Multikulti Algorithm: using genotypic differences in adaptive distributed evolutionary algorithm migration policies

Lourdes Araujo, Juan J. Merelo Guervós

Abstract— **Migration policies in distributed evolutionary algorithms are bound to have, as much as any other evolutionary operator, an impact on the overall performance. However, they have not been an active area of research until recently, and this research has concentrated on the migration rate. In this paper we compare different migration policies, including our proposed *multikulti* methods, which choose the individuals that are going to be sent to other nodes based on the principle of *multiculturalism*: the individual sent should be as different as possible to the receiving population (represented in several possible ways). We have checked this policy on two discrete optimization problems for different number of nodes, and found that, in average or in median, multikulti policies outperform others like sending the best or a random individual; however, their advantage changes with the number of nodes involved and the difficulty of the problem. The success of these kind of policies is explained via the measurement of entropies, which are known to have an impact in the performance of the evolutionary algorithm.**

I. Introduction and state of the art

The fact that parallel evolutionary algorithms can obtain better results than sequential ones for the same computational effort [1] has been sometimes attributed to the fact that evolution proceeds differently in each node, and the effect that the immigrants from one node to another have on its diversity. The mating restriction that is inherent to the isolation of the population in several islands avoids premature convergence of the *whole* population, while the increased diversity attained with the incoming member of the other populations takes it closer to finding a solution. However, according to the intermediate disturbance hypothesis [2], the closer the immigrant is the current state of the population, the smaller effect it will have on the overall performance.

Diversity in the subpopulation is so important that it leads to improvement in quality and efficiency at the same time; and there are several mechanisms to preserve it: Herrera et al. [3], for instance, proposed a hierarchical configuration of subpopulations, each one of them running an evolutionary algorithm with different parameters, and connected so that there is a small variation from one subpopulation to the others connected to it. However, the migration mechanism itself can also be studied, since the selection of immigrants and the other aspects of migration will obviously have an effect on the overall performance. Here are some of these aspects:

1) the number of individuals undergoing migration,
2) the frequency of migration, i.e. the number of generations or evaluations between migrations,
3) the policy for selecting immigrants,
4) the immigrant replacement policy,
5) the topology of the communication among subpopulations,
6) the synchronous or asynchronous nature of the the connection among subpopulations.

Some of them have been studied in the literature: for instance, Alba et al. [4] look at the last one, concluding that asynchrony does not have a negative effect on performance, and can even outperform synchronous ones; Merelo et al. [2] looked at what would be the degree of asynchrony that would achieve the best algorithmic performance, applying also the above mentioned theory of intermediate disturbances.

Papers such as the ones by Cantú-Paz [5], [6], Alba and Troya [7], and Noda et al. [8] are more comprehensive in the study of different migration policies. Several results presented in these mentioned works indicate that diversity is a fundamental key in the success of the island model. Particularly, Cantú-Paz studied the four possible combinations of random and fitness-based emigration and replacement of existing individuals. He found that the migration policy that causes the greatest reduction in work (takeover time[1]) is to choose both the immigrants and the replacements according to their fitness, because this policy increases the selection pressure and may cause the algorithm to converge significantly faster. However, if convergence is too fast it can lead to algorithm failure, as Cantú-Paz [6] states referring to parallel EAs:

> *Rapid convergence is desirable, but an excessively fast convergence may cause the EA to converge prematurely to a suboptimal solution.*

So, other policies must also be considered. Some authors [9] have proposed a model different to the island one, which follows an approach of segregation and reunification. In this case subpopulations evolve independently until detecting local premature convergence, which is indicated by a selection pressure value computed in each subpopulation. If stagnation is detected the calculations for this subpopulation are stopped until the next reunification phase is reached. Such a reunification phase is initiated, if all subpopulations have converged prematurely.

Other authors Alba and Troya [7] found that in the island model migration of a random string prevents the "conquest"

Lourdes Araujo is with Dpto. Lenguajes y Sistemas Informáticos. UNED, Madrid, Spain (email lurdes@lsi.uned.es)

Juan J. Merelo Guervós is with Departamento de Arquitectura y Tecnología de Computadores, Universidad de Granada, Spain (email jmerelo@geneura.ugr.es)

[1]Number of generations required to converge to the best individual from the initial population, by applying selection only

978-1-4244-2959-2/09/$25.00 ©

effect in the target island for small or medium sized subpopulations. Finally, Noda et al. [8] proposed choosing which individuals to migrate and/or replace adaptively depending on some knowledge-oriented rules. To do this, each agent receives information about the fitness function from its peers. Besides, it considers, among other policies, one in which the individuals sent are chosen to be quite different from others previously sent. The tested adaptive policies have proved useful providing best solutions than the sequential execution. A later paper by Yang et al. [10] proposes selecting immigrants from an *elite*, instead of random ones, and using them and their complementaries as a pool for creating a set of immigrants; this yields good results in dynamic environments, which are known to need a high population diversity.

