume that there are N different instances of the same problem previously selected. For example, N dynamic networks when considering the Dynamic Community Detection Problem or N jobs/machines configurations when working on Job-shop scheduling problems. In addition, to obtain robust statistical evidence, we assume that each problem has been executed independently M times using each of the immigrant strategies selected. It should be noted that N and M should be chosen taking into account the type of problem and the desired statistical significance. Finally, we take for granted that the changes occurring during each of the N problems are a priori known. From these samples, our approach assess the performance of the strategies under study.

A. DIMENSIONS

As aforementioned, each immigrant scheme is evaluated using three different dimensions: *Quality*, *Speed* and *Stability*. Each dimension is calculated using the results from the M executions of each instance of the problem independently. Figure 2 shows the workflow for calculating each dimension. These dimensions are described in detail below:

- *Quality*. This dimension measures the quality of the Pareto Fronts generated by an immigrant strategy. One of the most popular metrics for measuring the quality

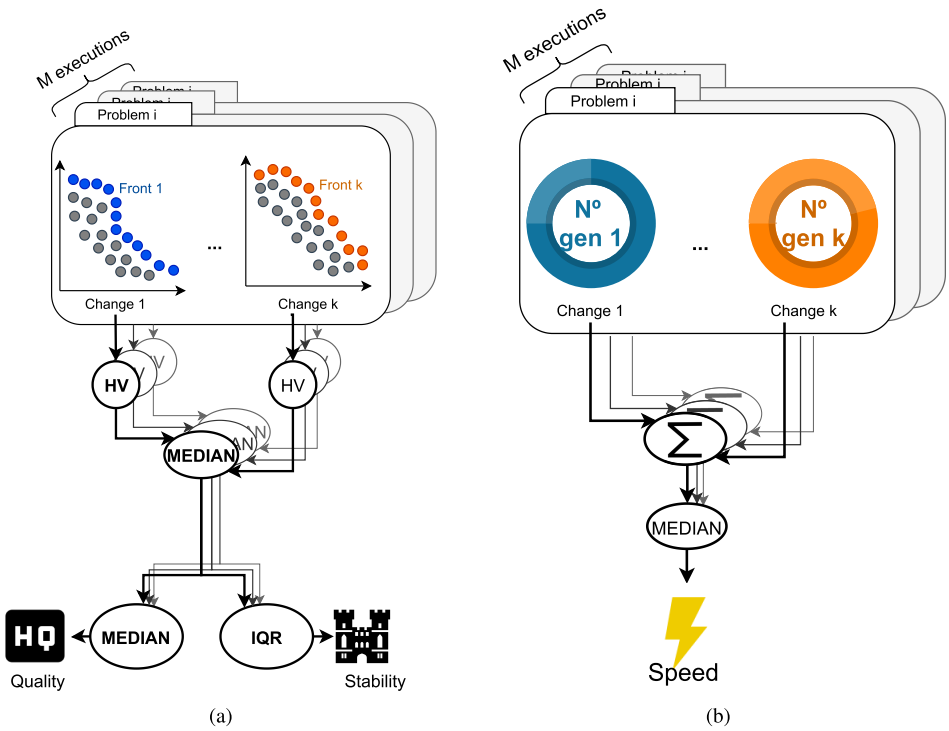


FIGURE 2. **(a) Workflow for assessing the quality and stability dimensions. b) Workflow to assess the speed dimension.**

of Pareto Fronts, and the one used in our methodology, is the Hypervolume (HV) [27]. The HV measures the area of the objective space covered by a non-dominated set with respect to a reference point. In order to make a fair comparison of the HV obtained when working with different objective functions and problem instances, a common reference point must be set and each Pareto Front obtained by the algorithm needs to be normalized. This normalization uses the maximum and the minimum values found by all the immigrant strategies in all the problem instances and all the executions. To calculate the Quality metric of a particular instance, first, the median of the HV of the normalized Pareto front generated after each change for each execution of a problem instance is calculated. Next, the median of the M executions is used as the final *Quality* metric value. It should be noted that the higher the value of this metric, the better the performance of the immigrant strategy.

- *Speed*: This dimension measures the convergence speed of an immigrant strategy. This is a crucial aspect, since the whole purpose of the immigrant technique is to reuse previous information (i.e. knowledge) to accelerate the evolutionary process. To measure the Speed metric for an instance of the problem, first, the number of generations required to converge after each environment change is calculated. A method is faster than another method when a fewer number of generations is needed to converge. For this reason, we total the values previously calculated, i.e., we calculate the sum of the number of generations until the algorithm converges after each environment change. Finally, the median of this value over the M repetitions is used as the Speed dimension value.
- *Stability*. This dimension measures the reliability of the results of an immigrant strategy. The application of an immigrant technique could influence the exploration capabilities of the MOGA, making it less or more prone to get stuck in local minima. When the latter happens, the MOGA becomes less reliable, as a single run of the evolutionary algorithm could give a worse result with a higher probability. The procedure to calculate this metric is the same as in the *Quality* metric, except for the last step. For the *Stability* metric, instead of using the median of the M executions of each test case, the Interquartile Range (IQR) of the values is used. The IQR is a measure of the dispersion of the data. Therefore, the higher the value, the lower the stability, causing that the quality of the solutions obtained has a higher variability between different runs.

B. WORKFLOW

The workflow of the methodology proposed is composed of three steps. First, all the immigrant strategies under study are compared against the same MOGA without immigrants. From now on, we will refer to this algorithm as *Standard*. Then, in the second step, all the immigrant strategies are

compared against each other, in order to analyze which ones present better results. Finally, a fine-grain analysis takes place, where specific problem instances are evaluated independently using the *Population mixture visualization technique*. Next, we detail each of the steps of the workflow proposed.

In the first step, all the immigrant strategies are compared against *Standard* for each dimension independently. To this end, the Friedman Signed Ranked Test and the Holm post-hoc analysis [28] are applied over the scores obtained by each strategy in all the problem instances. This allows discerning whether an immigrant strategy improves or hinders some dimension. An immigrant strategy, to be considered successful, should improve the *Speed* dimension while maintaining a similar or better performance in the *Stability* and *Quality* metrics. Since the main purpose of an immigrant strategy is speeding the convergence when using MOGAs in dynamic environments, we consider mandatory the improvement of the *Speed* dimension. However, applying immigrants techniques should not hinder the exploration capabilities of a MOGA. Therefore, we expect the algorithm to perform at least as good as the base technique (the one that does not use immigrants).

Regarding the visualization of these results, we recommend presenting this information on a table where the rows are the different immigrant strategies and the columns are the three dimensions. Table 5 shows an example of this visualization. Each cell of these columns contains a green triangle pointing up (▲) if the rank obtained by the immigrant strategy is higher (lower in case of minimization) than the one obtained by *Standard* and there is enough statistical evidence to support that the immigrant strategy performs differently from *Standard*. On the contrary, a cell contains a red triangle pointing down (▼) if the rank obtained is lower (higher in case of minimization). Finally, a cell contains the approximated sign (≈) when there is not enough statistical evidence to consider that the immigrant strategy performs differently from *Standard*.

The previous step allows understanding if a strategy improves or hinders some dimension. However, it does not give any information on the differences of the metric values for each of the algorithms. Hence, in the second step, all the outputs of the different immigrant strategies are compared against each other using a Friedman Signed Ranked Test and a Shaffer post-hoc analysis [28]. Regarding the visualization of this step, we suggest depicting the relation between the results obtained by each of the different strategies using a graph. Figure 3 shows an example of the proposed visualization. In this graph, the nodes represent the different immigrant strategies, and the edges join two nodes when there is not enough statistical evidence to consider that the two methods perform differently. In addition, the size of the nodes is inverse proportional to the rank obtained by each strategy for the quality dimension and directly proportional for the speed and stability ones, i.e., bigger nodes imply a better performance.

Finally, for the strategies that do not work as intended, the methodology includes the third step, where a fine-grain analysis is performed. First, problem instances that present difficulties are identified following the same process as in the first step, except that each problem instance is evaluated independently. It should be noted that that in this step, instead of using the median of the M executions of the different dimensions, the M values are used. Once the problem instances that are hard to solve for the strategy have been identified, it has to be identified in which periods, regarding the time stamps or environment changes, the evolution process is not working properly. This is done using HV evolution (Figure 4 (b)) and Empirical Attainment Surfaces [29] (Figure 4 (c)) plots. On the one hand, the HV evolution allows identifying if a particular strategy is suffering from premature convergence. On the other hand, the Empirical Attainment Surfaces, which graphically shows a boundary between dominated and non-dominated points, allows identifying areas of the Pareto Front that are not being explored enough by the strategy. Finally, insight about what is hindering the evolution process can be obtained using the *Population mixture* visualization technique on those specific periods. This technique helps to discern whether the immigrants of a strategy are contributing to the improvement of the population, are taking over the entire population, or, on the contrary, are being excluded from the population.

C. POPULATION MIXTURE VISUALIZATION TECHNIQUE

This section describes the visualization technique designed to analyze what is going on with the immigrants during the evolution. To this end, each individual of the population is labeled with a particular type. When two individuals mate, their types determine the new type of their offspring. During the evolution, the number of individuals of each type in the population is counted on each generation. Finally, these values are plotted as curves where one axis contains the generation (first, second, third, and so on), while the other axis has the number of individuals in the population of a particular type. In addition, the curve for each type of individual receives a different color. Figure 7 shows several examples of this kind of visualization.

The available label types for an individual are:

(1) *n (native)*: an original individual of the genetic algorithm.
(2) *i (immigrant)*: individuals external to the original population, which are continuously inserted into the population.
(3) *i+i (immigrant's offspring)*: individuals whose parents are two immigrants or immigrant's offspring.
(4) *n+n (native's offspring)*: individuals whose parents are two native or native's offspring.
(5) *n+i (half-blood)*: an individual which one of the parents is a native or native's offspring, while the other parent is an immigrant or immigrant's offspring.

