Soft Comput (2013) 17:1209–1225
DOI 10.1007/s00500-013-0995-9

FOCUS

Matching island topologies to problem structure in parallel evolutionary algorithms

Ignacio Arnaldo · Iván Contreras · David Millán-Ruiz · J. Ignacio Hidalgo · Natalio Krasnogor

Published online: 6 March 2013

Abstract In the context of Parallel Evolutionary Algorithms, it has been shown that different population structures induce different search performances. Nevertheless, no work has shown a clear cut evidence that there is a correlation between the solver's population structure and the problem's network structure. In this work, we verify this correlation performing a clear and systematic analysis of a large set of population structures (based on the well known β-graphs and *NK*-landscape problems. Furthermore, we go beyond our findings in these idealised experiments by analysing the performance of variable-topology EAs on a dynamic real-world problem, the Multi-Skills Call Centre.

Keywords Parallel Evolutionary Algorithm · Island model · Problem structure

Communicated by E. Alba.

I. Arnaldo · I. Contreras · D. Millán-Ruiz · J. I. Hidalgo
Parallel Architectures and Bioinspired Algorithms (PABA) Research Group Universidad Complutense de Madrid, Madrid, Spain
e-mail: iarnaldo@pdi.ucm.es

I. Contreras
e-mail: ivancontreras@pas.ucm.es

D. Millán-Ruiz
e-mail: david.millan82@gmail.com

J. I. Hidalgo
e-mail: hidalgo@dacya.ucm.es

N. Krasnogor (✉)
Interdisciplinary Computing and Complex Systems (ICOS) Research Group School of Computer Science, University of Nottingham, Nottingham, UK
e-mail: Natalio.Krasnogor@nottingham.ac.uk

1 Introduction

Network-based approaches are a powerful tool for understanding the properties of complex systems (including optimisation dynamics by means of metaheuristic methods) and represent a useful data structure for capturing information of processes taking place at multiple temporal or spatial scales. The paper by Michell (1904) is widely regarded as one of the earliest industrial papers to recognise the crucial role that "topology" plays when solving a specific problem. Since then, of course, many other studies have emerged that profoundly changed our understanding of the role of networks in complex systems. Indeed, the crucial work by Watts and Strogatz (1998) on small-world (SW) networks launched the field of networks science in earnest and was followed by rapid advances by, e.g. Barthelemy and Amaral (1999), Barabási and Albert (1999), Newman et al. (2000), Wang and Chen (2003), Barrat and Weigt (2000) among others.

The small-world networks described by Watts and Strogatz (1998) raised a great deal of interest in different research areas as they postulated that, apparently different networks arising in biological, social or technological systems had, at their core, some common characteristics that helped to organise the universe of possible networks into well-defined classes with well-defined features. For example, some naturally occurring small-world networks present a high-clustering coefficient and yet a small characteristic path length that enables the rapid percolation of information across the network. Later, Barabási and Albert (1999) suggested that the distribution of highly connected vertices in networks such as the WWW or the citation of scientific publications is far from being random. In fact, in the so-called scale-free networks, vertex connectivities follow a scale-free power-law distribution, meaning that a

reduced set of vertices dominate the connectivity of the network. This feature is a consequence of two generic mechanisms: networks expand continuously by the addition of new vertices, and new vertices attach preferentially to nodes that are already well connected.

Advances in network science resonated well with evolutionary algorithm research, specifically with work on parallel and cellular evolutionary algorithms were structured populations were introduced that modified the dynamics of information exchange (e.g., through genetic recombination, solution migration policies or memetic transmission) within a population. It is currently accepted that, contrary to panmitic populations, the use of decentralised populations confers the evolutionary algorithm the opportunity for a better exploration of the search space and can improve both the numerical and runtime behaviour of the algorithm. For example, in Cantú-Paz (1999), a relation was established between the network topology, the deme size, the migration rate, and the efficacy of a given algorithm for some idealised problems. Moreover, Cantú-Paz (1999), showed that the choice of migration and replacement strategies affected the takeover time within multi-population Genetic Algorithms (e.g. choosing migrants or replacements according to fitness increases global selection pressure and causes faster convergence; in turn, shortened convergence times, although desirable, may also constitute a potential source of failure due to the premature loss of diversity).

Evolutionary algorithms that embraced a refined population structure can be roughly classified into two main families, namely, the Cellular Evolutionary Algorithm (CEA) in which genetic interactions may only take place in a small neighbourhood that is defined around each individual and Parallel Evolutionary Algorithms (PEA), also known as Distributed Evolutionary Algorithms, in which a single population is partitioned into several subpopulations or "islands" that exchange individuals according to a given migration policy. The concept of migration policy was further formalised in Alba and Tomassini (2002) as a tuple of five values indicating the migration rate, the frequency of migration, the policy for selecting migrants, the replacement policy, and whether or not the migration is synchronous. The authors showed that the balance between exploration and exploitation and hence, the probability of success of a given algorithm, was directly affected by the migration policy.

In Giacobini et al. (2006), the properties of CEAs with populations structured as Watts–Strogatz small-world graphs and Albert–Barabási scale-free graphs were investigated using as benchmarks problems of different difficulty. Their results showed that small-world topologies allow for a trade-off between robustness and speed of the search. In terms of success rate, these topologies behaved often better than the panmitic case but with slower convergence rates. On the other hand, scale-free topologies did not seem to be appropriate for the given benchmarks, probably due to premature convergence problems.

Lattice topologies were also explored for memetic algorithms where a cellular memetic algorithm was used to successfully solve a range of standard continuous optimisation benchmark problems (Quang et al. 2009) while Woolley et al. (2011) tailored the approach for a real-world optimisation of parameters for scanning probe microscopy.

The work presented in Wang et al. (2011) provides a systematic study of the performance of a PEA under a number of simple topologies including a single population (effectively a panmitic EA), a set of distributed populations but without links between them, paired populations, two-layered lattice connections and a fully connected topology. Upon these topologies they run the 0–1 Knapsack and the Weierstrass Function minimisation problems. The results showed that the two-layered lattice connections and a fully connected topology outperform the others as the complexity of the considered problems increased.

A different approach was taken by Whitacre et al. (2008) who, rather than fixing the population topology, they allowed it to co-evolve with the solutions being sought for the target problem.

Network-centric perspectives have benefitted other metaheuristics too. For example, Kennedy and Mendes (2002) analysed the effect of different topologies on particle swarm optimisation. They showed that for PSO, some random networks achieve outstanding results while the commonly used structures (Fully Connected and Ring) correspond to sub-optimal solutions. In fact, the evolutionary search process might benefit from some topological properties of these random structures, resulting in a good balance between exploration and exploitation. The paper by Li et al. (2009) shows an extensive study of the topology space, in which the effect of 1,200 different network topologies is analysed in the context of self-assembling programs. The cited work studies a wide range of graph topologies, covering simple reticular structures, small-world networks and fully random networks. It is shown that different topologies and average interconnection distances within the network have an influence on the software self-assembly process, along with resulting complexity and diversity of the generated programs.

The above summary, albeit by necessity only partial, reflects the variety of works that have been undertaken into the amalgamation of Evolutionary Algorithms and Network Science. Notably, and notwithstanding (a) the diversity and quality of the work done in the past and (b) that is currently beyond doubt that different network structures induce different search performance, no work has shown a clear cut evidence that *there is a correlation between the solver's population structure and the problem's network structure*. It is this correlation that we seek

to verify by a set of idealised, simple and clear experiments based on the well-known β-graphs (as the solvers' population structure) and the *NK*-landscapes as the problem network structure. Furthermore, we go beyond our findings in these idealised experiments by analysing the performance of variable-topology EAs on a real-world problem.

The rest of the paper is structured as follows: Sect. 2 explains the methodology of the experiments. Experimental results are detailed on Sect. 3 together with the discussion. Section 4 concludes the paper.

2 Methodology

In this section we describe the benchmarks used to ascertain whether there are correlations between problem structures and the topologies used to interconnect a set of population islands. We describe first the problems used to benchmark the different topologies and the evolutionary algorithms employed.

2.1 Benchmarks

We have used three different problems to ascertain whether correlations exists between problem structures and population topologies. The first two types of benchmarks are idealised problems, *OneMax* and *NK*-landscapes, that allow for a precise control of the problem structure. The last benchmark is a real-world, dynamic and stochastic problem that is used to evaluate whether the findings uncovered with the idealised problems scale-up to more realistic scenarios. We describe in details each of these benchmarks next.

2.2 *OneMax* benchmark

The One-Max problem is a simple and well-known problem that consists in maximising the number of ones in a bitstring. Usually, it can be located in studies that evaluate the performance of different methods or algorithms (see Goeffon and Lardeux 2011; Fialho et al. 2008). Formally, this problem can be described as finding a string $\boldsymbol{x} = \{x_1, x_2, \ldots, x_N\}$, with $x_i \in \{0, 1\}$, that maximises the following function:

$$F(\boldsymbol{x}) = \sum_{i=1}^{N} x_i \tag{1}$$

2.3 *NK*-landscape benchmark

NK-landscapes have been widely used as a test case in the field of evolutionary search Verel et al. 2011. These problems represent a rich problem domain as it is possible to obtain problem instances of adjustable difficulty from a simple model. Each instance is characterised by a number of genes N, and a number of interactions between genes K such that, $0 < N, 0 \leq K \leq N - 1$. The epistatic interactions or neighbourhood of a given bit i can be chosen at random or in a regular manner, selecting the K nearest bits to i. The result is a $N \times 2^{K+1}$ matrix E that represents the epistatic interactions of K bits. A solution S to the problem is a binary string of length N, and its fitness is computed as follows:

$$\text{Fitness}\,(S) = \frac{\sum_{i=1}^{N} f_i(S_i, S_{i_1}, \ldots, S_{i_K})}{N} \tag{2}$$

where $f_i(.)$ is an entry into E, S_i the value of string S at position i and S_{i_j} is the value of string S at the jth neighbour of bit i.