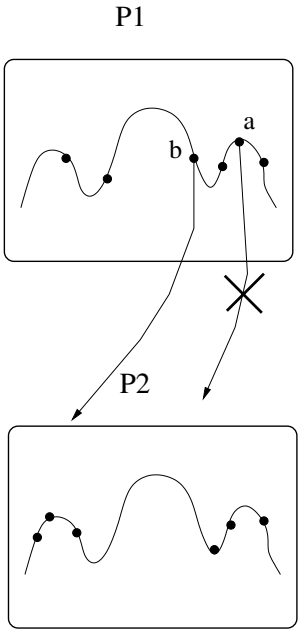


Fig. 1. Illustration of the *multikulti* immigrant selection policy: Individual a in subpopulation P1 has a higher fitness than individual b. However b is more different to the best individual in subpopulation P2. Accordingly, b is selected to be sent in the migration.

Some of the papers above study the how immigrants are selected; however, very few of them offer adaptive policies that, at the same time, try to enhance diversity via sending individuals that are as different as possible (but, as per the intermediate disturbance hypothesis mentioned above, not *too* different). In [11] authors proposed the MultiKulti algorithm and applied it to some discrete optimization problems, finding that it might yield better results than random or best-individual migration policies in some cases. The MultiKulti algorithm selects individuals to send to other populations based on its difference with a representative string of the receiving population. By sending strings that are *different enough*, it ultimately tries to increase performance via the diversity-boosting effect that an incoming *multicultural* immigrant might have.

The aim of this work is to exploit differences in the various subpopulation. To do this, we focus on the selection of the individuals to be sent to other subpopulation. Our thesis is that migrating individuals different enough to the destination subpopulation instead of the best individuals can result in a better performance through the diversity enhancement it produces. Consider the Figure 1 with two subpopulations. The black points represent the distribution of the population along the function to optimize. Individual a in subpopulation P1 has the highest fitness, and thus it would be sent to subpopulation P2 following the most common migration policies. We propose to send individual b, whose genotype is quite different from those of subpopulation P2. In the example it would lead to exploration of a new area of the search space where the global optimum is placed. In order to achieve this, the process corresponding to subpopulation P1 needs to receive information on the composition of the individuals in subpopulation P2.

We have considered different ways of providing this information in a concise manner. One of them is taken the best individual of subpopulation P2 as representative. The other one is using a kind of average genotype, the consensus sequence described later, as representative of subpopulation P2. Another important issue to investigate if the trade-off between promoting diversity and favoring the best individuals. Sending the most different individual as immigrant can imply that if its fitness value is low compared to those of the destination subpopulation, the immigrant would probably disappear immediately.

In this paper we will perform systematic experimentation of the different ways of performing the selection of immigrants on two functions, and parallel environments with different number of nodes; the number of nodes has an impact on performance, but also on diversity, with more nodes usually meaning better performance (in average number of evaluations), but also a higher chance of premature convergence.

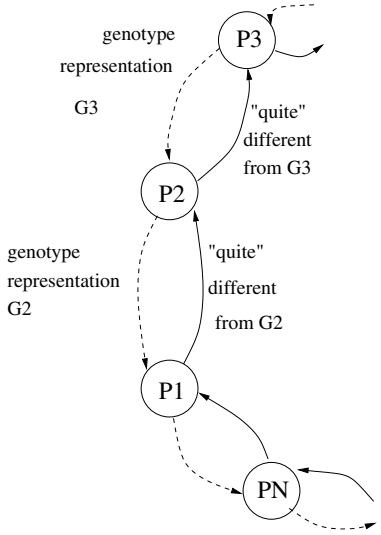


Fig. 2. Scheme of the multikulti algorithms.

The rest of the paper is organized as follows: section II describes the model details; section III is devoted to describe the evolutionary algorithm and its implementation; section IV presents and discusses the experimental results, and section V draws the main conclusions of this work.