Table 1 shows the five proposed types of immigrants and the type of the resulting offspring after they are generated

TABLE 1. The rows and columns show the type of the parents, while the cells depict the type of the resulting offspring after crossover. Since the table is symmetric, only the upper diagonal of the table is shown, for the sake of simplicity.

	i	**i+i**	**n+i**	**n+n**	**n**
i	i+i	i+i	i+i	n+i	n+i
i+i		i+i	i+i	n+i	n+i
n+i			n+i	n+n	n+n
n+n				n+n	n+n
n					n+n

by mating. For example, when two parents of type *'i'* mate (first row and column of the table), the resulting offspring is of type *'i+i'*. Likewise, if for example, the parents have types *'n+n'* and *'n+i'* (third row and fourth column in the table), the resulting offspring is of type *'n+n'*.

Analyzing the curves that show the evolution of the different types of individuals during the execution, can help to understand whether the immigrants are contributing to the evolution of the population, or are taking over or being excluded from it. Specifically, the presence of a half-blood ($n+i$) population curve indicates that a healthy mixture of both populations, native and immigrant, is taken place. Hence, the immigrants are contributing to the local population. However, on the one hand, the over-representation of the immigrant's offspring ($i+i$) population curve indicates that the immigrants are taking over the native population. If this effect is too extreme, all the native population is expelled from the evolutionary process without a chance of passing their genes to the next generations, which is equivalent to evolve the immigrant population alone. On the other hand, the over-representation of native's offspring ($n+n$) population curve indicates that the immigrants are not contributing to the evolution process. In extreme cases, all the immigrants can be expelled from the population without having a chance to pass their genes to the next generations. This is equivalent to using a smaller population because, even though the immigrants are part of the population, they are not allowed to participate in the evolution process. These two latter cases usually result in a loss of performance by the immigrant strategy.

III. CASE STUDY: DYNAMIC COMMUNITY DETECTION

In order to test the validity of the proposed methodology, a test case is studied. In particular, we applied our methodology for studying a Multi-Objective Genetic Algorithm that combines an Immigrant's scheme with local search strategies to the dynamic community detection problem [30]. In this case, we use the proposed methodology for evaluating and selecting the most suitable immigrant scheme.

The remainder of this section first introduces some preliminaries on the DCD problem. Then, the problem instances used to conduct all the experiments are introduced. After that, the particularities of the MOGA for solving the DCDP are explained. Finally, the results obtained by applying the proposed methodology are presented and a detailed analysis is performed.

A. BASICS ON DYNAMIC COMMUNITY DETECTION

The Dynamic Community Detection problem is the dynamic version of the Community Detection Problem (CDP), which consists in splitting a network into groups of nodes, usually known as communities. In a nutshell, it could be considered as clustering in graphs. There is no an unique definition of *Community* agreed upon by the scientific community. Nevertheless, a widely accepted one is that nodes belonging to the same community must be strongly interconnected, while maintaining sparse connections to nodes of other communities.

Static Community Detection is a hot research area and a lot of different methods have been proposed to tackle this problem over the years [31]–[34]. Of all the strategies used to detect communities, bio-inspired meta-heuristics have proven great success [35], specially genetic algorithms both on the mono-objective [36] and multi-objective [37], [38] configurations.

However, as in so many other areas, the dynamic version of the community detection problem has been neglected until recently. The exponential growth of online Social Networks has increased the demand for efficient dynamic community detection algorithms, which has boosted the interest in the area. When considering the time variable, the quantity of possible analyses broadens. When time is involved, one can be interested in finding one community structure that fits the whole network's lifetime (like in ROOT), detecting significant points on the network's timeline, or studying communities' dynamics (if they split, merge, grow ...etc.) among others. Hence, the DCD area is composed of an amalgamation of problems quite different in nature. We refer the readers interested in this topic to the review [39], where both the static and dynamic variants of the problem are discussed in extensive detail and the most common tools available are reviewed.

In this case study, we will center on the problem of finding a suitable community structure for each point of a network's timeline. The most popular framework to do that is the *evolutionary clustering* [40]. This framework aims to imbue the solutions with a certain smoothness. In evolutionary clustering, two forces must be balanced, one that pushes communities to adjust to the present moment and another that constrains them from deviating from their history. Undoubtedly, bio-inspired meta-heuristics, especially MOGAs, are one of the most popular techniques to do evolutionary clustering. Folino and Pizzuti [41], Chen *et al.* [42], Bara'a and Khoder [43], and Ma *et al.* [44] have proposed MOGAs that follow the evolutionary clustering framework. The former two are based on NSGA-II [45] and the latter two on MOEA/D [46].Outside genetic algorithms, other bio-inspired metaheuristics have also been proposed to a lesser extent. For example, Yin *et al.* [47] have proposed a Particle Swarm Optimization (PSO) with custom initialization and crossover

operators, and Zhou *et al.* [48] have proposed a Bat Algorithm (BA). However, all of the above methods define trivial criteria for selecting a single Pareto front solution, namely, they stick with the one that best fits the current network topology. Which favours one objective over another. How to make such a decision remains an open question in the area that is worth investigating.

Contrary to the examples above, the MOGA used for this case study does not follow the evolutionary clustering framework. Instead, it forgets about the recent history of the communities and focuses on their quality at the present moment. Our MOGA optimizes two different negatively correlated quality metrics, as recommended in [49]. Note that the aim of this use case is not to propose a better MOGA for DCD and that the conclusions drawn for our MOGA could easily be extrapolated to other similar MOGAs in the state of the art.

B. PROBLEM FORMULATION

Given a dynamic network modeled as a sequence of snapshots $N = \{G^0 \dots G^T\}$. Where T is the number of timestamps and each element $G^t(V^t, E^t) \wedge 0 \leq t \leq T$ of the sequence is a graph composed of a set of V^t nodes and a set $E^t = \{(u, v)|u \in V^t, v \in V^t\}$ edges. The objective of the community detection problem is to generate a set of dynamic communities $\{DC_0 \dots DC_m|DC_i = \{C_i^t|0 \leq t \leq T\}\}$ where m is the number of dynamic communities and each element of a dynamic community DC_i is in turn a static community $C_i^t = \{u|u \in V^t\}$. However, it is normal to find this problem split in two in the state of the art. Usually, a method finds promising C_i^t communities for each snapshot G^t and a different method builds the dynamic communities DC_i using those promising communities C_i^t. The methods proposed in this case study tackle the first part of the problem. Hence, given a dynamic network $N = \{G^0 \dots G^T\}$, their objective is to find a set of promising crisp static communities $\{C_0^t \dots C_n^t\} \wedge \bigcup_i = 0_n C_i^t = \emptyset$ for each snapshot G^t.

C. PROBLEM INSTANCES UNDER STUDY

For testing the performance of the different immigrant techniques, 18 different synthetic networks have been used (the N parameter in our methodology). The benchmark is composed of these dynamic networks and their corresponding ground truth. We have opted for using only synthetic networks because with them we can generate a benchmark where it can be ensured that a complete and balanced set of scenarios take place. To generate the benchmark the DANCER software has been used [50]. This software allows creating dynamic networks, configuring the number of changes that occur during the evolution of a dynamic network, as well as, the type of these changes. Two different operations are available for generating the dynamic networks:

- The *micro* operation introduces little changes in the network at the node scale (insert/delete a few edges or nodes).
- The *macro* operation introduces big changes in the network at the community scale by merging and splitting them.

Taking this into account, synthetic networks with different types of dynamics are generated to study the efficiency of different immigrant strategies for dynamic networks with different characteristics. Therefore, the synthetic networks generated use all the *micro operations* available all the time. However, for the generation of each benchmark a different combination of *macro operations* is used. It is worth mentioning that to generate each dynamic network both operations are applied with different intensities, where a higher intensity implies that more changes occurs during the evolution.

TABLE 2. Description of the different types of dynamic networks generated for the case study.

Name	Description
mx_XX	Network applying *micro operations* during its evolution with three levels of intensity: low (XX = 01), medium (XX = 02) and high (XX = 03)
mg_XX	Network applying *merge* macro operation during its evolution with three levels of intensity: low (XX = 01), medium (XX = 02) and high (XX = 03)
sp_XX	Network applying the *split* macro operation during its evolution with three levels of intensity: low (XX = 01), medium (XX = 02) and high (XX = 03)
sm_eq_XX	Network applying both the *split* and the *merge* macro operations in equal shares during its evolution with three levels of intensity: low (XX = 01), medium (XX = 02) and high (XX = 03)
sm_mm_XX	Network applying both the *split* and the *merge* macro operations during its evolution, with the *merge* one happening more frequently and at three levels of intensity: low (XX = 01), medium (XX = 02) and high (XX = 03)
sm_sm_XX	Network applying both the *split* and the *merge* macro operations during its evolution, with the *split* one happening more frequently and at three levels of intensity: low (XX = 01), medium (XX = 02) and high (XX = 03)

Table 2 shows the different synthetic dynamic networks generated for testing and the specific characteristics of each one to test. Each row of the table corresponds to one combination of macro operators. All the networks contain nine snapshots (i.e., changes in the network) with between 100-200 nodes and 200-1000 edges. The naming system for the different types of networks consists in several letters followed by a '_XX'. The letter indicate the type of operations that happens in the networks while the '_XX' indicates the intensity. The nomenclature for the intensities is 01 for *low*, 02 for *medium* and 03 for *high*.

D. MOGA PARAMETRIZATION FOR THE DCDP

The MOGA analyzed for the DCDP uses the NSGA-II algorithm [51] to optimize two objectives: the "Community

TABLE 3. MOGA operators.

OPERATORS	
Encoding	Locus based adjacency
Selection	Tournament
Crossover	Two-point crossover
Mutation	Custom function that randomly selects several genes of an individual and changes their values to a random neighbor community
Stop criteria	Single-Sample Chi-Square Test and max generations

TABLE 4. MOGA parameters.