It has been shown that the neighbourhood structure determines the complexity of the problem. In fact, if the structure used is that of adjacent neighbours then the problem can be solved in polynomial time. On the other hand, if the neighbours are chosen at random the problem can be NP-Hard (Weinberger 1996). Furthermore, the landscapes can be tuned from smooth to rugged by increasing the value of the parameter K. Krasnogor and Gustafson (2004) and Krasnogor (2004) showed that it was possible to evolve local searchers for Memetic Algorithms that would match the structure of the problem being solved, in particular, *NK*-landscape instances. In this paper we would like to evaluate whether one could match different island topologies to different instances of the *NK*-landscape. Thus, we study four different scenarios: low epistasis and poly-time solvable, high epistasis and poly-time solvable, low epistasis and NP-hard and high epistasis and NP-hard instances of the *NK*-landscape problem as done by Krasnogor and Gustafson (2004).

2.4 Systematic topologies via β-graphs for idealised problems

In order to simply and clearly assess whether different island topologies could better serve different problem structures (i.e. different N and K in the *NK*-landscape benchmark) we use a family of graph models, β-graphs, which were proposed to analyse small world phenomena (Watts and Strogatz 1998), as a systematic source of island topologies. The question Watts tries to answer can be briefly explained as: What are the most general conditions under which the elements of a large, sparsely connected network will be close to each other. The closeness of vertices is determined by the length property of the graph, which has been an active research area and has been studied on different problem classes, for example, the performance of computer networks, telecommunication

network and, more recently, we showed that these graphs topologies greatly impact the diversity of generated programs in a "GP-like" setting (Li et al. 2006, 2009).

β-Graphs capture a variety of network topologies from a highly ordered to a completely random graph. Three parameters are used to define the properties of graphs generated under the β-graph model, namely, n representing the number of vertex in the graph, k determining how many initial nearest neighbours each vertex has and—finally—β that defines the rewiring rate.

Characteristic path length (abbreviated as CPL in the remaining of this paper) is one of the most important statistics used to measure the shortest distance between each vertex (i, j) in a graph. The formal definition of CPL is given by Watts and Strogatz (1998) as "The characteristic path length (CPL) of a graph (G) is the median of the means of the shortest path lengths connecting each vertex $v \in V(G)$ to all other vertices. That is, calculate $d(v, v_j) \forall v_j \in V(G)$ and find $\overline{d_v}$ for each v. Then define L as the median of $\{\overline{d_v}\}$." Furthermore, for each graph one can also define the clustering coefficient (abbreviated as CC in the remaining of this paper) as the degree to which a vertex neighbours are also neighbours of each other: $\mathrm{CC_v} = \frac{\mathrm{E(v)}}{\binom{k_v}{2}}$, where $E(v)$ is the number of edges incident in v and k_v the maximum number of possible edges.

We have chosen *NK*-landscapes and β-graphs as, given that their internal structure bear a remarkable similarity (see Figs. 1, 2), if we *do not find correlations* between β-graphs induced island topologies and *NK*-landscapes structures, then it will be very difficult to justify that *specific island topologies* are better for *specific problem structures*. On the other hand, if a clear evidence that these two types of structures can be functionally linked, then a new research avenue for improved optimisation will be open.

In Fig. 2, for reference, we show the CPL and CC for the β-graphs we used later on our experiments.

2.5 A real-world benchmark problem: dynamic optimisation in a Multi-Skill Call Centre

The two idealised benchmarks described above, together with a systematic topology generation through β-graphs, must be complemented with a real-world problem for which, *a priori*, one has no possibility of predetermining an optimal population structure. We will use the problem

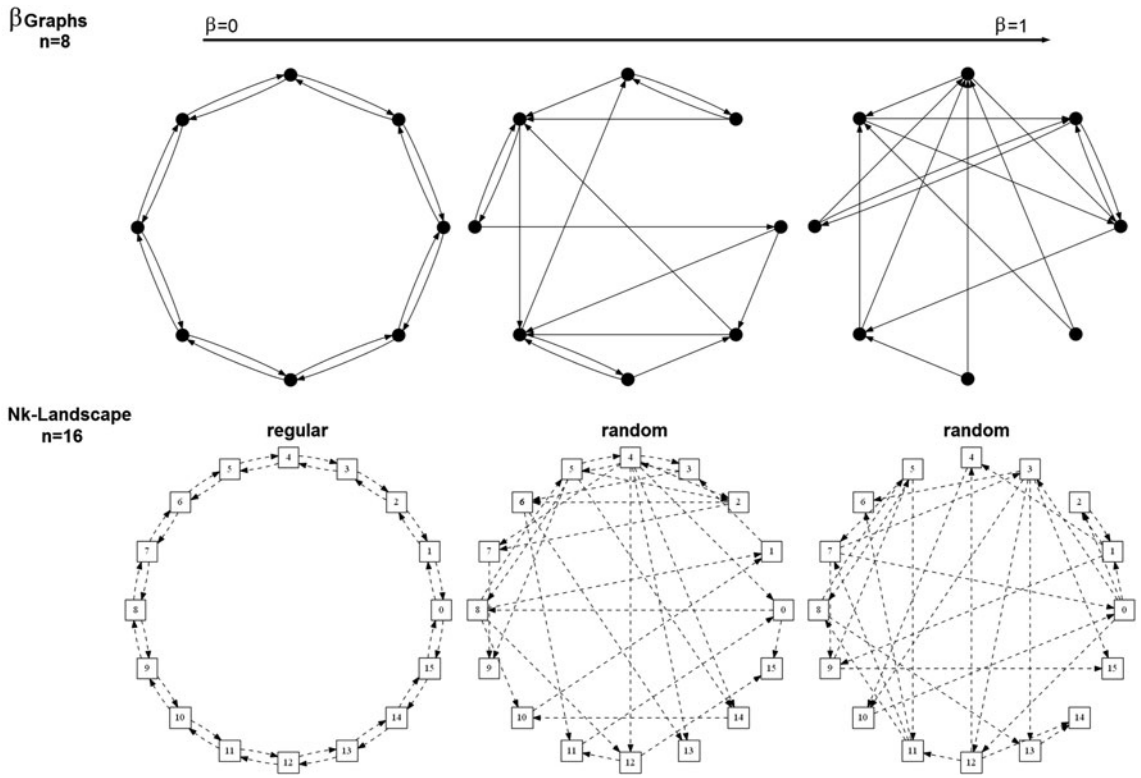

Fig. 1 Pictorial representation of *NK*-landscapes and β-graphs. The *top panel* shows three β-graphs examples with $n = 8$ and with $\beta = 0, 0.5, 1$. In the *bottom panel*, three instances of the *NK*-landscape problem are shown. All the instances have $N = 16$ and $k = 2$, with the landscape to the left having a regular nearest neighbours epistatic structure, while the other two have a random set of epistatic interactions

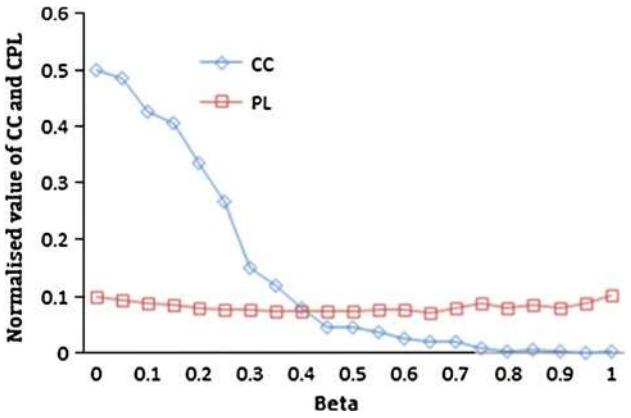

Fig. 2 Clustering coefficient (CC) and characteristic path length (CPL) as a function of β for β-graphs with $n = 16$ and $k = 4$

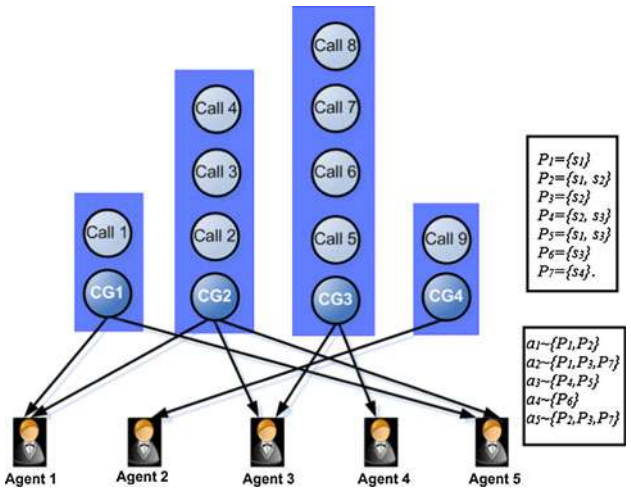

Fig. 3 Inbound traffic scheme in MSCCs

described below to investigate (later in the paper) whether it is, in principle, possible to let the optimisation algorithm (in our case a Genetic Algorithm) choose the best topology as the dynamic problem unfolds. Thus, this benchmark will be used to ascertain whether different topologies are better at different stages of the search process under a dynamic optimisation scenario.

In a Multi-Skill Call Centre (MSCC), there are n incoming customer calls $C = \{c_1, c_2, \ldots, c_n\}$ grouped in k call groups $\mathrm{CG} = \{\mathrm{cg}_1, \mathrm{cg}_2, \ldots, \mathrm{cg}_k\}$ according to the call type, and m agents $A = \{a_1, a_2, \ldots, a_m\}$ that have a subset of all the possible skills ($S = \{s_1, s_2, \ldots, s_k\}$) to attend the corresponding call groups (having the skill s_i enables you to attend the call group cg_i). Not all the agents have the same skill set and the number of skills per agent is different. Agents can only attend the call groups they have been trained for. This implies that each agent can attend different call types and, given a call type, it can be answered by several agents who have the associated skill. Note that agents cannot attend any kind of customer calls as they are usually specialised in concrete tasks (they do not have the complete skill set) or sometimes limited by the law regulations. Although agents may have multiple skills, each agent can only process one call at the same time. Furthermore, given a call, it requires an unknown amount of time to be accomplished. Besides, each agent must orderly process each call during an uninterrupted period of time; in other words, the call cannot be divided or postponed once it has been started.

Figure 3 illustrates the relationship among client calls, queues and agents. This figure describes an example for nine client calls grouped in four CGs, five agents having different real skills and seven different profiles.