II. Model Description

Without losing generality, we have considered a ring topology (Figure 2), in which each node can only send or receive information from the next and previous nodes in the ring. The node P_i receives from node P_{i+1} a message with a genotype that *represents* the population; we have considered two different ways of doing this in a concise manner:

1) With the **best individual** of the subpopulation. After a number of generations without exchanging individuals we can expect that each subpopulation is close enough to convergence for the best individual being a fair representation of the whole population. Obviously, this is not true in the first generations, but, in any case, it is impossible to be different from *all* individuals in the population, and what matters is that the immigrant sent is different enough from those individuals with which it is most likely to mate. This is the *base* approach.
2) With the population **consensus sequence**. This is a concept taken from biology [12] where it is defined as the sequence that reflects the most common base (in the case of DNA/RNA strings) or amino acid (in the case of protein strings) found at each position in a genome. The consensus sequence is a compact formulation to represent all possible alignments for any given numbers of sequences. In biology sometimes it is necessary for certain positions in a sequence to be made ambiguous because some residues cannot be resolved during laboratory experiments. A sequence with ambiguity codes is actually a mix of sequences, each having one of the nucleotides defined by the ambiguity at that position. The consensus sequence in essence is a condensed sequence with ambiguity codes that shows what nucleotides are allowed in each column. In our case it is composed of the most frequent allele for each position of the genotype. This approach is labeled *consensus* from now on.

Once the node P_i has got this information, it sends to node P_{i+1} an individual different enough from the individual representing subpopulation P_{i+1}. How this was implemented is explained below.

III. Experimental setup

Chromosomes of our GA are fixed-length binary strings. The selection mechanism to choose individuals for the new population uses a steady state algorithm, with single or two-point crossover operator and single-bit-flip mutation.

P-Peaks and the massively multimodal deceptive problem (MMDP), two of the three discrete optimization problems presented by Giacobini et al. in [13] have been selected for testing. These problems, while being both multimodal, represent different degrees of difficulty for parallel evolutionary optimization. They are described below.

These problems have been implemented and integrated in the `Algorithm::Evolutionary` library, which is freely available under the GPL license from `http://fon.gs/ae-perl/`. In order to simulate a parallel algorithm, the *cooperative multitasking* Perl module POE has been used; each node is represented by a POE *session*. The rest of the evolutionary algorithm has been implemented using the same `Algorithm::Evolutionary` Perl module [14]. The program, along with the parameter sets used, is also available under an open source license from the same site. In order to speed up experiments, all experiments for the same fitness function were made in a single run, so that the evaluation caché would be effectively put to use; this feature present in Algorithm::Evolutionary effectively reduced run time to a few hours at most for the most difficult problem, MMDP.

Tests were initially made for 2, 4 and 8 nodes, to see how the different migration methods fared under different initial diversity conditions. The population was kept constant, dividing it among the nodes.

In this simulated parallel scenario, each node runs a rank-based substitution, steady state algorithm. At the end of a preset number of generations, each node sends a single individual to the other node according to the policy being tested. These problems, and the specifics of their implementation, are explained below.

A. Problems tested

The MMDP [15] is a deceptive problem composed of k subproblems of 6 bits each one (s_i). Depending of the number of ones (unitation) s_i takes the values depicted next:

$$\begin{array}{ll} fitness_{s_i}(0) = 1.0 & fitness_{s_i}(1) = 0.0 \\ fitness_{s_i}(2) = 0.360384 & fitness_{s_i}(3) = 0.640576 \\ fitness_{s_i}(4) = 0.360384 & fitness_{s_i}(5) = 0.0 \\ fitness_{s_i}(6) = 1.0 & \end{array}$$

The fitness value is defined as the sum of the s_i subproblems with an optimum of k (equation 1). The number of local optima is quite large (22^k), while there are only 2^k global solutions. In this paper, we consider a single instance with $k = 20$.

$$f_{MMDP}(\vec{s}) = \sum_{i=1}^{k} fitness_{s_i} \tag{1}$$

We have considered instances with $k = 15$ and $k = 20$ subproblems respectively, whose maximum is then $f_{MMDP}(\vec{s}) = 15, 20$; solutions will be represented in a chromosome of length 90/120. Due to the nature of this problem, and the fact that in the initial tests the algorithm was not able to find the solution in many cases, we hypothesized that it might be due to crossover acting also as a mutation algorithm and thus disrupting the subproblems already *solved* by the EA. That is why, besides the traditional two points

Parameter	Value	
	xOver	GBX
Chromosome length	90	120
Population	2048	
Selection rate	20%	
Generations to migration	10	
Mutation priority	2	
2-point crossover priority	3	
Max number of evaluations	150000	

TABLE I

EVOLUTIONARY ALGORITHM PARAMETERS USED IN THE MMDP EXPERIMENTS, WITH THE TRADITIONAL 2-POINTS CROSSOVER ON THE LEFT, THE GENE-BOUNDARY RESPECTING CROSSOVER (GBX) ON THE RIGHT.