COMMON PARAMETERS	
Tournament size	2
Tournament replace	True
Population size	300
Offspring size	300
Crossover rate	100%
Mutation rate	1% Individual length
Single-Sample Chi-Square	0.05
Max generations	300
RIGA and EIGA	
Number immigrants	10% population

Score'' [52] and the ''Average ODF'' [53]. The ''Community Score'' is an inter-community objective, which calculates how strongly the nodes of a community are connected. In particular, it measures the density of connections in relation to the number of nodes inside a community. Equation 1 shows the formula followed to calculate the Community Score. where C is the set of communities that an individual encodes; S is each of those communities; ms is the number of edges inside a community S and ns is the number of nodes in community S.

$$C\_SCORE = \sum_{S \in C} \left(\frac{2ms}{ns}\right)^2 \tag{1}$$

The ''Average ODF'' is an inter-community objective, which calculates how strongly the nodes that belong to different communities are connected. In particular, it measures the average fraction nodes' edges pointing outside their communities. The formula to calculate the Average ODF can be found in Equation 2.Where where C is the set of communities that an individual encodes; S is each of those communities; ns is the number of nodes in the community S and $d(u)$ is the degree of node u.

$$AVG\_ODF = \sum_{S \in C} \left(\frac{1}{ns} \sum_{u \in S} \left(\frac{|\{u, v | v \notin S\}|}{d(u)}\right)\right) \tag{2}$$

The MOGA encodes the individuals using the ''locus-based adjacency'' [54] encoding. In this encoding, each gene is associated to a node of the network and the allele values correspond to nodes of the graph. The genetic operators of the MOGA are the tournament selection, the two-point crossover and a specially-tailored for the problem mutation operation, which randomly selects several genes of an individual and changes their values to a random neighbor community. Finally, the stopping criteria checks the convergence of the algorithm using a Single-Sample Chi-Square Test over the HV of the past 20 Pareto fronts. In addition to this, a maximum number of generations is also used as stopping criteria, in order to assure the end of the execution when the algorithm does not converge. Notice that a summary of the operators used in all MOGAs is available at Table 3

Regarding the hyper-parameters used, which are summarized in Table 4, both the population and offspring sizes are 300 individuals. Please note that the size of the population remains constant over all the evolution process. In addition, the crossover rate is 100% and the mutation rate is set to the 1% of the genes of an individual. Regarding the immigrants, a 10% of the population is replaced with immigrants. Notice that this parameter is only used in the methods were a fixed number of immigrants are introduced into the population. Finally, the alpha used for the Single-Sample Chi-Square Test is 0.05 and the maximum number of generations is 300.

In order to calculate the HV, first the *Community Score* Objective is inverted by subtracting 1.0. Then, the Pareto Fronts are normalized as stipulated in Section 2. Next, the HV is calculated using the ideal reference point (1.1, 1.1). Finally, the number of executions of each MOGA (the M parameter in our methodology) has been set to 20.

E. APPLICATION OF THE METHODOLOGY: ANALYZING THE MOGA IN A DCD PROBLEM

In this case study, we consider the three immigrant strategies from the state of the art: RIGA, EIGA, and HIGA (which were described in Section I). In the experimental evaluation, we have also included two additional strategies, in order to set an actual comparison basis:

- Hypermutation Genetic Algorithm (HGA) [55]. This method is not based on the immigrant strategy. This algorithm, instead of continuously inserting immigrants into the population, increases the mutation rate of the population when a change occurs in the environment and then the evolution process continues normally. This method is included in our experimental evaluation since it uses a different strategy than immigrants for dealing with the changes in the environment, which is simpler and less memory consuming since just a single population needs to be stored.
- Delayed Elite Immigrants Genetic Algorithms (DEIGA). This method is based on the immigrant strategy and its similar to EIGA, but with the difference that immigrants are only inserted into the population after a fixed number of generations. This method is included in the study since it allows understanding if the immigrant and native populations explore different sections of the search space. With this method we can analyze if given enough

time, both immigrant and native populations converge to the same Pareto fronts or to different ones.

1) FIRST STEP: COARSE-GRAIN ANALYSIS AGAINST STANDARD

As aforementioned, the first step of the methodology is to compare the performance of the immigrant techniques against the performance of the Standard method. This allows evaluating if the algorithm being tested is able to achieve better results than the base algorithm (without any immigrant schemes).

Table 5 shows the results of the comparative performance of the five methods considered in the study with respect to the *Standard* method. This comparison is based on the usage of the three dimensions proposed in the methodology: quality, speed and stability. The numerical results are mixed with regard to the *quality* of the solutions found. On the one hand, out of the three immigrant techniques under study, EIGA is the only one that improves over *Standard*, while RIGA and HIGA exhibit similar or worse performance, respectively. On the other hand, between the other two methods, DEIGA shows a better performance, while HGA exhibits a similar one. The fact that DEIGA has a better numerical performance than *Standard* was expected as EIGA also does, and DEIGA is just a EIGA with a delay on the insertion of the immigrants. However, the fact that HGA does not improve the results reveals that the gradual insertion of individuals into a population brings benefits to the search mechanism over the direct re-utilization of past populations in DCD problems. Regarding the *speed* of the algorithms, all the techniques studied show a better convergence *speed* than *Standard*. This was an expected result as the main reason for applying a mechanism that reuses past populations is improving the convergence speed by reusing past knowledge. Finally, contrary to what happened with the *speed* dimension, all the algorithms considered in the experimental evaluation present a worse *stability* than the *Standard* algorithm. This implies that the HV of the Pareto Fronts obtained by all the methods have greater variations between consecutive executions compared to the control method. This is a sign that reusing individuals from past populations could deteriorate the exploration capabilities of a MOGA, making it more prone to get stuck in a local minimum/maximum. Please notice that not only do the immigrant strategies suffer from poorer stability but also does the HGA and the Delayed EIGA.

2) SECOND STEP: COARSE-GRAIN ANALYSIS BETWEEN IMMIGRANT STRATEGIES

The second step of the methodology consist in conducting a comparative study between all the techniques under study using the Friedman Signed Ranked Test and Shaffer post-hoc. The study is carried out on the basis of the three dimensions: quality, speed, and stability. The results obtained are presented using graphs in Figure 3. Each node represents a different technique, where the size of the node is inversely proportional to the rank obtained by each method in the Friedman Signed Ranked Test in the Quality dimension and directly proportional in the Stability and Speed ones, i.e., bigger nodes imply a better performance. Two nodes are connected by an edge if the Shaffer post-hoc analysis shows that there is no statistical evidence that the two methods perform differently (i.e., the two methods can be considered equal).

TABLE 5. Results for the Friedman Signed Ranked Test and the Holm post-hoc using STD as the control method. The first group contains the three immigrant strategies, and the second one the rest of the methods. '▲' '▲' indicates that a method performs better than STD. Contrary, performs worse than STD. Finally, '≈' indicates that the method performs equally to STD.

Method	Stability	Quality	Speed
RIGA	▼	≈	▲
HIGA	▼	▼	▲
EIGA	▼	▲	▲
HGA	▼	≈	▲
DEIGA	▼	▲	▲

The previous step revealed that EIGA and DEIGA are superior to *Standard* regarding the quality, while RIGA and HGA have a similar performance than the control method. Nonetheless, the second step of the methodology reveals that HGA performs better than RIGA since the HGA node is bigger than the RIGA node in Figure 3 (a) and there is no edge between RIGA and HGA. In addition, this step also allows appreciating that both EIGA and DEIGA have a similar numerical performance (there is an edge between both nodes). From these results, we can conclude that for this problem is better to reuse previous populations with a mutation phase than inserting random immigrants during the evolution process because HGA performs better than RIGA. However, we cannot draw any conclusion about if the immigrant and native population explore different areas of the search space, as both EIGA and DEIGA have a similar performance.

Next, regarding the *Speed* dimension, from the previous step we know that all the methods are faster than *Standard*. However, this step shows that are two different groups (see Figure 3 (b)). The first group is composed of RIGA, HGA, and EIGA, whose nodes form a clique. The second group is composed of HIGA and DEIGA, whose nodes form a kind of tail. From these results, we can conclude that inserting random (RIGA) or elite (EIGA) immigrants exhibit a similar convergence speed than directly re-using previous populations (HGA). Surprisingly, the HIGA method, which uses a combination of elite and random immigrants, presents a worse convergence speed than both of the other immigrant techniques.

Finally, regarding the *Stability* dimension, the previous step of the methodology has shown that all the algorithms considered have worse *stability* than *Standard*. Analyzing the results obtained in this step, shown in Figure 3 (c), we can conclude that all the techniques perform equally bad

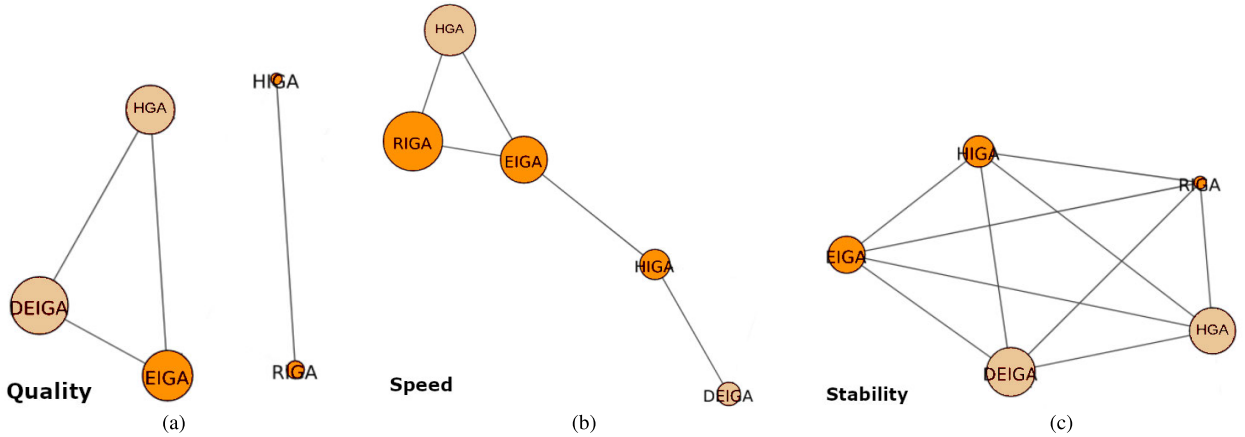


FIGURE 3. From left to right, the results of the Friedman Signed Ranked Test and the Shaffer post-hoc analysis for the Quality, Speed, and Stability metrics. Each node represents a different algorithm. The nodes in dark color correspond to the immigrant techniques under study and the light ones to the other methods. The size of the nodes is inversely proportional to its Friedman test rank in the Quality dimension and directly proportional in the Speed and Stability ones (i.e., bigger nodes have a better performance). Two nodes are connected when the difference between two methods are not statistically significant.

regarding this dimension being that the resulting graph forms a clique.