The main objective of this real-world problem is to get, for each time-frame (t), an automatic allocation of agents and call groups ($\{a_i, cg_j\}_t$ when a_i is related to s_j) that maximises the service level [see Millán-Ruiz and Hidalgo (2010)]. It stands to reason that we want to devote more agents to those call groups with greater traffic volume or to those with higher priority or relevance.

The problem of workforce distribution in MSCCs is a very complex and dynamic real-world problem. Usually, the number of incoming calls (n) is much larger than the number of agents (m) and the flow of calls is very dynamic over time, making this problem really hard. Intuitively, this problem is much more complicated than having a simple pool of incoming calls where agents take work from, since it requires the assignment of customer incoming calls to the agents having the right skills, satisfying a given set of additional constraints and respecting the dependencies among individual tasks and differences in the execution skills of the agents [see Millán-Ruiz and Hidalgo (2010) for getting further information about this problem]. This problem is somehow related to other classic changing scenarios where staffing requirements are identified to insure that the organisation has the right number of agents at the right time. This is a highly difficult problem because we are not only dealing with an NP-hard problem like the job assignment problem (Brucker 2007), but the problem also considers rapidly varying conditions, massive incoming calls and a large number of agents having hard constraints to process certain tasks. Reviewing the state-of-the-art, we can find a number of strategies and algorithms to solve this problem (see Millán-Ruiz and Hidalgo 2010a, b and c) where Parallel Genetic Algorithms (PGA) have proved to be the most competitive approach.

2.6 Algorithms and experimental setup for the *OneMax* and *NK*-landscape problems

We systematically generated 20 β-graphs per topology. Note that each topology is defined with a tuple (n, k, β) in the following ranges:

$n \in [1, 2, 4, 8, 16, 20]$
$k \in [0, \ldots, N-1]$
$\beta \in [0.05, 0.10, 0.15, \ldots, 0.90, 0.95, 1.00]$.

With the above parameters and deleting the redundant combinations, we reach a total of 851 topologies. Each of these topologies were evaluated into two batches for the *NK*-landscape problems:

- *Batch 1*: $N = 50$ and $K \in [2, 4, 8, 14]$
- *Batch 2*: $N = 100$ and $K \in [2, 4, 8, 14, 28, 56]$.

In both cases, random epistatic interactions and contiguous epistatic interactions were tried out.

For the *OneMax* problem we used one instance of 5,000 bits (preliminary experiments with smaller instances were easily solvable by our GA).

Both problems were solved with a simple Parallel GA that was executed 20 times per topology. The topology of island connectivities was given by the β-graph topologies as defined above and was kept fixed throughout the evolutionary process. The main parameters of the PGA are:

1. *Initialisation* The algorithm starting population is initialised randomly.
2. *Selection* A classical binary Tournament Selection has been implemented to select the parents of the offspring. A fitness-based match is used where the selected parents survive until the next generation.
3. *Crossover* The offspring is generated by a single point crossover (SPX). The probability of crossover is 0.9;
4. *Mutation* In order to avoid another variable, no mutation was used.
5. *Migration policy* It is fixed to a simple replacement policy with a bandwith of 10 % of an island population. This means that the best individuals, 10 % of each emitting island, replaces randomly a part of the population of the receiving island (always preserving the elitism).

2.6.1 *Algorithms and experimental setup for the Multi-Skill Call Centres*

It is not always straightforward to control the internal dynamics of a PGA based on the island model, especially while seeking to ensure a fair balance between exploration and exploitation in a dynamic real-world environment. In real production environments, engineers do not always have enough time to test out and compute all the possible combinations to determine the optimal island connectivity configuration as there are many factors that may have an effect on the overall performance and accuracy (number of islands, topology, migration and replacement policies, amount of migrants, frequency of migrations, number of individuals in each island, type of synchronism, etc). This problem is even more severe when dealing with dynamic optimisation under uncertainty such as in the Multi-Skills Call Centre problem. To select the optimal configuration, we have developed a Meta-PGA that automatically determines a sufficiently competent configuration for a second, "internal", PGA is the one that actually solves the MSCC problem.

Some authors have already developed Meta-GAs in the past. Wright (1991) was one of the pioneers in using GAs to optimise problems over several real parameters. Lee and Takagi (1993) proposed an automatic fuzzy system design method that used a GA and integrated three design stages. Their method determined membership functions, the number of fuzzy rules and the rule-consequent parameters at the same time. Clune et al. (2005) used a Meta-GA to investigate the evolution of parameter settings (genetic operators) for genetic and evolutionary algorithms in the hope of creating a self-adaptive algorithm. Nannen and Eiben (2006) presented and evaluated a method for estimating the relevance and calibrating the values of the parameters of an evolutionary algorithm. The method provided an information theoretic measure on how sensitive a parameter was to the choice of its value. In Nannen and Eiben (2007), the same authors proposed an advanced method that helped to calibrate the parameters of an evolutionary algorithm in a systematic and semi-automated manner. The method for relevance estimation and value calibration of evolutionary algorithm parameters was empirically evaluated in two different ways. More recently, Brain and Addicoat (2010) made use of a Meta-GA to optimise the parameters of a simple GA through an evolutionary process. They addressed the problem of determining the electronic structure of long chain molecules. The same year, Shahsavar et al. (2011) proposed a methodology for both optimal pattern selection and tuning. They employed a robust GA to solve a project scheduling problem.

All these algorithms were focused on classical GA, but we now propose a Meta-PGA for parameter calibration that automatically tests out different islands and migration configurations. It entails a number of independently evolving populations to determine the right setting-up of an internal PGA.

The chromosome of our Meta-PGA, which has six genes, follows an integer encoding scheme. These genes refer to diverse parameters that affect the final performance of the internal PGA (see Fig. 4).

Let us present the pseudo-code before going on with the explanation (see below) .

Meta Parallel Genetic Algorithm

Generate a random population;
Evaluate the individuals' fitness by running
...the internal PGA with current configuration;
Store configurations and associated fitness to
...avoid running those configurations again in the future;
$generations \leftarrow 0$;

While($generations \leq 200$)

Select a subset of individuals;
Apply crossover;
Apply Mutation;
Check those configurations previously
...calculated and get their fitness values;
Run the internal PGA for new configurations;
$generations \leftarrow generations + 1$;

End While

Algorithm 1: Pseudocode of the Meta Parallel Genetic Algorithm

In our Meta-PGA, we encode each solution as an array of integers whose indexes represent each parameter and the array contents refer to the values that these parameters can take. These genes can take the following values:

1. *Number of islands* (from 1 to 12 populations).
2. *Topology* (star, bidirectional ring, all-to-all).
3. *Population size* (from 4 to 100 individuals per island).
4. *Migration and replacement policies* Best-Fitted Individuals by Worst-Fitted Individuals (BFI-WFI), Best-Fitted Individuals by Random Individuals (BFI-RI), Best-Fitted Individuals by Best-Fitted Individuals (BFI-BFI), Best-Fitted Individuals by Most Different Individuals (BFI-MDI), Best-Fitted Individual + "Annealing" by Worst-Fitted Individuals (BFIA-WFI).
5. *Migration frequency* (30 or 60 s).
6. *Amount of migrants* (percentage from 10 to 30 %).

The evolutionary operators of the Meta-PGA has been set up as follows:

1. *Fitness function* We measure the service level resulting from each configuration (see Millán-Ruiz and Hidalgo 2010).
2. *Population size* The population contains 20 different individuals encoded as hinted above.
3. *Initialisation* The initial population is randomly generated.
4. *Selection* We have applied a binary tournament selection to select individuals from the population.
5. *Crossover* The offspring inherits the common points in their parents and randomly receives the rest of genes from them.
6. *Mutation* We apply a perturbation over each gene of the chromosome with a probability of 0.1.

Fig. 4 Encoding of the Meta Genetic Algorithm

Now, we provide the details of the internal PGA (the one that in fact solves the problem being analysed). Its configuration is as the following:

1. *Encoding* We encode every solution as an array of integers whose indexes represent the available agents at a given instant and the array contents refer to the profile assigned to each agent.
2. *Fitness function* We measure the service level resulting from the configuration of agents and incoming calls (see Millán-Ruiz and Hidalgo 2010).
3. *Initialisation* The initial population is randomly generated.
4. *Selection* Individuals are selected, using a binary tournament mechanism.
5. *Crossover* The offspring inherits the common points in their parents and randomly receives the rest of genes from them.
6. *Mutation* We apply a perturbation over each gene of the chromosome with a probability of 0.03.
7. *Replacement policy* We consider elitism with a probability of 0.93 to replace the worst-fitted individuals of the population in next generation. And with a probability of 0.07, a worse-fitted individual may be captured. Note that our basic GA relies on a steady-state scheme.
8. *Parallel GA's operators* The PGA's parameters and evolutionary operators to play with are: number of islands, topology, population size, migration and replacement policies, migration frequency and amount of migrants.

3 Results

In this section we provide the results we have obtained. We focus first on analysing the idealised problems, *OneMax* and *NK*-landscapes, and then we provide results for the Multi-Skills Call Centre problem.

3.1 *OneMax* and *NK*-landscape

As mentioned in the previous section, we have conducted a total of 357,420 experiments, i.e. 20 experiments per each of the 851 different island topologies and each of the 21

tested problems. These experiments were organised in two batches.

3.1.1 Batch 1

Batch 1 was a preliminary set up based on the *NK*-landscape with $N = 50$ and $K \in [2, 4, 8, 14]$ in both a contiguous and random epistatic interaction structure and a 5,000 bits *OneMax* instance. For each of these problem instances and topologies, 20 different random realisations of the β-graph island model were used. The island topologies were derived from β-graphs with topologies in the range $n \in [1, 2, 4, 8, 16, 20] \times k \in [0, \ldots, N-1] \times \beta \in [0, \ldots, 1, +0.05]$).