Parameter	Value
Chromosome length	64,100,128
Population	256
Selection rate	20%
Generations to migration	20
Mutation priority	1
2-points crossover priority	4
Max number of evaluations	100000

TABLE II

EVOLUTIONARY ALGORITHM PARAMETERS USED IN THE P-PEAKS EXPERIMENTS.

crossover, we used another crossover operator that *respects gene boundaries* (which we have called GBX), that is, only interchanges whole genes (whole subproblems, in this case) between parents, leaving the mutation function to the bit-flip operator. Besides, these initial tests showed that results obtained by the 4- and 8-node system were much worse than for two nodes, due possible to the lack of initial diversity in the smaller populations. The parameters used in the EA are shown in Table I, and feature a low selection rate (to give incoming immigrants the chance to survive for several generations), high relative mutation rate (40% of chromosomes, once *priority* has been normalized), and low relative crossover rate (40%). In this case such a high mutation rate, leading to a high degree of exploration, was used to explore a highest proportion of the search space due to the difficulty of the problem.

On the other hand, the P-Peaks problem is a multimodal problem generator proposed by De Jong [16], and is created by generating P random $N - bit$ strings where the fitness value of a string $\vec{x}$ is the number of bits that $\vec{x}$ has in common with the nearest peak divided by N. In the experiments made in this paper we will consider $P = 100$; the optimum fitness is 1.0.

$$f_{P-Peaks}(\vec{x}) = \frac{1}{N} \max_{1 \leq i \leq p} \{N - HammingDistance(\vec{x}, Peak_i)\} \quad (2)$$

We consider an instance of $P = 100$ and 64,100 and 128 bits where the optimum fitness is 1.0 (Equation 2). The parameters used in the EA are shown in Table II. Selection rate was set at the same value as before for the same reasons, but mutation priority was lowered to make this instance more *exploitative* with respect to the one above.

B. Other implementation details

As mentioned above, the source for the experiments as well as the parameter files are available from our group's CVS server, and will eventually be incorporated into the main A::E distribution. The main feature of this implementation, besides using caches for fitness evaluation and other functions such as checking the arity of genetic operators, was the use of the language YAML for storing the experiment results. YAML is a language for data structure serialization, which allowed to easily extract information from the experiments and generate data files, which were then fed into the R statistics package to compute statistics and plot them.

IV. EXPERIMENTAL RESULTS

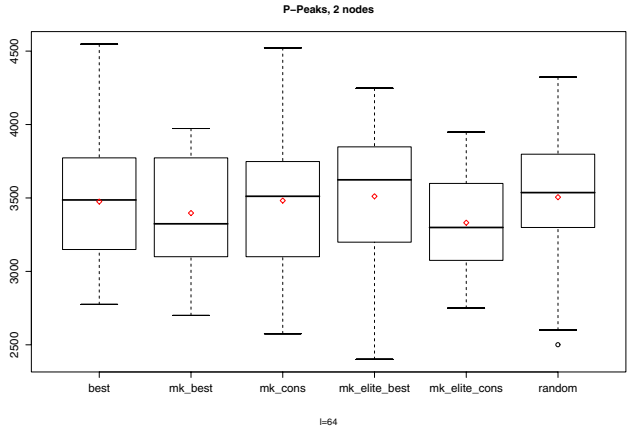
Four different versions of the multikulti migration policy have been tested here: multikulti *base* and *elite*, each one with choosing the most different immigrant based on the *best* and *consensus* individual. Accordingly, in our experiments we compare results for the following policies:

- *best_none*: the immigrant sent is the best individual of the source population.
- *multikulti_best/_consensus*: the immigrant is the individual farthest away (in Hamming distance) from the best individual/consensus string of the target population.
- *multikulti_elite_best/_consensus*: the immigrant is the individual most different to the best one or the consensus string of the target population, but chosen from the population with fitness above median.
- *random_none*: the immigrant is chosen randomly.