Summarizing, from the results obtained in the first two steps for the problem at stake, it can be concluded that all the immigrant techniques exhibit worse *stability* compared to the *Standard* method and there are no differences between them. In addition, it can also be concluded that gradually inserting elite immigrants into the population has produced better results compared to directly reusing previous populations. However, this is not the case when inserting random immigrants. Surprisingly, the method that combines the use of both elite and random immigrants (HIGA) shows the worse results and even has a worse performance in terms of *quality* than *Standard*. Finally, with the results obtained so far, it can not be evaluated if the native and immigrant populations explore different search space areas, as EIGA and DEIGA have a similar performance in all aspects except for speed (which it was expected due to the delay in the insertion of immigrants in DEIGA).

3) THIRD STEP: FINE-GRAIN ANALYSIS

Following the third step of the methodology, the immigrant strategies are evaluated on specific problem instances were difficulties are found. In order to carry out this analysis, first, a study of the quality of the Pareto Fronts found by the algorithms is carried out to find periods where the algorithm is not working properly. Next, an in-depth analysis of those periods is carried out using the *Population mixture visualization technique* (Section II-C). This visualization technique allows examining what could be causing these difficulties.

Following the methodology, first, the performance of the immigrant techniques for each dynamic network generated is evaluated on the basis of the Quality metric to identify which immigrant strategies are performing poorly. Table 6 shows the quality obtained by each immigrant method compared to the standard method for each problem instance tested. A green triangle pointing up (▲) indicates that a method performs better than STD. Contrary, a red triangle pointing down (▼) indicates that a method performs worse than STD. Finally, the approximated sign (≈) indicates that the method performs equally to STD. The evaluation has been done using the Friedman Signed Ranked Test and the Holm post-hoc analysis using as data the 20 Quality values (one per execution of a problem instance) obtained by each strategy. Each quality value is calculated by doing the median of the HV obtained at each of the nine snapshots that form a dynamic network.

TABLE 6. Results for the Friedman Signed Ranked Test and the Holm post-hoc on the HV using STD as the control method.

Instance	HIGA	RIGA	EIGA	HGA	DEIGA
mg_01	▼	▼	≈	≈	≈
mg_02	▼	▼	≈	≈	≈
mg_03	▼	▼	≈	≈	≈
mx_01	≈	≈	▲	▲	▲
mx_02	≈	≈	▲	▲	▲
mx_03	≈	≈	▲	▲	▲
sm_eq_01	▼	▼	▼	▼	≈
sm_eq_02	▼	▼	▼	▼	≈
sm_eq_03	▼	≈	▲	▲	▲
sm_mm_01	▼	≈	▲	≈	≈
sm_mm_02	▼	▼	▼	▼	≈
sm_mm_03	≈	≈	▲	▲	▲
sm_sm_01	▼	▼	≈	≈	▲
sm_sm_02	▼	▼	≈	≈	≈
sm_sm_03	▼	≈	▲	≈	▲
sp_01	▼	≈	▲	▲	▲
sp_02	▼	≈	▲	≈	▲
sp_03	▼	▼	≈	≈	≈

The results from Table 6 reinforce the conclusions drawn in the two previous steps, using elite immigrants does have benefits over reusing past populations, while using random immigrants does not. It should be noted that HIGA and RIGA have a worse numerical performance than HGA, which in

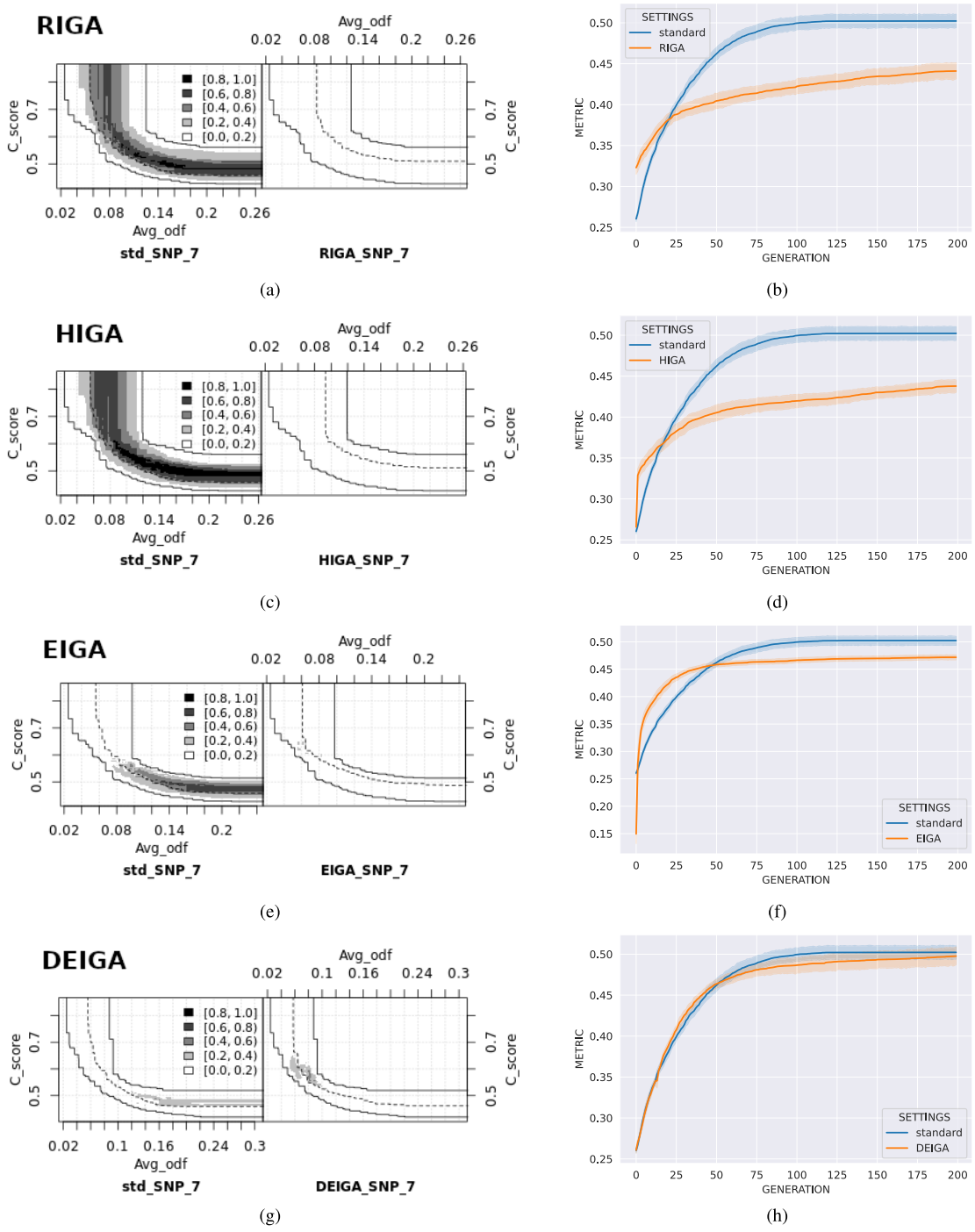


FIGURE 4. Left figures (a, c, e, g) shows the attainment surfaces of the final Pareto Fronts found by the STD method and the immigrant strategies. The STD method is shown on the left panel of the attainment surfaces. Right figures (b, d, f, h) shows the evolution of the HV over the different generations of the MOGA for the same methods. The STD method is shown in orange, while the immigrant strategy is shown in blue.

turn performs worse than EIGA. However, it can be seen that there are several networks for which all of the immigrant techniques under study obtain worse *quality* than the baseline. In particular, the networks *sm_eq_01*, *sm_eq_02*, and *sm_mm_01* are of special interest because all the methods fail except DEIGA. To further analyze the previous phenomena, the rest of the section studies the behavior of each of the immigrant techniques on the *sm_eq_01* network. Only the results of the *sm_eq_01* network are shown because the conclusion obtained by analyzing the others are similar.