From each run of the PGA with a given island topology, we obtained the fitness of the best individual and we averaged the fitness obtained from these 20 runs; the average fitnesses so collected were used to calculate the ranking of island topologies (i.e. from best performing to worst performing in relative terms rather than through absolute fitness values) and then assigned to a matrix of pairs made up of the *NK*-landscape instances and the β-graph island topologies where these were solved. To analyse in a concise and clear way the data collected, we performed a biclustering (Liaw 2006) of the resulting matrix (composed of all such pairs). Figure 5 shows the results, thus, obtained; please note that branches A, B and C can be "rotated" without affecting the dendogram branches lengths as to make a perfect cluster ordering the problem families from low K to high K and hence, it is possible to see that the biclustering correctly groups together the easier instances (*NK*2, *NK*4, *NK*8, *NK*14) and the harder, indeed NP-Hard for random epistatic connections, ones (*NK*2R, *NK*4R, *NK*8R, *NK*14R) while separating the large *OneMax* instance from the *NK*-landscapes ones. More importantly, it clearly highlights island topology groups that, with high confidence, perform poorly on several *NK*-landscape instances (green cells) and groups that, with high confidence, perform well on several *NK*-landscape problems. Furthermore, for the two most difficult *NK*-landscape instances, *NK*8R and *NK*14R, it is possible to see a more distinct pattern of island topologies competency than for the easier ones of this problem type for which a different green-black-red pattern appears. Notably, the green-black-red pattern for *OneMax* is tantalisingly different than for the *NK*-landscapes, in particular those easier instances, e.g. *NK*2 and *NK*4, that have only a few (2 or 4) contiguous (thus, polynomially solvable) epistatic interaction, and hence, one could have expected *OneMax* to share some of the good/bad topologies with these problem. These combined observations suggest that, as the problems become more difficult, a better "signal-to-noise ratio" could be obtained in what pertains to matching island topologies to problem structure. A similar effect was identified by Krasnogor and Gustafson (2004) and Krasnogor (2004) when evolving specific local searchers for *NK*-landscape in Memetic Algorithms (Krasnogor 2012). We thus performed an extended set of experiments with harder instances that are described next.

3.1.2 Batch 2

This experimental batch employs harder instances of the *NK*-landscape by setting $N = 100$ and utilising a larger range of K, in particular, $K \in [2, 4, 8, 14, 28, 56]$. As we did before, we compute a biclustering of the topologies versus

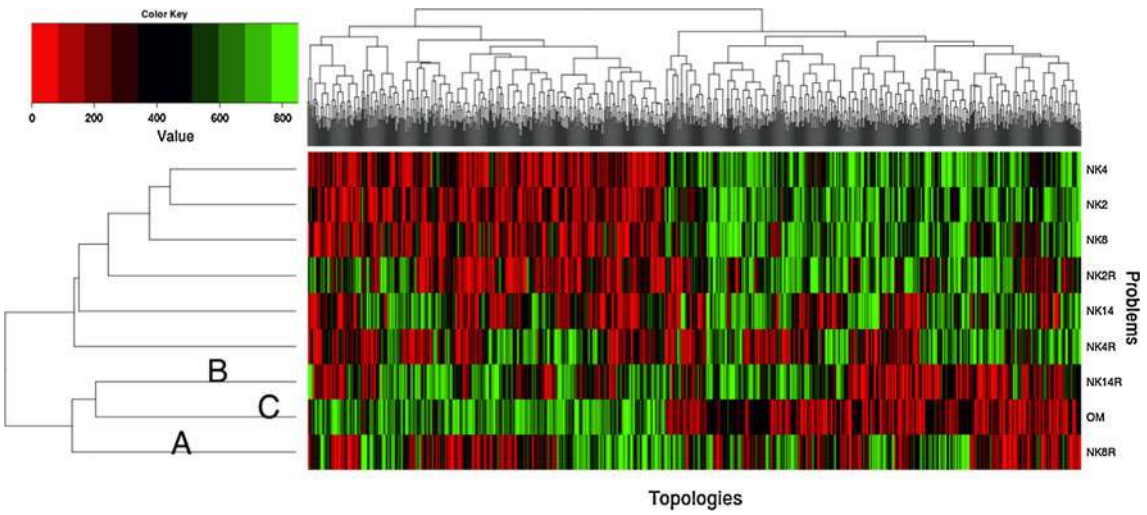

Fig. 5 *NK*-landscape problems with $N = 50$ and *OneMax* with stringsize of 5,000. We evaluate the 851 different β-graphs [tuples (n, k, β)]. For each tuple we generate 20 replicas. The average fitness for these 20 replicas is computed. All the different tuples are ranked according to their score for the different problems. A biclustering is performed with the computed ranks

the problem instances according to the ranking obtained from averaging 20 PGA runs. Figure 6 shows the results, thus, obtained.

More clearly than Figs. 5, 6 show clusters where groups of island topologies perform well (or bad) with high confidence on groups of problems. To analyse in more details which topologies are the more productive for different problems we resort to group the topologies based on their CC and CPL and re-apply the biclustering algorithm. To do this we performed the following calculations

First, we compute the CC and the CPL for each of the studied topologies. The resulting CC ranges from 0 to 1, while to compare the CPL for graphs of different sizes, we normalise CPL with respect to the maximum possible path

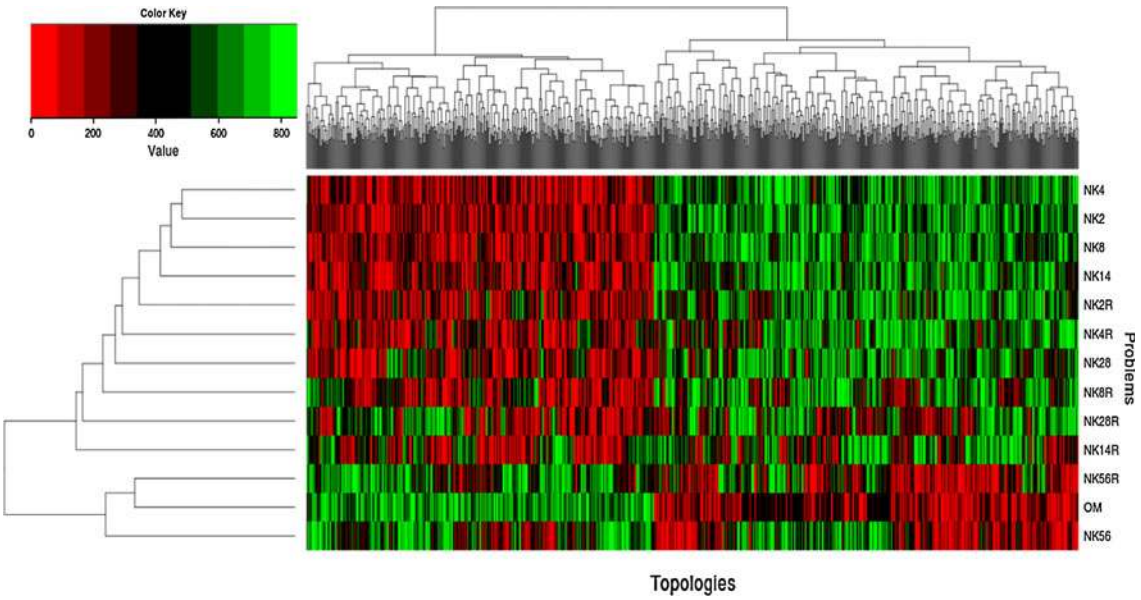

Fig. 6 *NK*-landscape problems with $N = 100$ and *OneMax* of size 5,000. We evaluate the 851 different β-graphs. For each tuple we generate 20 replicas. The average fitness for these 20 replicas is computed. All the different tuples are ranked according to their score for the different problems. A biclustering is performed with the computed ranks

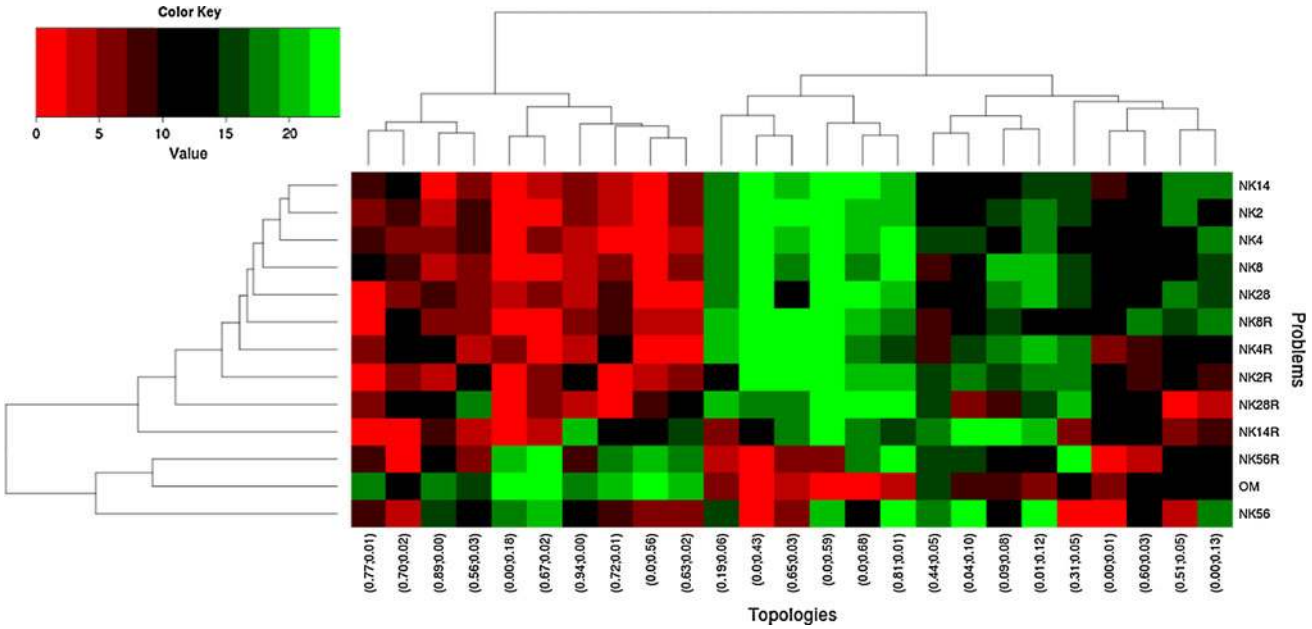

Fig. 7 The 851 β-graph topologies are distributed into 25 bins accordingly to their CC. For each bin, the average fitness was computed and then the bins were ranked accordingly to this average. The labels in the x-axis have the format $(\overline{\text{CC}}, \overline{\text{CPL}})$

length, namely $n - 1$. The normalised CPL ranges between 0 and 1; a CPL = 0 means that the given graph is a complete one, while a value of 1 means that the CPL = #*nodes*. We then "bin" all island topologies according to the (1) CC, (2) CPL, and a joint (3) CC and CPL weighted average. To decide which island topologies belong to a given bin we compute their euclidean distance based on the CC and CPL of the compared topologies plus a penalisation for dissimilar values. The distance between two β-Graphs g_1 and g_2 is computed as follows:

$$D(g_1, g_2) = \sqrt{|\mathrm{CC}_{g_2} - \mathrm{CC}_{g_1}|^2 + |\mathrm{CPL}_{g_2} - \mathrm{CPL}_{g_1}|^2} + \max\{|\mathrm{CC}_{g_2} - \mathrm{CC}_{g_1}|, |\mathrm{CPL}_{g_2} - \mathrm{CPL}_{g_1}|\} \quad (3)$$

Each topology bin is then characterised with the average clustering coefficient ($\overline{\mathrm{CC}}$) and average characteristic path length ($\overline{\mathrm{CPL}}$) of the island connectivities it contains. As an example, these values are shown for the obtained bins in the case of joint CC and CPL clustering in Table 1.