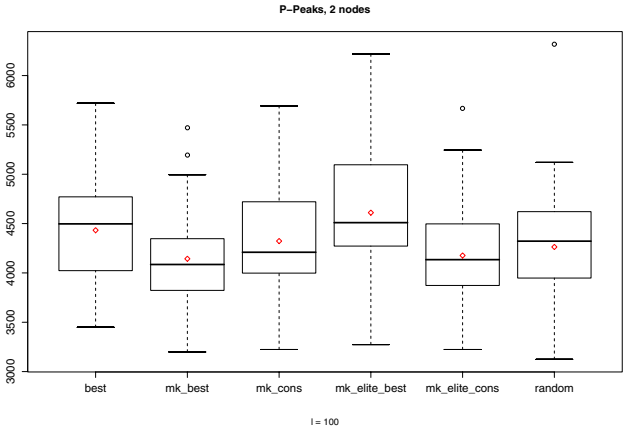
Each combination was run 30 times, with termination condition being success or a maximum of evaluations. Results are shown in Figures 3 and Figures 4.

There are several conclusions to this experiment, which are tangential to the target of this paper: the highest influence on the number of evaluations (algorithmic performance) is the number of nodes, with more nodes meaning less evaluations. As has been indicated in the state of the art, EAs profit from division in islands, achieving a double benefit from parallelization: more evaluations happening at the same time, and also less evaluations needed to reach target. From 2 to 8 nodes, P-Peaks needs half as many evaluations for all lengths (64, 100 and 128).

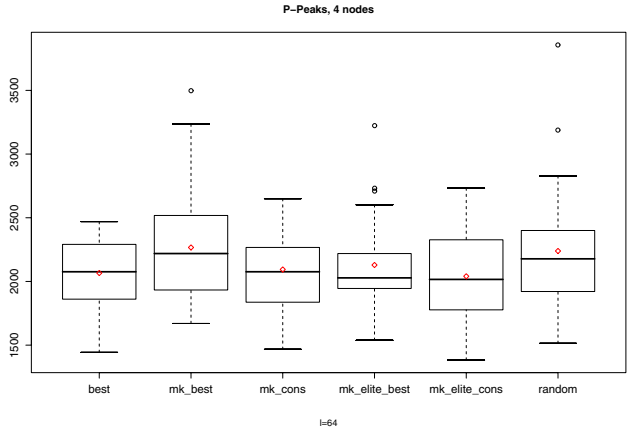
Next conclusion is that while differences are not too big for the easiest problem ($l = 64$), on average, multikulti algorithms are better than non-adaptive *random* or *best* strategies for $l = 100, 128$. Curiously enough, there is more difference for the biggest and smallest number of nodes (higher and lower initial diversity), while the difference is small for the intermediate number of nodes (4). The problem is that, as it can be seen in the graphs, the winner multikulti strategy is not always the same. While multikulti_best seems to obtain the best results for 8 nodes, it obtain worse results for $l = 64$ and 4 nodes.