With the above in mind, in a second step, the attainment surfaces and the HV evolution of the Pareto Fronts found

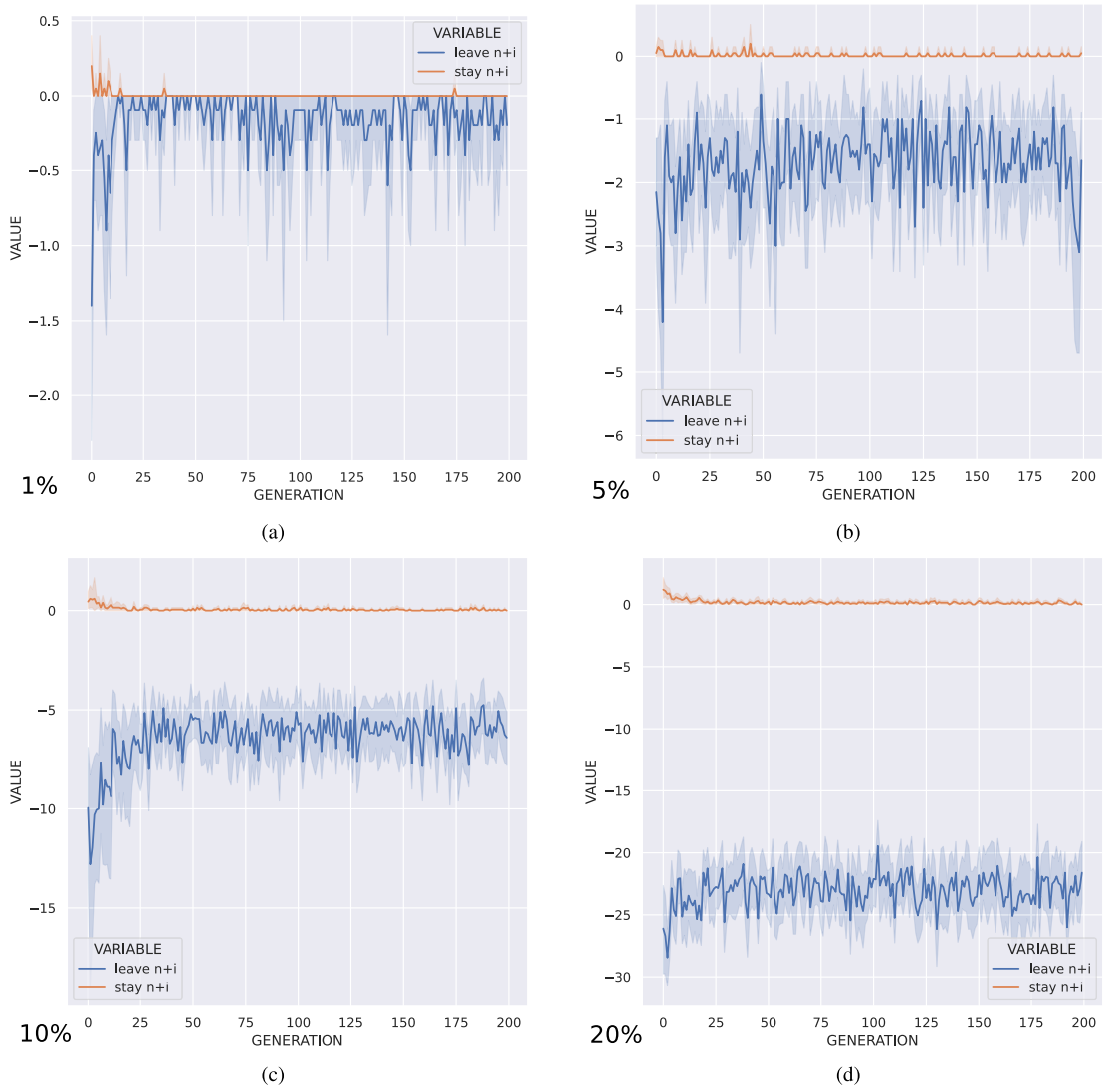


FIGURE 5. Number of *half-blood* individuals for the RIGA method that survive (in orange) and are expelled (in blue) of the population between two generations. The X axis shows the number of generation and the Y axis the number of individuals of a particular type. The percentage on the bottom right corner of the plot indicates the percentage of immigrants inserted into the population at each generation.

at each snapshot of the network *sm_eq_01* are evaluated. This allows identifying in which snapshots the algorithms are working properly in which they don't. The network under study is composed of nine snapshots. All the methods included in the study perform well in all the snapshots, except for the last three. We focus our analysis on the results of the seventh snapshot, even though the same effect could be found in the other two ones.

As aforementioned, Figure 4 only shows the attainment surfaces and the HV evolution of the immigrant techniques obtained by all the immigrant techniques in the seventh snapshot of the *sm_eq_01* network. The attainment surfaces plot shows that all the techniques except for DEIGA perform worse than *Standard*; notice the dark areas on the left attainment surfaces. Moreover, in the HV evolution plots, it can be seen that all immigrant techniques except for DEIGA suffer from premature convergence to a greater or lesser extent. It should be noted that the orange line of the HV plots gets stuck in a local minimum compared to the blue one. To further study why the different immigrant techniques are not preventing this premature convergence, each immigrant technique is studied independently using the populations mixture visualization technique:

- Starting with the RIGA technique, the Figure 5 shows the number of *half-blood* (n+i) individuals (i.e., an offspring of a native and an immigrant individual) that continues from one generation to the next (in orange) and that are removed from the population (in blue). The different sub-figures in Figure 5 correspond to different percentages of immigrants inserted into the population

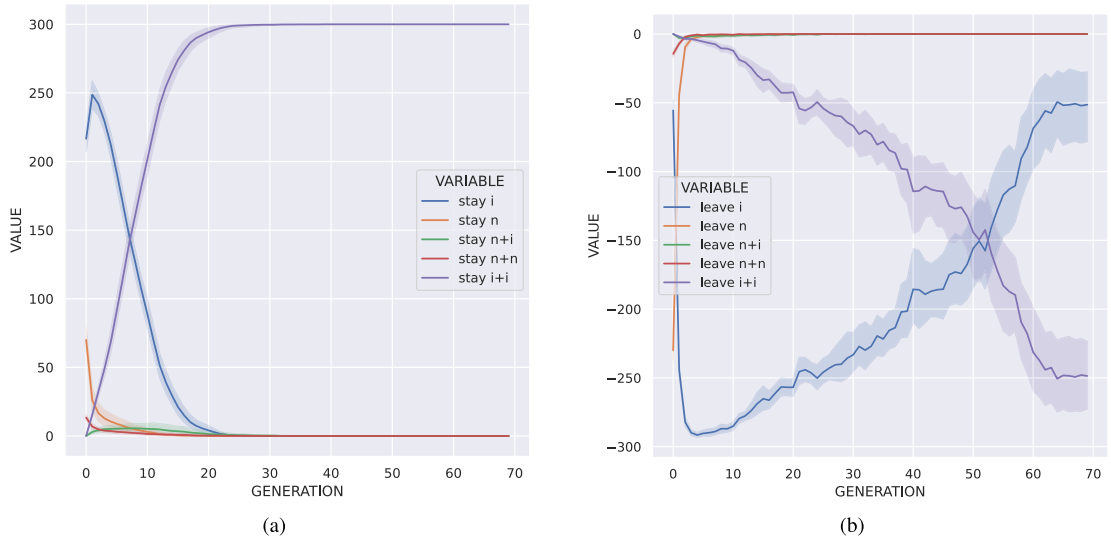


FIGURE 6. Number of individuals of different types for the HIGA method. The X axis show the number of generation during the evolutionary process and the Y axis the number of individuals of a particular type. Sub-figure (a) shows the number of individuals that survive from one generation to the next. Sub-figure (b) shows the number of individuals that are expelled from the population.

at each generation; 1% (3 individuals), 5% (15 individuals), 10% (30 individuals) and 20% (60 individuals) of the population are replaced by immigrants is shown in sub-figures (a), (b), (c), and (d), respectively. As it is shown in the figure, the mixture between natives (individuals from the previous snapshot in the case of RIGA) and immigrants (completely random individuals in the case of RIGA) have not enough quality and are almost completely removed from the population in the same generation that are created. Moreover, the percentage of immigrants inserted into the population does not change this effect. Therefore, it is hard that two immigrant individuals mate in RIGA and almost impossible that their offsprings continue to the next generation. Since the immigrants are just being expelled from the population without having a chance of passing their genes to the next generation, they are essentially wasting space on the population. As a result, in practice, the RIGA technique in the DCD problem is equivalent to having a smaller population, which produces the fast convergence speed and the low quality of the solutions already found in Section III-E1.

- Regarding the HIGA technique, Figure 6 shows the number of individuals of the different types during the evolutionary process for the HIGA method. The figure on the left (sub-figure (a)) shows the number of individuals that continue from one generation to the next, while the one on the right (sub-figure (b)) shows the ones that are being expelled from the population. The sub-figure (a) shows that the HIGA method inserts a large number of immigrants (individuals from the previous snapshot) during the first generations of the evolution. In fact, it inserts so many immigrants that the native population becomes extinct after the fifth generation. Nevertheless, analyzing the sub-figure (b), it can be noticed that after the native population has become extinct, the number of immigrant individuals that leave the population decreases and then plateaus. It is not unreasonable to hypothesize that this plateau is produced by the algorithm trying to enhance its exploration capabilities by adding random immigrants, which is the same technique that RIGA uses. Summarizing, the HIGA method is too aggressive at the beginning of the evolutionary process and removes all the native individuals from the population, essentially behaving like HGA. After that, it tries to enhance the population diversity by introducing random individuals, just like RIGA, which results in the immigrants just using space and not contributing to the exploration (i.e., it is like using a smaller population). This behavior explains the bad quality of the results of HIGA, just like RIGA. However, contrary to RIGA, HIGA does it gradually during the evolutionary process, hence the lower convergence speed.
- With respect to EIGA, Figure 7 shows the number of individuals of different types for a particular generation. Each sub-figure shows how the different types vary for distinct percentages of immigrant individuals. Contrary to the other methods, in EIGA there is an adequate mixture between native and immigrant individuals, which can be noticed in the half-blood line (the n+i green line). As expected, there is a transition phase where the two populations (immigrant and native) mate, producing valid offsprings. After that period, both populations continue mixing, but they result in solutions with lower quality that are expelled from the population. The length