Table 1 Shows the average clustering coefficient and the characteristic path lenght of the 25 considered bins in the case of joint CC and CPL clustering

Bin	CC	CPL
0	0.9445	0.0031
1	0.8986	0.0056
2	0.8381	0.0091
3	0.7932	0.0117
4	0.7491	0.0145
5	0.7014	0.0179
6	0.6618	0.0203
7	0.7203	0.0236
8	0.6638	0.0244
9	0.6022	0.0277
10	0.4439	0.0309
11	0.4598	0.0343
12	0.4183	0.0391
13	0.2671	0.0451
14	0.4727	0.0491
15	0.2037	0.0577
16	0.2902	0.0710
17	0.1733	0.0821
18	0.0914	0.1143
19	0.0548	0.1446
20	0.0227	0.2100
21	0.0014	0.2833
22	0.0005	0.4937
23	0.0000	0.7160
24	0.0000	0.8301

In Fig. 7 we can see the bicluster obtained using bins based on CC with clearly defined bins of high performing island topologies for certain problem structures. Similarly, Fig. 8 shows well-performing bins once these are clustered based on CPL. More specifically, from Fig. 7 we can see that island topologies with $\overline{\mathrm{CC}} \leq 0.63$ perform, generally, poorly on most problems with the exception of *NK*56, *NK*56R and *OneMax*. Good performing bins for most problems (except of *NK*56, *NK*56R and *OneMax*) have either a low $\overline{\mathrm{CC}}$ (=0.0) or a relatively high $\overline{\mathrm{CC}}$ if it is accompanied by a very low $\overline{\mathrm{CPL}}$ (≤ 0.05).

Looking at Fig. 8 we can observe that island topologies with $0.10 \leq \overline{\mathrm{CPL}} \leq 0.18$ produce good results for *NK*-landscapes with up to K = 28. To better correlate both topological measures simultaneously, we have computed a biclustering using Eq (3). This is shown in Fig. 9 where the top dendogram for the island topologies has clearly identified three distinct regions, to the left those topologies that are, mostly, poorly performing (except for the most difficult problems), in the centre the topological families that perform well up to and including problem *NK*14R and, to the right, topologies for which there is no statistical significance to their rankings (mostly in black).

3.2 Multi-Skill Call Centres

Up till now, we have presented the results of the two idealised problems which illustrate the relationship between island topologies and problem structure. For the real-world problem, we have created a predefined set of representative topology structures according to the findings unveiled by the two aforementioned idealised problems. The purpose is to delimit the topology space to analyse how those discoveries uncovered by the idealised problems scale-up to a real-world problem and study how other parameters may have an impact on the relationship between topologies and problem structure.

We now describe the problem instance of medium difficulty that we have created to test out our Meta-PGA in a real environment. This problem instance is composed by real data taken from an MSCC during a common day. The size of the snapshot where each configuration has been executed is 300 s (5 min). Note that around 800 incoming calls (n) simultaneously arrive during a normal day in such a time interval. The number of agents (m), for each time interval, oscillates around 700, having 16 different skills for each agent on average, grouped in skill profiles (sets of skills) of 7 skills on average. The total number of call types considered for this study is 167.

For a fair comparison, every configuration has been run over the problem instance 30 times. Figure 10 presents some numbers/figures to realise the magnitude and

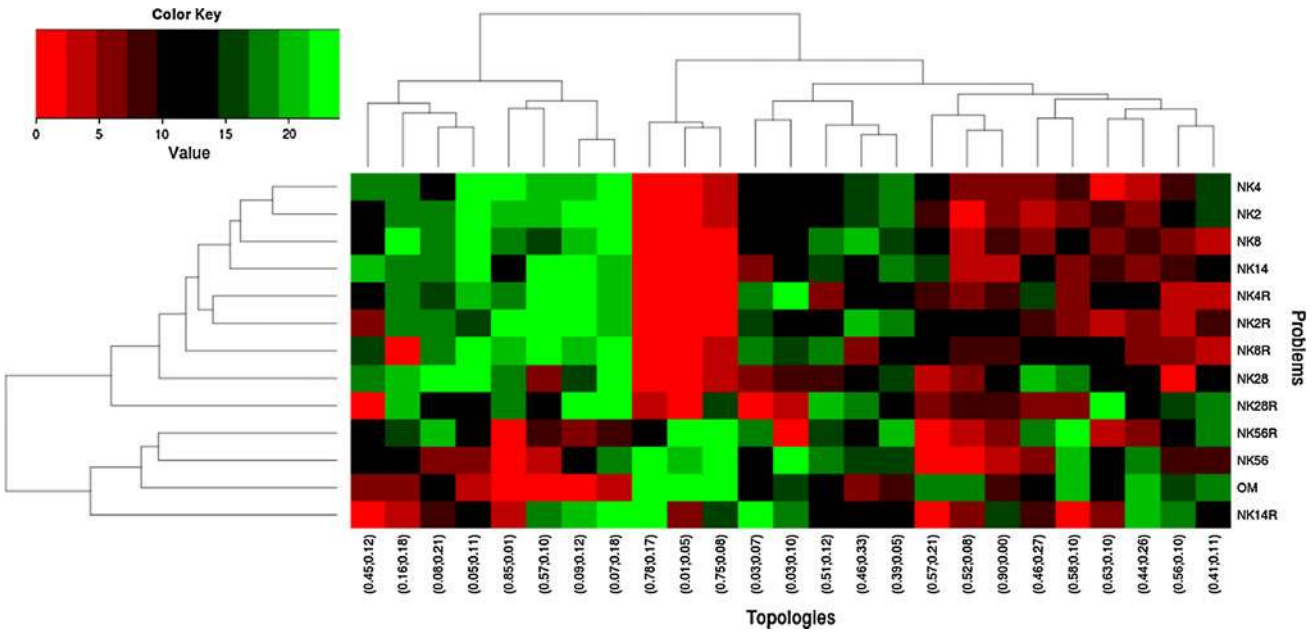

Fig. 8 We distribute the 851 families in 25 bins according to their characteristic path length. The average fitness for each bin is computed. For each of the problems, the bins are ranked according to the fitness. A biclustering is performed with these ranks. The labels in the x-axis have the format $(\overline{\text{CC}}, \overline{\text{CPL}})$

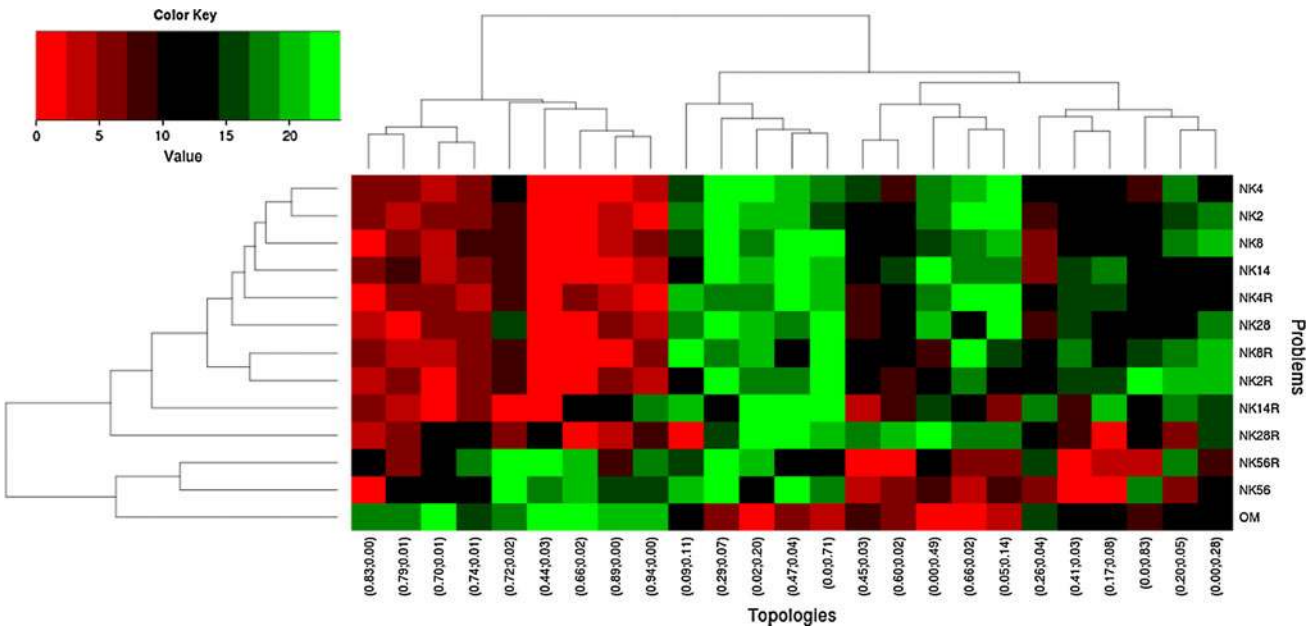

Fig. 9 We distribute the 851 families in 25 bins according to their clustering coefficient and the characteristic path length. The average fitness for each bin is computed. For each of the problems, the bins are ranked according to the fitness. A biclustering is performed with these ranks

dynamism of the MSCC being studied. This figure shows the number of incoming calls at different levels of granularity (on a monthly, daily, hourly and minutely basis).