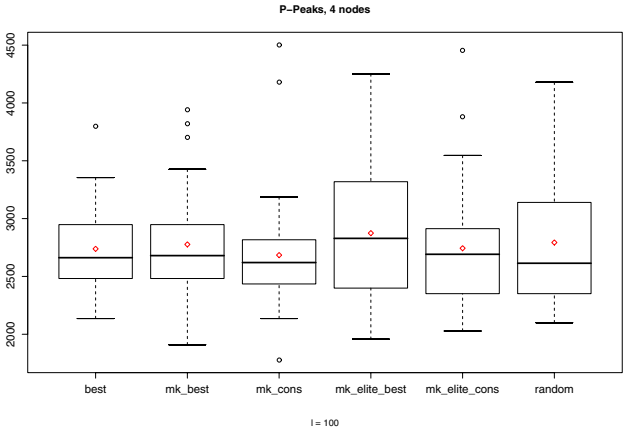
(a) 2 nodes



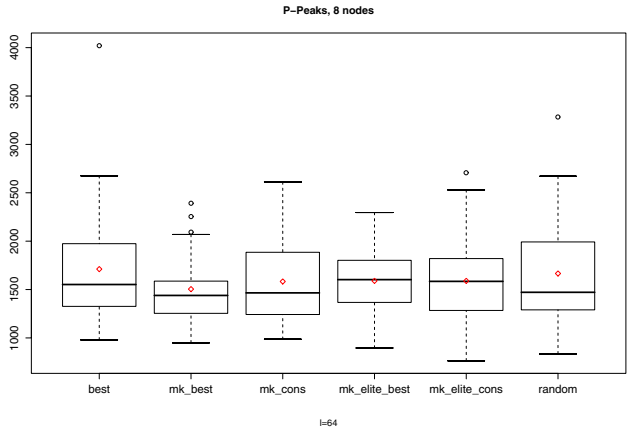
(b) 2 nodes



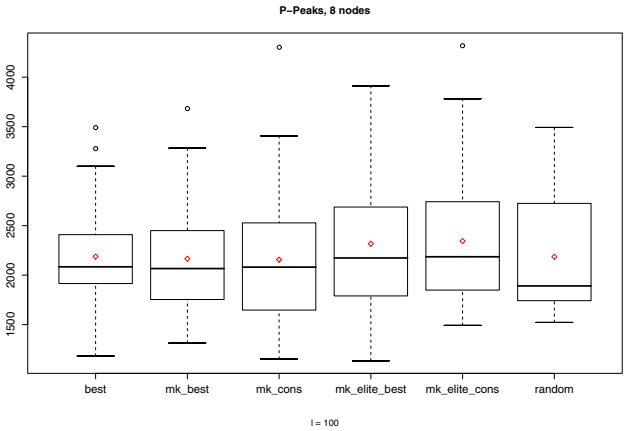
(c) 4 nodes



(d) 4 nodes



(e) 8 nodes

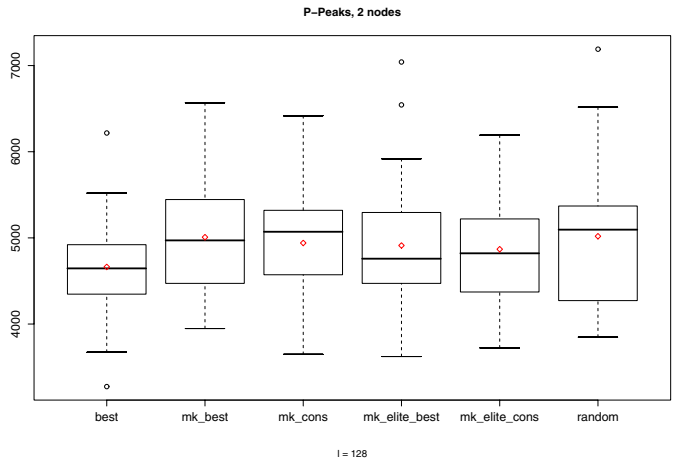


(f) 8 nodes

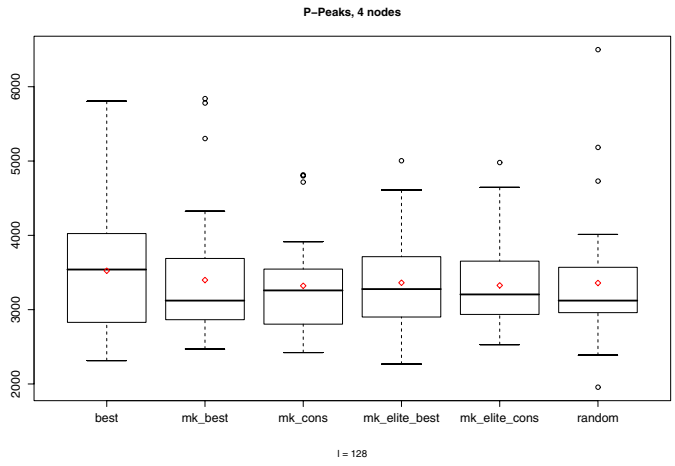
Fig. 3. Boxplot of the number of evaluations needed to find the solution for P-Peaks with 64 (figures on the left) and 100 bits (figures on the right) for 2, 4 and 8 nodes. Averages are represented as red dots. Best stands for best_none, mk for multikulti, cons for consensus, and random for random_none.

The only case where a non-multikulti strategy beats the others is for $l = 100$ and 8 nodes, where the random migration strategy appears as winner. It should be noted also that sending the best individual (the usual strategy in most distributed EC papers) is never better than any other, and in many cases obtains the worst result.