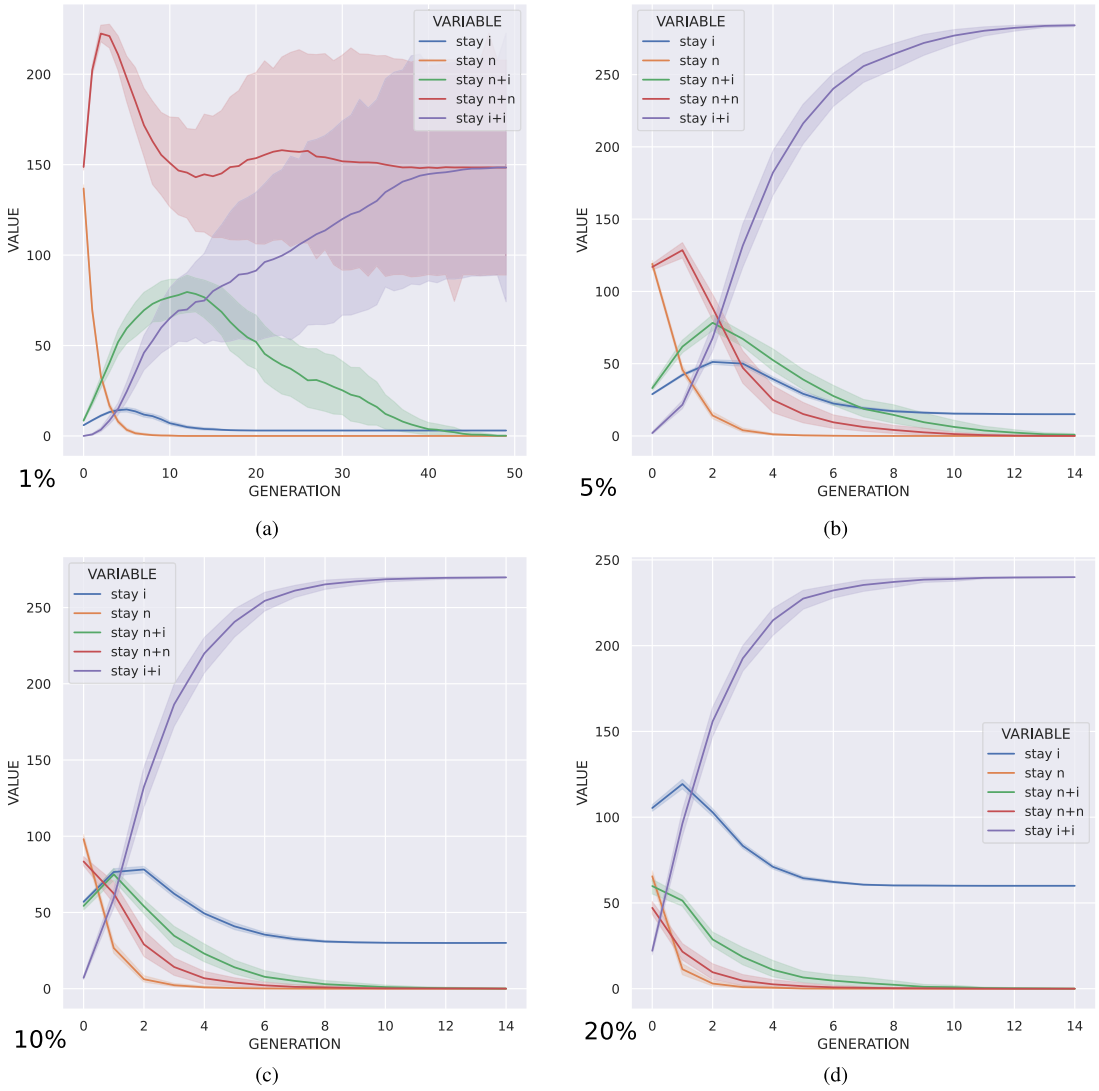


FIGURE 7. Number of individuals of different types for the EIGA method. The X axis show the number of generation during the evolutionary process and the Y axis the number of individuals of a particular type. The percentage on the top right corner of the plot indicates the percentage of immigrants inserted into the population at each generation.

of this mixing period varies depending on the percentage of immigrant individuals inserted into the population, the higher the percentage of immigrants, the shorter the period. Furthermore, for percentages as high as 5%, the immigrant population (or their descendants) take over the population (sub-figures (b), (c), and (d)). However, when the percentage is only 1% (subfigure (a)), it can be seen that no population fully dominates the other. The evolutionary process reaches an equilibrium where the immigrant population (or their descendants) occupies a part of the population, while the native one occupies the rest. This fact suggest that the immigrant and native population could explore different regions of the landscape. Summarizing, the EIGA strategy is the only immigrant technique from the study that achieves an adequate mixture between populations, hence its better results and fast convergence speed, combining the quality of HGA with the speed of convergence of RIGA.

- Finally, we analyze DEIGA, which was included for comparative reasons. As aforementioned, this method works like EIGA, but the insertion of immigrants is delayed a certain number of generations. From the results shown in Figure 7 sub-figure (a), it can be hypothesized that if the native population is allowed to evolve for a certain number of generations, so they can better adapt to the environment, the mixture of individuals could produce better results. DEIGA method allows checking this hypothesis. Figure 8 shows the number of individuals of different types during the evolutionary process for DEIGA. Each sub-figure shows how the different types varies when waiting for a different number of generations, which is shown at the top right corner.

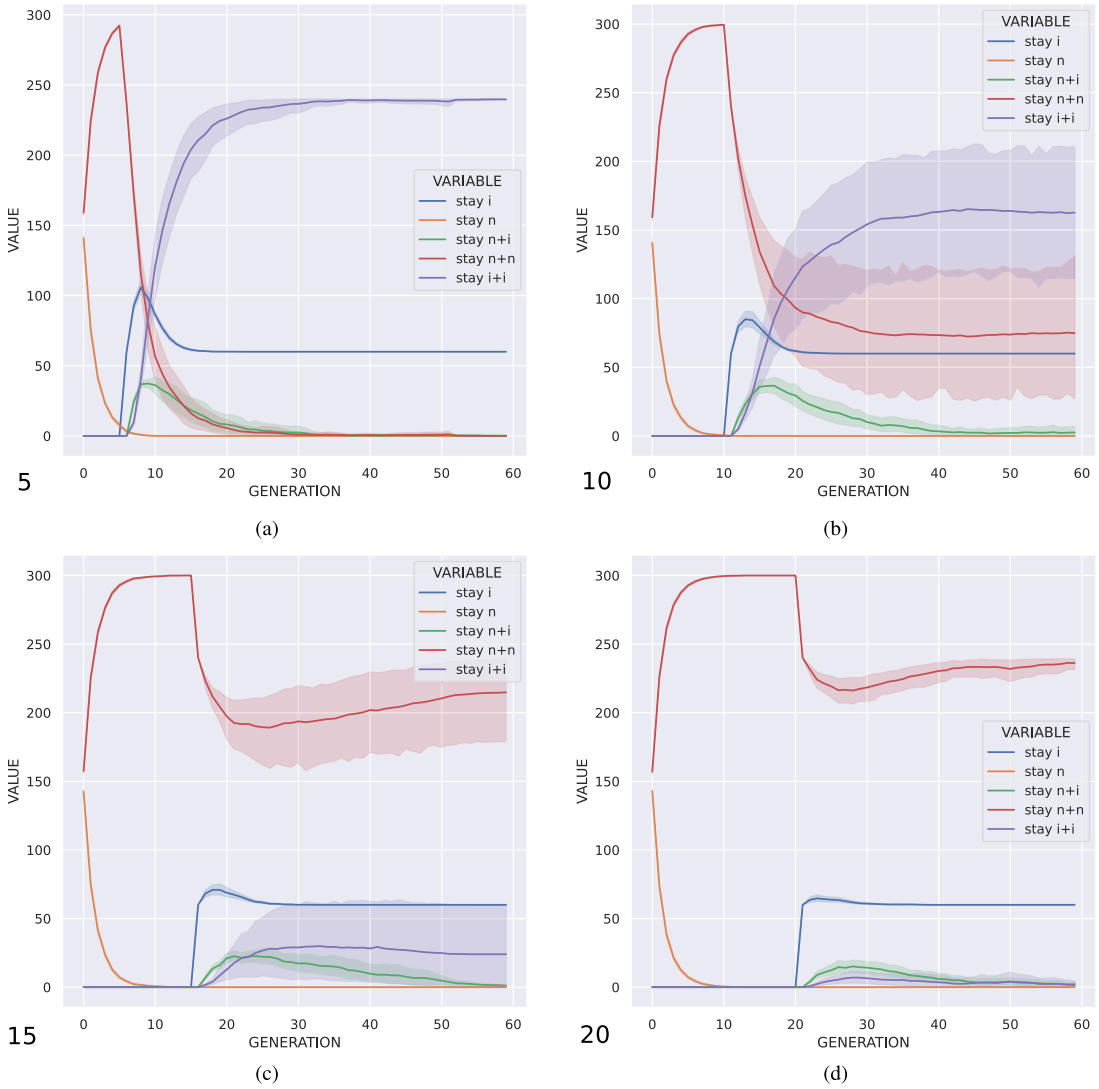


FIGURE 8. Number of individuals of different types for the DEIGA method. The X axis shows the number of generation during the evolutionary process whereas the Y axis shows the number of individuals of a particular type. The number on the top right corner of the plot indicates the number of generations waited until the immigrants are inserted into the population. In all plots a 10% of immigrant individuals are inserted at each generation.

In all plots a 10% of immigrant individuals are inserted on each generation. As it can be seen, even waiting a small number of generations (e.g., 10) prevents the immigrant population from taking over the whole population (notice the difference with Figure 7 (c)). Also, the larger the number of generations DEIGA waits to insert the immigrants, the smaller is the size difference between the populations when an equilibrium is reached. However, if the value is too large, as in the sub-figure (e), the opposite effect is produced since the immigrants of the population have not enough quality to occupy their own space in the population. Nevertheless, when the immigrant solutions enter the population, there is always a mixture between both type of individuals (the n+i green line is present in all the plots). Summarizing, letting the population evolve some generations before inserting immigrant solutions produces the best results as prevents the immigrants from taking over all the population and allows a good mixture between populations. However, waiting too many generations produces the opposite effect and prevents the immigrant solutions from entering into the population. Nevertheless, even in those cases, some kind of mixture between the populations always appears.