Given the values for the 6 genes of the Meta-PGAs chromosome, there are 6,480 possible combinations ($8 \times 3 \times 9 \times 5 \times 2 \times 3 = 6{,}480$). This may seem an easy search space but every evaluation takes time, as we have to re-execute the internal PGA each time, which is unfeasible in a production environment that requires fast adaptations. Of course, we can optimise this by avoiding recalculations previously made by the Meta-PGA. The challenge should now be to develop a fast and effective

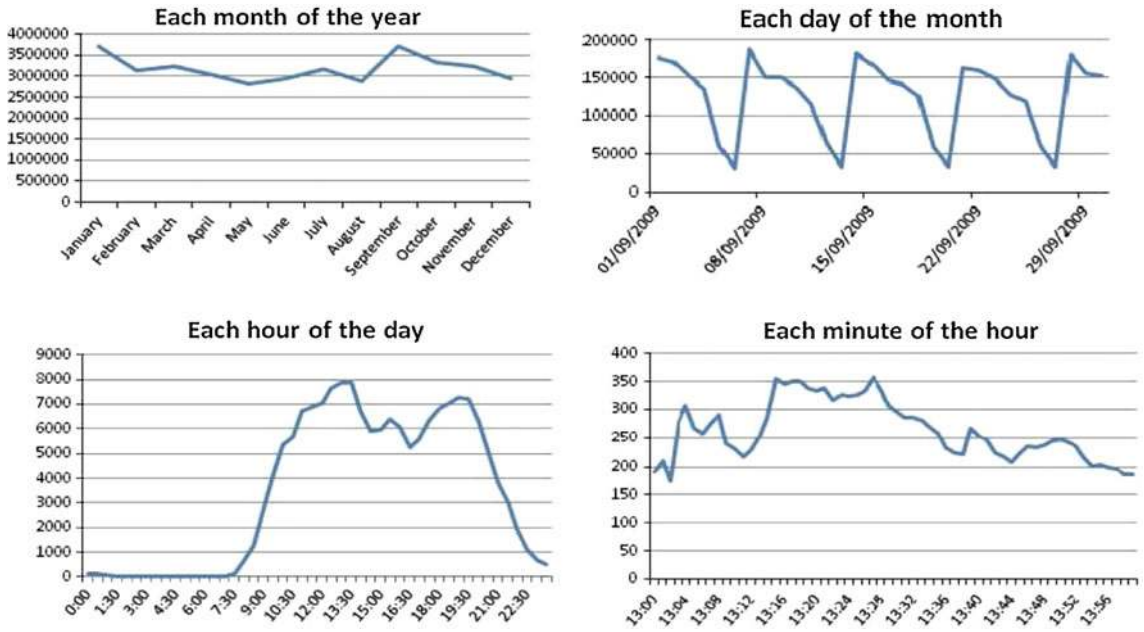

Fig. 10 Dynamism of the MSCC being analysed at different levels of granularity. *Y*-axis represents the number of incoming calls and the *x*-axis denotes the time in terms of months, days, hours and minutes, respectively. We see that, at minute level, the variability of the system increases importantly

Meta-PGA that avoids performing too many iterations to find the right configuration or, at least, a good enough approximation (see the Meta-PGA previously described).

In Fig. 11, we can see that the best-fitted individual in the population of the Meta-PGA evolves very quickly. We can even find the optimal configuration around generation-175. Best configuration found has been: eight populations, bidirectional ring, 30 individuals per population, BFIA-WFI scheme, migrations each 60 s, 20 % of migrants.

In Fig. 12, we can observe that, in few iterations, we can find a set of good candidates/individuals as the mean fitness of the population on each generation is quite high in <30 generations. This reflects that our Meta-PGA does not only

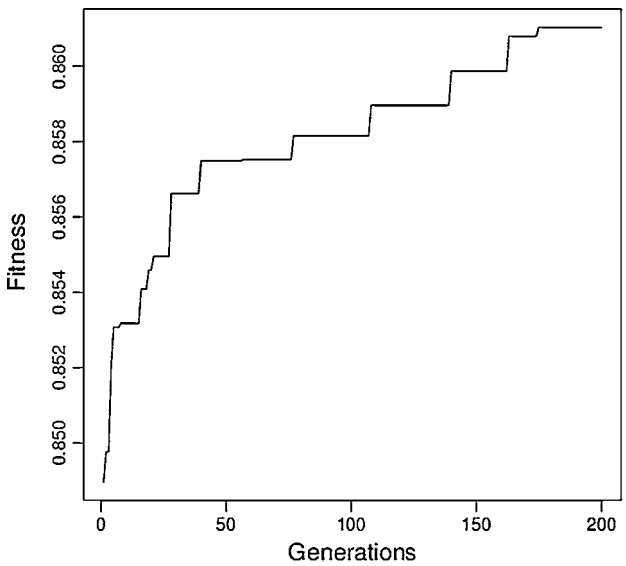

Fig. 11 Fitness value of the best-fitted individual in the population of the Meta-PGA generation-by-generation. We perceive that there is continuous evolution, especially at the beginning, and the fitness value reaches appealing levels

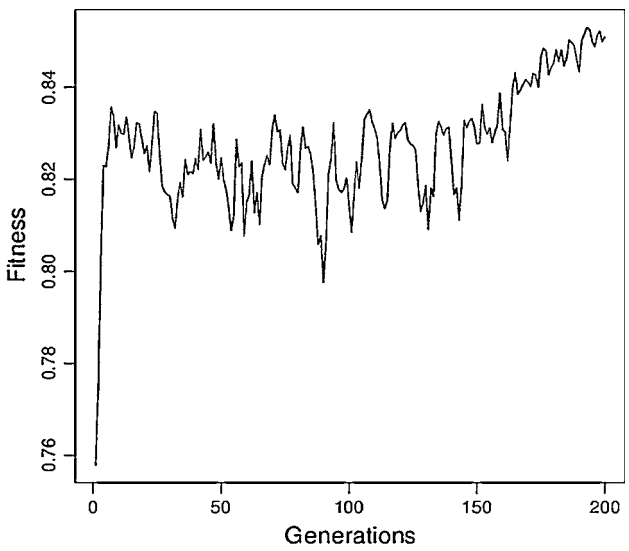

Fig. 12 We show, for each generation, the mean fitness value of the individuals that compose the population of the Meta-PGA. It reflects the mean quality of the individuals as the algorithm evolves

provide a single good solution, but also multiple high-quality candidate solutions.

Our Meta-PGA also clearly outperforms its panmictic version as Fig. 13 demonstrates. One of the main reasons of this good performance lies in the capability of evolving the topology and the migration and replacement policies when needed. The topology has an imperative impact on the migration and replacement policies, since we can perceive a variation on the migration policies when the topology evolves (e.g., in generation-60).

In Fig. 14, we show the number of individuals having each topology in their genes, generation-by-generation. As there are 20 individuals in the population of our Meta-PGA, the sum-up of the three curves is always 20.

Bidirectional ring outperforms other more connected topologies, especially when the number of islands increases. When this happens, the population quickly converges towards the same solution. Therefore, bidirectional ring seems to be the most appropriate topology for dynamic environments, most likely because this topology allows for opportune convergence, while preserving the desired diversity. It is important to highlight that, for this problem, it is crucial to have a connected topology rather than several isolated islands working in parallel (this problem requires a collaborative scheme).

The star topology also entails high-quality outcomes but quickly suffers premature convergence. The reason is that the master island receives many migrants from the subordinate islands after some migrations (and it is even worse when there are many subordinate islands), implying that populations eventually become very similar. This intuitively involves a lack of diversity so that the gain of fitness gets fatally damaged. This phenomenon affects much more strongly to the hub topology as, being all the islands interconnected to each other, the diversity diminishes too much after one or two migrations.

The two previous paragraphs confirm the results of the previous section, reflecting that each problem structure needs a different island topology configuration. Dynamic, complex problems should be supported by medium-connected topologies like the bidirectional ring to make the PGA evolve properly.

In Fig. 15, we show the number of individuals having each combination of migration-replacement policies in their genes generation-by-generation. As there are 20 individuals in the population of our Meta-PGA, the sum-up of the three curves is consequently 20.

The migration and replacement of individuals is another important feature to set-up. In this manner, replacing the worst-fitted individuals in the receiving population by the best-fitted individuals of the source population does not always behave better than taking the most different individuals. The process of analysing differences in the chromosomes in contrast implies that the internal PGA can run

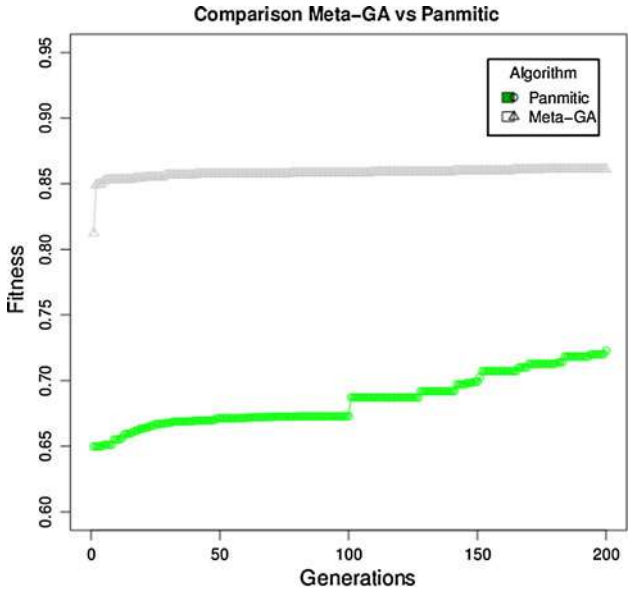

Fig. 13 Fitness-based comparison between the panmictic algorithm and our Meta-PGA. This figure shows the uplift of our Meta-PGA as compared to the panmictic version of our PGA