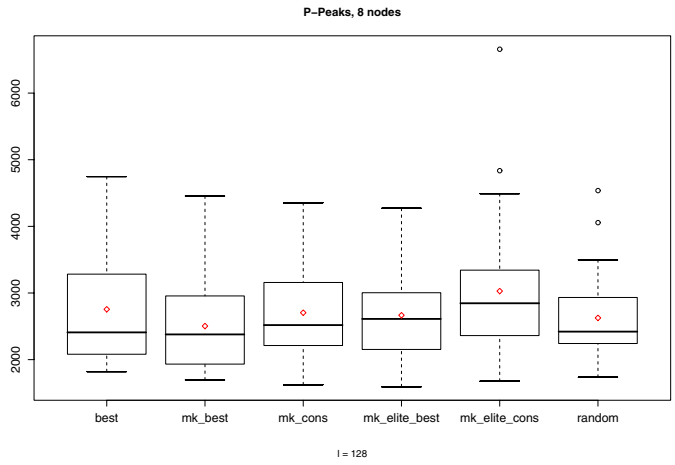
The conclusion from these experiments is that while trading immigrants that are different enough does have a definite effect on the performance, it is relatively easy to overshoot the difference, or else being too similar to have any effect. It is easy to imagine, for instance, that at the end of the run all solutions are close to each other, and the



(a) 2 nodes

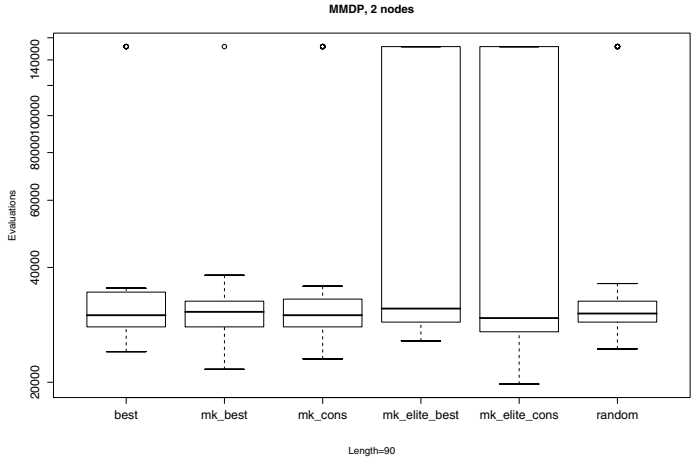


(b) 4 nodes

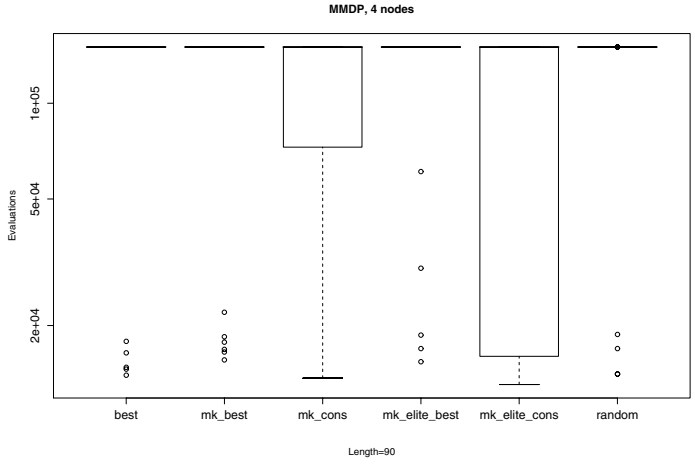


(c) 8 nodes

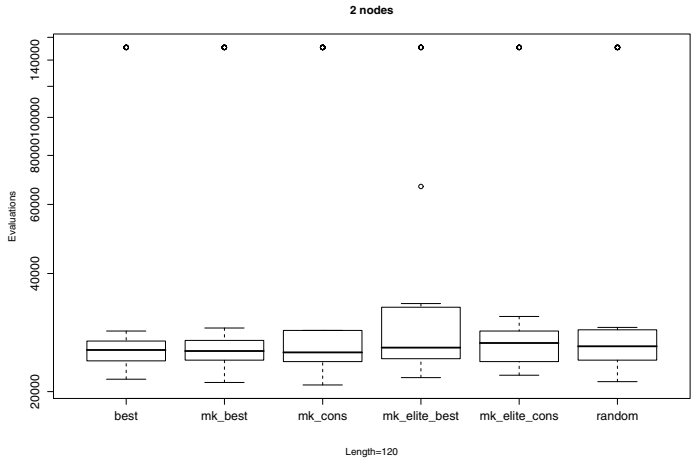
Fig. 4. Boxplot of the number of evaluations needed to find the solution for P-Peaks with 128 bits and 2 to 8 nodes. Averages are represented as red dots.