F. CONCLUSION OF THE EXPERIMENTAL ANALYSIS

The experimental evaluation has shown that both RIGA and HIGA have a low numerical performance. The fine-grain analysis carried out revealed that the insertion of completely random individuals into an already established population produces offspring of so little quality, compare to the rest, that they are immediately discarded from the population.

In practice, these methods are reducing the size of the population since the immigrant solutions are discarded without contributing to the exploratory capabilities of the algorithm. However, the EIGA method shows better results as it is the only method where an adequate mixture between the native and immigrant populations is achieved. Nevertheless, the percentage of immigrants inserted in the population affects the quality of the numerical results. If too many immigrants are inserted, the selection pressure of them is too high and they over take the whole population, making the native solutions extinct. Surprisingly, when the percentage of immigrants inserted is not too high, a equilibrium is achieved and both population continue evolving independently.

To further study this phenomenon the EIGA algorithm was evaluated including a delay in the insertion of immigrants (DEIGA). These experiments have shown that waiting as little as ten generations before inserting the immigrants produces an equilibrium where the immigrants and the native population continue evolving independently, even for high percentages of immigrant insertions like 10% of the population and the further the algorithm waits, the lower is the different in size between both populations during the equilibrium. However, if the algorithm waits too many generations, the opposite effect happens and the immigrant individuals have not enough quality to enter the population. Nevertheless, a mixture between populations is always present.

The appearance of this equilibrium where both populations evolve in parallel, suggest that, for this particular setup (multi-objective genetic algorithms for dynamic community detection), the immigrant and the native solutions have different exploration capabilities and in order to achieve the best results the population needs some time to mature. For this reason, it is crucial that the selection pressure of the algorithm does not expel the native solutions from the population too fast, before the individuals are exposed to other type of individuals.

However, the results presented in this section should be considered preliminary for the DCD field, as they have only been tested using synthetic networks. For more robust results, benchmarks composed of real networks are also needed. However, at the time of writing, finding them is not an easy task. In addition, all the conclusions presented here are only relevant when estimating the capability of a method to find non-dominated sets of solutions. Which solutions should be selected from each non-dominated set to form a good dynamic community is still an open question in the area of dynamic community detection. Depending on the criteria followed, the results could change. Hence, we cannot assure that the method that generates better non-dominated fronts will give the best community dynamics. At least until some good criteria is found and tested. However, as there is no defined criteria jet, is not a far stretch if we hypothesize that having the richest and most varied set of solutions will be a good start for any future criteria.

IV. CONCLUSION AND FUTURE WORK

This article presents a new methodology for evaluating immigrant strategies in MOGAs on the basis of three different dimensions: Quality, Stability, and Speed. The purpose of this methodology is twofold. On the one hand, it allows evaluating if a particular immigrant strategy is improving o hindering one of the aforementioned dimensions. On the other hand, the methodology also allows obtaining insights about what could be happening in the case that some dimensions get hindered.

To prove the effectiveness of the new methodology, a case study has been carried out on the Dynamic Community Detection (DCD) problem. In our case study, we analyze the results of the three most common immigrant strategies: RIGA, EIGA, and HIGA. In addition, two other methods were included, HGA and DEIGA, in order to set an actual comparison basis.

The experimental analysis has shown that inserting random immigrants into an already established population produces low quality offspring that do not survive. In practice, for this particular case study, using random immigrants is equivalent to reducing the population size and it is better to directly re-use a previous population. As a consequence, RIGA and HIGA have a low numerical performance, even when compared with HGA. On the contrary, the study shows that using elite immigrants achieves promising results and produces benefits over the direct re-utilization of previous populations. Hence, the better performance of EIGA over all other methods, including the comparison basis method HGA. In addition, the experimental analysis has shown that EIGA reaches an equilibrium between both immigrant and native solutions in the population, suggesting that a multi-population genetic algorithm would be beneficial for this particular strategy on this particular problem. However, it is important to note that the results extracted from the case study should be considered preliminary, as they were only tested using synthetic networks. More robust results require benchmarks composed of real networks, which, at the time of writing, are difficult to find. However, despite this limitation, the results of the case study should be considered robust when assessing the reliability of the methodology, which is the main objective of the article. Therefore, after the above analysis, we can conclude that the proposed methodology has allowed systematizing the analysis of the different immigrant techniques and that the population mixture visualization technique provides valuable insight into their inner workings. Finally, with the above in mind, we can declare that our methodology could ease the successful application of immigrant techniques to MOGAs.

However, three areas that deserve further study are identified in this work. From the dynamics of the interactions between native and immigrant solutions found in EIGA, a first issue is to design a multi-population multi-objective genetic algorithm for the Dynamic Community Detection problem. This algorithm would have two populations, one with the native solutions and one with the immigrants, that work independently but periodically exchange solutions

between the populations. Alternatively, it would also be interesting to study the DCD problem from the perspective of an Evolutionary Multi-Task Optimisation (EMTO). In EMTO, instead of solving each task one at a time, all tasks are solved at the same time using the knowledge gained in one task to help the others. In particular, is of special interest the MFEA [56] algorithm in which all tasks (snapshots in DCD) are optimized at the same time by a single population. Secondly, and since the DCD problem is a discrete problem, we aim to extend this type of analysis to more complex dynamic problems like continuous problems or tackling other dynamic multi-objective problems to completely asses the evidence of the benefits of the analysis methodology proposed in this article. Finally, since the proposed methodology allows systematizing the analysis in the case of dynamic multi-objective problems, we found interesting to adapt it first to dynamic mono-objective problems, and later to other related problems like EMTO.

REFERENCES

[1] G. Ramalingam and T. Reps, "On the computational complexity of dynamic graph problems," *Theor. Comput. Sci.*, vol. 158, nos. 1–2, pp. 233–277, May 1996.

[2] J. Branke and H. Schmeck, "Designing evolutionary algorithms for dynamic optimization problems," in *Advances in Evolutionary Computing*. Springer, 2003, pp. 239–262.

[3] M. Farina, K. Deb, and P. Amato, "Dynamic multiobjective optimization problems: Test cases, approximations, and applications," *IEEE Trans. Evol. Comput.*, vol. 8, no. 5, pp. 425–442, Oct. 2004.

[4] X. Yu, Y. Jin, K. Tang, and X. Yao, "Robust optimization over time—A new perspective on dynamic optimization problems," in *Proc. IEEE Congr. Evol. Comput.*, Jul. 2010, pp. 1–6.

[5] S. Yang, "Evolutionary computation for dynamic optimization problems," in *Proc. 15th Annu. Conf. Companion Genetic Evol. Comput.*, Jul. 2013, pp. 667–682.

[6] T. T. Nguyen, S. Yang, and J. Branke, "Evolutionary dynamic optimization: A survey of the state of the art," *Swarm Evol. Comput.*, vol. 6, pp. 1–24, Oct. 2012.

[7] K. C. Tan, L. Feng, and M. Jiang, "Evolutionary transfer optimization–A new frontier in evolutionary computation research," *IEEE Comput. Intell. Mag.*, vol. 16, no. 1, pp. 22–33, Feb. 2021.

[8] T. Wei, S. Wang, J. Zhong, D. Liu, and J. Zhang, "A review on evolutionary multi-task optimization: Trends and challenges," *IEEE Trans. Evol. Comput.*, early access, Dec. 30, 2021, doi: 10.1109/TEVC.2021.3139437.

[9] R. Azzouz, S. Bechikh, and L. B. Said, "Dynamic multi-objective optimization using evolutionary algorithms: A survey," in *Recent Advances in Evolutionary Multi-Objective Optimization*. Springer, 2017, pp. 31–70.

[10] H. G. Cobb and J. J. Grefenstette, "Genetic algorithms for tracking changing environments," Naval Res. Lab, Washington, DC, USA, Tech. Rep. ADA294075, 1993. [Online]. Available: https://apps.dtic.mil/sti/citations/ADA294075

[11] A. Simões and E. Costa, "An immune system-based genetic algorithm to deal with dynamic environments: Diversity and memory," in *Artificial Neural Nets and Genetic Algorithms*. Springer, 2003, pp. 168–174.

[12] K. Kim, R. McKay, and B.-R. Moon, "Multiobjective evolutionary algorithms for dynamic social network clustering," in *Proc. 12th Annu. Conf. Genetic Evol. Comput. (GECCO)*, 2010, pp. 1179–1186.

[13] S. Yang, "Memory-based immigrants for genetic algorithms in dynamic environments," in *Proc. Conf. Genetic Evol. Comput. (GECCO)*, 2005, pp. 1115–1122.

[14] S. Yang, "Genetic algorithms with memory-and elitism-based immigrants in dynamic environments," *Evol. Comput.*, vol. 16, no. 3, pp. 385–416, Sep. 2008.

[15] H. Cheng and S. Yang, "Genetic algorithms with immigrants schemes for dynamic multicast problems in mobile ad hoc networks," *Eng. Appl. Artif. Intell.*, vol. 23, no. 5, pp. 806–819, Aug. 2010.

[16] X. Yu, K. Tang, T. Chen, and X. Yao, "Empirical analysis of evolutionary algorithms with immigrants schemes for dynamic optimization," *Memetic Comput.*, vol. 1, no. 1, pp. 3–24, Mar. 2009.

[17] R. Tinós and S. Yang, "A self-organizing random immigrants genetic algorithm for dynamic optimization problems," *Genetic Program. Evolvable Mach.*, vol. 8, no. 3, pp. 255–286, Aug. 2007.

[18] S. Yang, "Genetic algorithms with elitism-based immigrants for changing optimization problems," in *Proc. Workshops Appl. Evol. Comput.* Springer, 2007, pp. 627–636.

[19] S. Yang and R. Tinós, "A hybrid immigrants scheme for genetic algorithms in dynamic environments," *Int. J. Autom. Comput.*, vol. 4, no. 3, pp. 243–254, Jul. 2007.

[20] E. Alba and B. Dorronsoro, *Cellular Genetic Algorithms*, vol. 42. Springer, 2009.