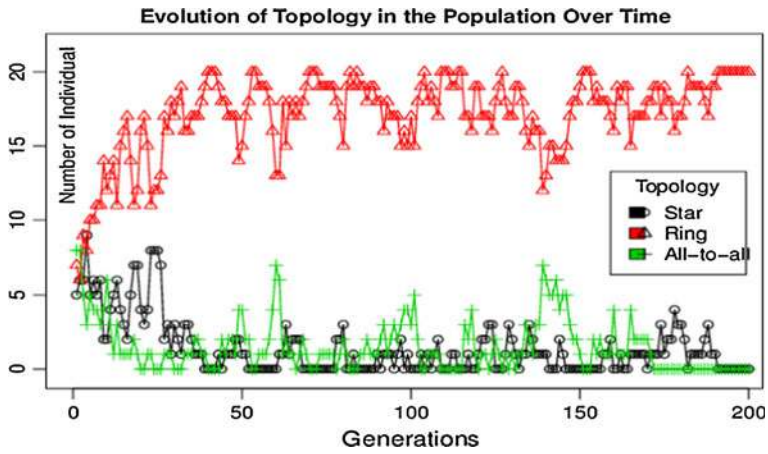

Fig. 14 Evolution of topology in the population over time. We can see the number of individuals having each topology in their genes generation-by-generation. As there are 20 individuals in the population of the Meta-PGA, the sum-up of the three curves is always 20

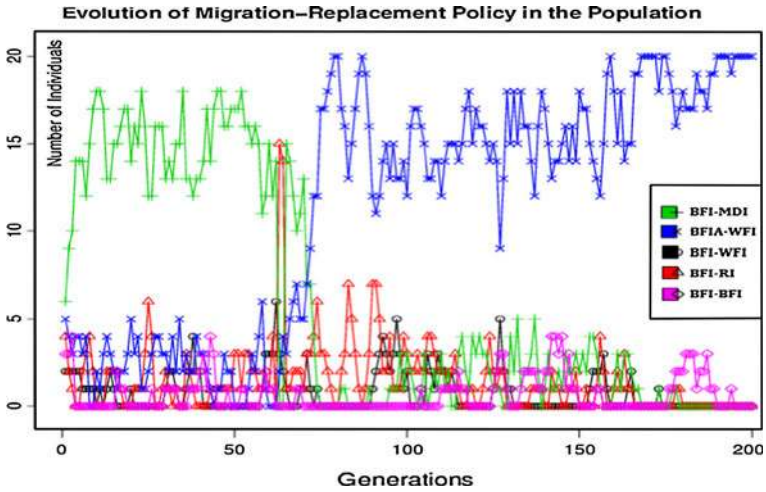

Fig. 15 Evolution of migration-replacement policy in the population over time. We can see the number of individuals having each combination of migration-replacement policies in their genes generation-by-generation. As there are 20 individuals in the population of the Meta-PGA, the sum-up of the three curves is consequently 20

fewer generations (as it is a costly operation), but entails better fitness values in the end. The underlying principle may be that fitness-based comparisons can occasionally be misleading or deceptive, leading to the situation in which two close individuals in terms of genes in common may have associated very different fitness values, whereas two far chromosomes in terms of genes in common may have assigned close fitness values. Another consequence of measuring gene differences as compared to gauging fitness values is that the lift of the fitness curve has a smoother slope in the first generations. Naturally, replacing the best-fitted individuals of the receiving population by the best-fitted ones of the source population implies a slower convergence in each processing node as we will find a larger percentage of less-fitted individuals. This way, the best migration policy has been sending the best fitted-individual with some non-necessarily best-fitted individuals (annealing set) as it provides diversity.

Another finding has been that having many individuals on each population makes the algorithm slower and fewer generations are executed. Best values seem to range from 15 to 30 individuals per population.

The migration frequency is also important in the performance. Migrations should not be done with too much frequency, each population needs to evolve separately enough time. Of course, the amount of migrants should not be very big as the internal PGA may converge very fast to the same solutions. The impact is higher when the number of islands is rather large.

We have seen that PGAs can also deal with complex, real-world application domains although they require specific tuning, depending on the nature of the problem being faced. This way, we have presented a Meta-PGA for fine-tuning PGAs based on the island model. Thanks to the results uncovered by the exhaustive study of the idealised problems, we have been able to effectively delimit the island connectivity to scale-up those findings to a very complex, real-world problem.

3.3 Discussion

The experiments performed on the *OneMax* and *NK*-landscapes using a series of β-graph based topologies for configuring the island connectivity of the parallel GA have clearly identified a direct correlation between problem structure and islands connectivity structures as evidenced by the various biclustering analysis we performed. The results discovered through the idealised problems have helped us to delimit the search space to scale-up those findings to the real-world problem. The resulting correlations are neither linear nor trivial (given the complex nature of evolutionary search) even for these kind of "perfectly known" problems and the situation is exacerbated with the Multi-Skill Call Centre Problem, a real-world scenario.

To further analyse the ranking obtained by solvers using different topologies operating on the *OneMax* and *NK*-landscapes, we have used the clusters of topologies (bins as defined in the previous sections, see Table 1) obtained previously and ranked them accordingly to how many other clusters are dominated by solvers within a given bin. A bin/cluster dominates another one if the average fitness associated to the topologies it contains is higher than that of another bin/cluster. Figure 16 graphically depicts the ranking obtained. The ranking was obtained through a mixture between the Friedman test and Holm–Bonferroni. The tests are a non-parametric tests that compare observations repeated on the same β-graphs families. Friedman test is based on the null-hypothesis that all β-graphs families are equivalent and according to this assumption all ranking must be equal. The statistic method for the Friedman's test is a Chi-square with $N - 1$ degrees of

freedom, where N is the number of repeated measures. When the p value for this test is small (usually <0.05) there is evidence to reject the null hypothesis. Holm–Bonferroni method is a simple sequentially rejective multiple test procedure that we use for extracting information of the ranking results of Friedman tests for all the clusters we generated. Remarkably, for all problems, except *NK2R*, only one (at most two) cluster of similar topologies are ranked first indicating a clear preference for a problem-dependent island topology. In Fig. 17 we show a histogram with the number of times a given bin has been the top ranked with statistical significance. Interestingly, bins with

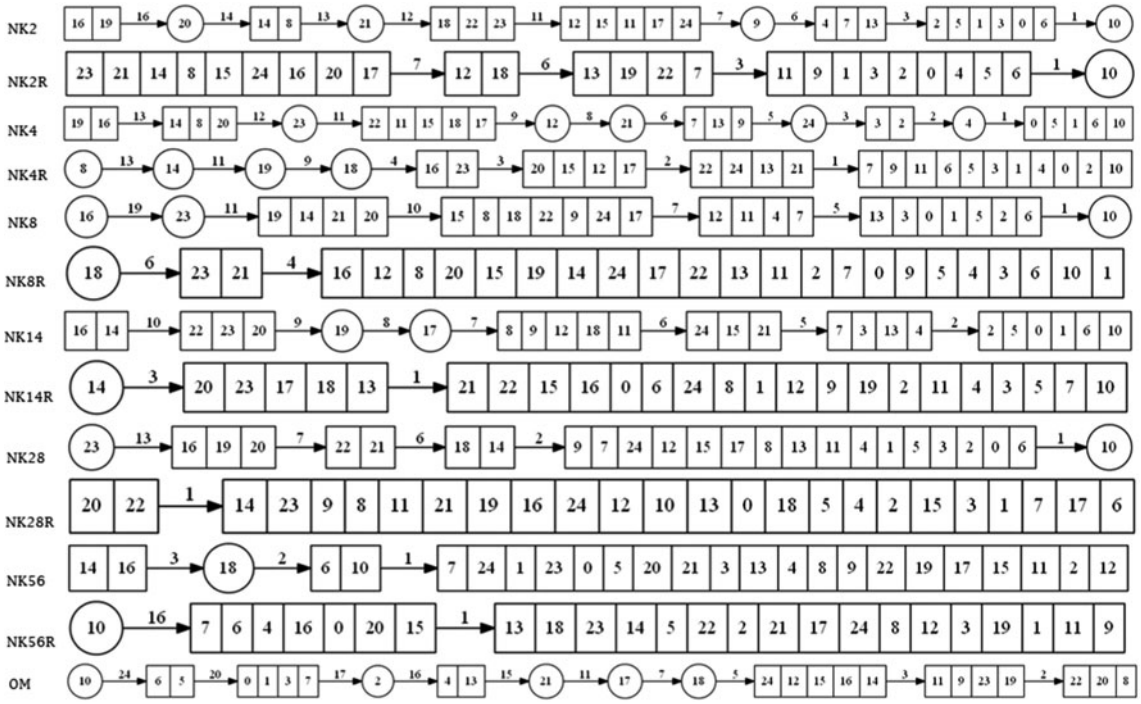

Fig. 16 This figure shows the 25 different clusters for the 13 considered problems. The clusters that share the same rank are represented together in a *rectangular box*. The high performing clusters are placed in the left of the graph and an order is established according to the scores obtained with the Friedman test plus Holm–Bonferroni method. Thus, in the case of the one-max problem the cluster 10 performs better than 24 other clusters with a 95 % percent confidence. In the same problem the clusters 6 and 5 share the same ranking as they both outperform 20 other clusters

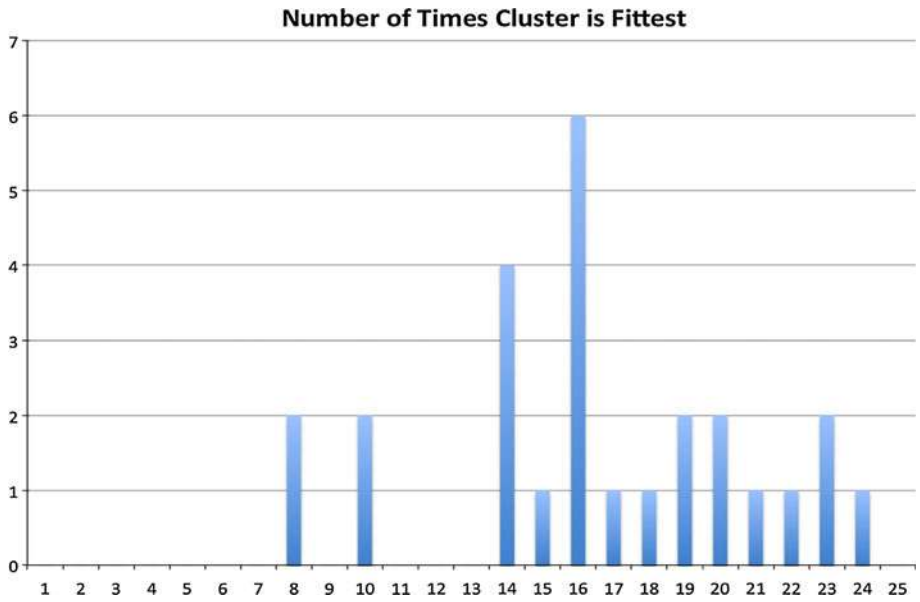

Fig. 17 The histogram of success for each cluster of island topologies

an average CPL ≤ 0.0451 have never (with only a few minor exceptions) been ranked top indicating that island topologies that are fully connected (i.e. functionally close to being panmictic) produce inferior performance. In the figure, the overall dominant cluster, being the fittest for 6 out of the 13 problems (see Table 1) has $\overline{CC} = 0.2902$ and $\overline{CPL} = 0.0710$, while the second dominant (4 out of 13) cluster has 0.4727 and 0.0491, respectively.