[21] M. Giacobini, M. Tomassini, A. G. Tettamanzi, and E. Alba, "Selection intensity in cellular evolutionary algorithms for regular lattices," *IEEE Trans. Evol. Comput.*, vol. 9, no. 5, pp. 489–505, 2005.

[22] M. Pedemonte, F. Luna, and E. Alba, "A theoretical and empirical study of the trajectories of solutions on the grid of systolic genetic search," *Inf. Sci.*, vols. 445–446, pp. 97–117, Jun. 2018.

[23] R. W. Morrison, "Performance measurement in dynamic environments," in *Proc. GECCO Workshop Evol. Algorithms Dyn. Optim. Problems*, 2003, pp. 5–8.

[24] K. Trojanowski and Z. Michalewicz, "Evolutionary algorithms for non-stationary environments," in *Proc. 8th Workshop: Intell. Inf. Syst.*, 1999, pp. 229–240.

[25] K. Weicker, "Performance measures for dynamic environments," in *Proc. Int. Conf. Parallel Problem Solving Nature*. Springer, 2002, pp. 64–73.

[26] M. Mavrovouniotis and S. Yang, "Genetic algorithms with adaptive immigrants for dynamic environments," in *Proc. IEEE Congr. Evol. Comput.*, Jun. 2013, pp. 2130–2137.

[27] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach," *IEEE Trans. Evol. Comput.*, vol. 3, no. 4, pp. 257–271, Nov. 1999.

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

[29] M. López-Ibáñez, L. Paquete, and T. Stützle, "Exploratory analysis of stochastic local search algorithms in biobjective optimization," in *Experimental Methods for the Analysis of Optimization Algorithms*. Springer, 2010, pp. 209–222.

[30] A. Panizo-LLedot, G. Bello-Orgaz, and D. Camacho, "A multi-objective genetic algorithm for detecting dynamic communities using a local search driven immigrant's scheme," *Future Gener. Comput. Syst.*, vol. 110, pp. 960–975, Sep. 2020.

[31] A. Clauset, M. E. Newman, and C. Moore, "Finding community structure in very large networks," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 70, no. 6, 2004, Art. no. 066111.

[32] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *J. Stat. Mech.: Theory Exp.*, vol. 2008, no. 10, Oct. 2008, Art. no. P10008.

[33] V. A. Traag, L. Waltman, and N. J. Van Eck, "From Louvain to Leiden: Guaranteeing well-connected communities," *Sci. Rep.*, vol. 9, no. 1, p. 5233, 2019.

[34] M. Coscia, G. Rossetti, F. Giannotti, and D. Pedreschi, "DEMON: A local-first discovery method for overlapping communities," in *Proc. 18th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2012, pp. 615–623.

[35] E. Osaba, J. Del Ser, D. Camacho, M. N. Bilbao, and X.-S. Yang, "Community detection in networks using bio-inspired optimization: Latest developments, new results and perspectives with a selection of recent metaheuristics," *Appl. Soft Comput.*, vol. 87, Feb. 2020, Art. no. 106010.

[36] G. Bello-Orgaz and D. Camacho, "Evolutionary clustering algorithm for community detection using graph-based information," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jul. 2014, pp. 930–937.

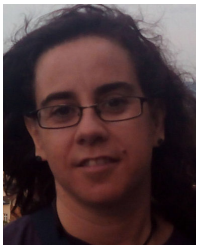
[37] G. Bello-Orgaz, S. Salcedo-Sanz, and D. Camacho, "A multi-objective genetic algorithm for overlapping community detection based on edge encoding," *Inf. Sci.* vol. 462, pp. 290–314, Sep. 2018.

[38] M. Pedemonte, A. Panizo-LLedot, G. Bello-Orgaz, and D. Camacho, "Exploring multi-objective cellular genetic algorithms in community detection problems," in *Proc. Int. Conf. Intell. Data Eng. Automated Learn.* Springer, 2020, pp. 223–235.

[39] D. Camacho, Á. Panizo-Lledot, G. Bello-Orgaz, A. Gonzalez-Pardo, and E. Cambria, "The four dimensions of social network analysis: An overview of research methods, applications, and software tools," *Inf. Fusion*, vol. 63, pp. 88–120, Nov. 2020.
[40] D. Chakrabarti, R. Kumar, and A. Tomkins, "Evolutionary clustering," in *Proc. 12th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2006, pp. 554–560.
[41] F. Folino and C. Pizzuti, "An evolutionary multiobjective approach for community discovery in dynamic networks," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 8, pp. 1838–1852, Aug. 2014.
[42] G. Chen, Y. Wang, and J. Wei, "A new multiobjective evolutionary algorithm for community detection in dynamic complex networks," *Math. Problems Eng.*, vol. 2013, May 2013, Art. no. 161670.
[43] B. A. Attea and H. S. Khoder, "A new multi-objective evolutionary framework for community mining in dynamic social networks," *Swarm Evol. Comput.*, vol. 31, pp. 90–109, Dec. 2016.
[44] J. Ma, J. Liu, W. Ma, M. Gong, and L. Jiao, "Decomposition-based multiobjective evolutionary algorithm for community detection in dynamic social networks," *Sci. World J.*, vol. 2014, pp. 1–22, 2014.
[45] K. Deb, U. B. Rao, and S. Karthik, "Dynamic multi-objective optimization and decision-making using modified NSGA-II: A case study on hydro-thermal power scheduling," in *Proc. Int. Conf. Evol. Multi-Criterion Optim.* Springer, 2007, pp. 803–817.
[46] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Trans. Evol. Comput.*, vol. 11, no. 6, pp. 712–731, Dec. 2007.
[47] Y. Yin, Y. Zhao, H. Li, and X. Dong, "Multi-objective evolutionary clustering for large-scale dynamic community detection," *Inf. Sci.*, vol. 549, pp. 269–287, Mar. 2021.
[48] X. Zhou, X. Zhao, and Y. Liu, "A multiobjective discrete bat algorithm for community detection in dynamic networks," *Int. J. Speech Technol.*, vol. 48, no. 9, pp. 3081–3093, Sep. 2018.
[49] C. Shi, P. S. Yu, Z. Yan, Y. Huang, and B. Wang, "Comparison and selection of objective functions in multiobjective community detection," *Comput. Intell.*, vol. 30, no. 3, pp. 562–582, Aug. 2014.
[50] O. Benyahia, C. Largeron, B. Jeudy, and O. R. Zaïane, "DANCer: Dynamic attributed network with community structure generator," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*. Springer, 2016, pp. 41–44.
[51] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Jan. 2002.
[52] C. Pizzuti, "GA-Net: A genetic algorithm for community detection in social networks," in *Proc. Int. Conf. Parallel Problem Solving Nature*. Springer, 2008, pp. 1081–1090.
[53] G. W. Flake, S. Lawrence, and C. L. Giles, "Efficient identification of web communities," in *Proc. 6th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2000, pp. 150–160.
[54] Y. Park and M. Song, "A genetic algorithm for clustering problems," in *Proc. 3rd Annu. Conf. Genetic Program.*, 1998, pp. 568–575.
[55] J. Kelsey and J. Timmis, "Immune inspired somatic contiguous hypermutation for function optimisation," in *Proc. Genetic Evol. Comput. Conf.* Springer, 2003, pp. 207–218.
[56] A. Gupta, Y.-S. Ong, and L. Feng, "Multifactorial evolution: Toward evolutionary multitasking," *IEEE Trans. Evol. Comput.*, vol. 20, no. 3, pp. 343–357, Jun. 2016.

ANGEL PANIZO-LLEDOT received the B.Sc. degree in computer science from the Universidad Complutense de Madrid and the M.Sc. degree in artificial intelligence from the Universidad Politécnica de Madrid (UPM). He is currently an Assistant Lecturer with the Computer Systems Engineering Department, UPM; and a member of the Applied Intelligence and Data Analysis Research Group (AIDA: http://aida.etsisi.upm.es), UPM. His main research interests include clustering, graph-based algorithms (specially focus on time evolving graphs), social data analysis, and bio-inspired methods.

MARTÍN PEDEMONTE (Member, IEEE) received the Ph.D. degree in computer science from the PEDECIBA Informática, Universidad de la República. He is currently a Senior Lecturer with the Facultad de Ingeniería, Instituto de Computación, Universidad de la República (UDELAR); and the Co-Head of the Heterogeneous Computing Laboratory (HCL: http://www.fing.edu.uy/inco/grupos/gpgpu/), UDELAR. His main research interests include computational intelligence (specially evolutionary computation and swarm intelligence) and high-performance computing (specially GPGPU and heterogeneous computing). He is also a member of the Task Force on Computational Intelligence in Latin America from the IEEE Computational Intelligence Society. He received the Best Thesis Award in computer science from the PEDECIBA Informática. He has been a Program Committee Member of the Genetic and Evolutionary Computation Conference (GECCO) since 2016.



GEMA BELLO-ORGAZ received the Ph.D. degree in computer science from the Universidad Autonoma de Madrid. She is currently a Senior Lecturer with the Computer Systems Engineering Department, Universidad Politécnica de Madrid (UPM) and a member of the Applied Intelligence and Data Analysis Research Group (AIDA: http://aida.etsisi.upm.es), UPM.

DAVID CAMACHO (Senior Member, IEEE) is currently a Full Professor with the Computer Systems Engineering Department, Universidad Politécnica de Madrid (UPM). He is also the Founder and the Director of the Applied Intelligence and Data Analysis Research Group (AIDA: http://aida.etsisi.upm.es), UPM. He has published more than 300 journals, books, and conference papers. His research interests include machine learning (clustering and deep learning), computational intelligence (evolutionary computation and swarm intelligence), social network analysis, fake news, and disinformation analysis. He has participated/led more than 50 research projects (National and European: H2020, DG Justice, ISFP, and Erasmus+). He also serves as an Associate Editor for several journals, including *Information Fusion*, *Ambient Intelligence and Humanized Computing*, *Expert Systems*, and *Cognitive Computation*.