4 Conclusion

In this paper we have performed a systematic analysis of the correlation between island topologies and problem structure. We have been able to clearly establish that different problem structures require different island topologies for a PGA both through an analysis of idealised problems, *OneMax* and *NK*-landscapes, and a real-world scenario, namely, the Multi-Skills Call Centre. The analysis performed, involving the execution of thousands of simulations, the utilisation of biclustering and statistic analysis, has shown that there is no single island topology that is best across different problems and that the link between island topologies and problem structures is highly complex. Thus, the utilisation of adapting and self-adapting solvers that can dynamically choose (perhaps even construct) the interconnection scheme between islands seems to be the preferred way forward.

Acknowledgments N. Krasnogor would like to acknowledge UK EPSRC funding for project EP/H010432/1 and The Weizmann Institute of Science for granting him a Morris Belkin Visiting Professorship during which a part of this paper was executed. Iván Contreras and Ignacio Arnaldo are supported by Spanish Government Avanza Competitividad I+D+I: TSI-020100-2010-962 and Iyelmo INNPACTO-IPT-2011-1198-430000 projects and the mobility Grant "Orden ECD /3628/2011, de 26 de diciembre, Dirección General de Política Universitaria, Ministerio de Educación, Cultura y Deporte". The work has also been supported by Spanish Government grants TIN 2008-00508 and MEC CONSOLIDER CSD00C-07-20811.

References

Alba E, Tomassini M (2002) Parallelism and evolutionary algorithms. Evol Comput IEEE Trans 6(5):443–462

Barabási AL, Albert R (1999) Emergence of scaling in random networks. Science 286:509–512. http://jmvidal.cse.sc.edu/library/barabasi99a.pdf

Barrat A, Weigt M (2000) On the properties of small-world network models. Eur Phys J B Condens Matter Complex Syst 13: 547–560. doi:10.1007/s100510050067

Barthelemy M, Amaral LAN (1999) Small-world networks: evidence for a crossover picture. Phys Rev Lett 82(15):5. http://arxiv.org/abs/cond-mat/9903108

Brain Z, Addicoat M (2010) Using meta-genetic algorithms to tune parameters of genetic algorithms to find lowest energy molecular conformers. In: Proc. of the Alife XII Conference

Brucker P (2007) Scheduling algorithms, 5th edn. Springer Publishing Company, Berlin

Cantú-Paz E (1999a) Migration policies and takeover times in genetic algorithms. In: Proceedings of the genetic and evolutionary computing conference (GECCO) 1999, p 775

Cantú-Paz E (1999b) Topologies, migration rates, and multi-population parallel genetic algorithms

Clune J, Goings S, Punch B, Goodman E (2005) Investigations in Meta-GAs: panaceas or pipe dreams? In: Proceedings of the 2005 workshops on genetic and evolutionary computation, GECCO '05. ACM, New York, pp 235–241. doi:10.1145/1102256.1102311

Fialho A, Costa L, Schoenauer M, Sebag M (2008) Extreme value based adaptive operator selection. In: Proceedings of the 10th international conference on parallel problem solving from nature: PPSN X. Springer, Berlin, Heidelberg, pp 175–184

Giacobini M, Preuss M, Tomassini M (2006) Effects of scale-free and small-world topologies on binary coded self-adaptive cea. In: Gottlieb J, Raidl G (eds) Evolutionary computation in combinatorial optimization. Lecture Notes in Computer Science, vol 3906, Springer, Berlin, pp 86–98. doi:10.1007/117300958

Goeffon A, Lardeux F (2011) Optimal one-max strategy with dynamic island models. In: 23rd IEEE International Conference on Tools with artificial intelligence (ICTAI), 2011, pp 485–488. doi:10.1109/ICTAI.2011.79

Kennedy J, Mendes R (2002) Population structure and particle swarm performance. In: Proceedings of the 2002 Congress on evolutionary computation, 2002: CEC '02, vol 2, pp 1671–1676. doi:10.1109/CEC.2002.1004493

Krasnogor N (2004) Self-generating metaheuristics in bioinformatics: the protein structure comparison case. Genetic Progr Evolv Mach 5(2):181–201. http://www.cs.nott.ac.uk/~nxk/PAPERS/GPEM04.pdf

Krasnogor N (2012) Memetic algorithms. In: Rozenberg G, Bck T, Kok J (eds) Handbook of natural computing. Springer, Berlin, pp 905–935. doi:10.1007/978-3-540-92910-9-29

Krasnogor N, Gustafson S (2004) A study on the use of "self-generation" in memetic algorithms. Nat Comput 3:53–76. doi:10.1023/B:NACO.0000023419.83147.67

Lee M, Takagi H (1993) Integrating design stage of fuzzy systems using genetic algorithms. In: Second IEEE International Conference on fuzzy systems, 1993, vol 1, pp 612–617. doi:10.1109/FUZZY.1993.327418

Li L, Garibaldi JM, Krasnogor N (2006) Automated self-assembly programming paradigm: a particle swarm realization. In: University of Granada, pp 123–134

Li L, Garibaldi JM, Krasnogor N (2009) Automated self-assembly programming paradigm: the impact of network topology. Int J Intel Syst 24(7):793–817. doi:10.1002/int.20361

Liaw A (2006) Enhanced heat map. http://hosho.ees.hokudai.ac.jp/~kubo/Rdoc/library/gplots/html/heatmap.2.html

Michell A (1904) The limits of economy of materials in frame structures. Philos Mag 8:589–597

Millán-Ruiz D, Hidalgo I (2010a) Comparison of metaheuristics for workforce distribution in multi-skill call centres. In: Proceedings of the international joint conference on computational intelligence (ICEC 2010)

Millán-Ruiz D, Hidalgo I (2010b) A parallel memetic algorithm for workload distribution in dynamic multi-agents systems. In: Proceedings of the 3rd workshop on parallel architectures and bioinspired algorithms held in conjunction with PACT 2010

Millán-Ruiz D, Hidalgo J (2010c) A memetic algorithm for workforce distribution in dynamic multi-skill call centres. In: Cowling P, Merz P (eds) Evolutionary computation in combinatorial optimization. Lecture Notes in Computer Science, vol 6022. Springer, Berlin, pp 178–189

Nannen V, Eiben A (2006) A method for parameter calibration and relevance estimation in evolutionary algorithms. In: Proceedings of the 8th annual conference on Genetic and evolutionary computation, GECCO '06. ACM, New York, pp 183–190. doi:10.1145/1143997.1144029

Nannen V, Eiben AE (2007) Relevance estimation and value calibration of evolutionary algorithm parameters. In: Proceedings of the 20th international joint conference on Artifical intelligence, IJCAI'07. Morgan Kaufmann Publishers Inc., San Francisco, pp 975–980. http://dl.acm.org/citation.cfm?id=1625275.1625433

Newman MEJ, Moore C, Watts DJ (2000) Mean-field solution of the small-world network model. Phys Rev Lett 84:3201–3204. doi:10.1103/PhysRevLett.84.3201

Quang Q, Ong Y, Lim M, Krasnogor N (2009) Adaptive cellular memetic algorithm. Evol Comput 17(3):231–256. http://www.mitpressjournals.org/doi/pdf/10.1162/evco.2009.17.2.231 (for the latest version of this paper please refer to the journal website)

Shahsavar M, Najafi AA, Niaki STA (2011) Statistical design of genetic algorithms for combinatorial optimization problems. Math Probl Eng 2011:17. doi:10.1155/2011/872415

Verel S, Ochoa G, Tomassini M (2011) Local optima networks of nk landscapes with neutrality. Evol Comput IEEE Trans 15(6):783–797. doi:10.1109/TEVC.2010.2046175

Wang G, Wu D, Szeto K (2011) Quasi-parallel genetic algorithms with different communication topologies. In: IEEE Congress on Evolutionary computation (CEC), 2011, pp 721–727

Wang XF, Chen G (2003) Complex networks: small-world, scale-free and beyond. Circuits Syst Mag IEEE 3(1):6–20. doi:10.1109/MCAS.2003.1228503

Watts DJ, Strogatz SH (1998) Collective dynamics of 'small-world' networks. Nature 393(6684):440–442. doi:10.1038/30918

Weinberger ED (1996) NP completeness of kauffman's n-k model, a tuneable rugged fitness landscape. Working papers, Santa Fe Institute. http://EconPapers.repec.org/RePEc:wop:safiwp:96-02-003

Whitacre J, Sarker R, Pham Q (2008) The self-organization of interaction networks for nature-inspired optimization. Evol Comput IEEE Trans 12(2):220–230. doi:10.1109/TEVC.2007.900327

Woolley R, Stirling J, Radocea A, Krasnogor N, Moriarty P (2011) Automated probe microscopy via evolutionary optimization at the atomic scale. Appl Phys Lett 98(25):253104

Wright AH (1991) Genetic algorithms for real parameter optimization. In: Foundations of genetic algorithms. Morgan Kaufmann, San Mateo, pp 205–218