(a) 2 point Crossover, 2 nodes



(b) 2 point Crossover, 4 nodes



(c) Gene-boundary Crossover

Fig. 5. Boxplot (with logarithmic y axis) of the number of evaluations needed to find the solution for MMDP for two nodes, and 90 bits/2 point crossover (top, for 2 and 4 nodes) 120 bits/GBX (bottom)

difference between emitting and receiving population are not so big.

In any case, we have used a more difficult problem, MMDP, to test the hypothesis of validity of multikulti policies, and how they become increasingly relevant with the difficulty of the problem. Experiments made on MMDP were as above, with 30 runs for every parameter setting, but we have used only two and 4 nodes, since with this operator rates setting it couldn't find the solution on most runs for 8 nodes. This implies performance decreased with the number of nodes, probably due to the loss of initial diversity inherent in smaller populations (1024 for 2 nodes, 512 for 4 nodes).

Figure 5 shows the results obtained for the MMDP problem with 2/4 nodes and different types of crossover: two points crossover, which has been run for the 90 bit problem (15 subproblems, 15×6 = 90) and GBX crossover with 120 bits (20 subproblems, 20×6 = 120). We can observe that the GBX operator leads to faster convergence, and concerning the migration policies we can observe again that the multikulti policies perform better in both charts. However, once again there is a difference. In the first set of experiments with $l = 90$, the multikulti policies outperform all others. The best median for 2 nodes is obtained by multikulti_elite_best and the best mean by multikulti_best, and multikulti_elite_consensus beats all others, on average, for 4 nodes (medians in all cases are equal to the maximum number of evaluations). The fact that mean and average do not coincide point to a skewed distribution, with many runs not reaching the optimum, but, in any case, with no multikulti policy arising as the clear winner, it is clear that multikulti policies outperform non-adaptive best and random.

Results are quite different when trying to solve a harder problem ($l = 120$, 20 MMDP subproblems) using the GBX operator. Best average is obtained by best_none (followed by multikulti_elite_consensus, while the best median is given by the multikulti_consensus policy, followed closely by multikulti_best and best_none. Average differences, in any case, hover around 1%.

In order to investigate whether the quality of the solutions found is truly correlated with entropy, we have measured the entropy for some *MMDP* experiments. The results are shown in Figure 6, which shows the different evolution paths of phenotypic entropy (computed using the Shannon formula) with the multikulti-elite migration policy (left) and the best migration policy (center). This multikulti policy, which has not performed specially well in the experiments above, has been chosen in order to test if it does have an effect on diversity as measured by entropy; other multikulti policies with a better behaviour are expected to have a bigger effect. Even so, the graph shows that the behavior is quite different. The multikulti policy, not only keeps the entropy high, but considerably increases it in some populations during evolution. The policy of migrating the best provides quite much lower levels of entropy; with a decreasing trend that never changes, leading to a collapse of entropy from cycle 12. This proves the utility of the multikulti policy to maintain diversity, and supports the result that the improvement in the number of evaluations is due precisely to this diversity-enhancing effect brought by the *multikulti* policies.

V. Conclusions

This paper has explored new alternatives to promote diversity in an island model. This is achieved by selecting as immigrant individuals with a genotype different enough to the destination population. Because there is a trade-off between promoting diversity and favoring the best individuals, we have performed experiments to find out the choice (of immigrants and representatives) policy which produces the best results.

Results in several problems show that, in most cases, multikulti policies are able to outperform non-adaptive random and best-individual migration policies. However, while in the P-Peaks problem, in general, multikulti policies are better (see Figure 3 (a), (b) (c) and (d), and Figure 4 (a) and (b)), the MMDP problem yields conflicting results, with random or best-individual policies achieving results on a par with multikulti (see Figure 5).

However, it is not clear which multikulti policy achieves the best results; each one arises as a winner in a particular combination of problem size and parallel environment. multikulti_elite_cons does show up repeatedly as the best (see Figure 3 (a), (b) and (c)), or close to the best, but differences are not significant as to choose it above the rest. This might be due to the representativity of the best or consensus chromosome, which is used along all the algorithm to select the outgoing immigrant. The consensus sequence might be a better representative (being closer on average to all the population) at the beginning of the run, while the best chromosome might be better at the end of the run. In any case, adaptively selecting a population representative might yield better, or at least more consistent, results.

Entropy measures, on the other hand, do show that multikulti policies lead to increased diversity, and that when they do, they will outperform best or random policies. However, there are several issues: first, it is difficult to characterize a population using a single string, second, it is also hard to send an individual that is different enough and, at the same time, is not immediately selected out of the population by its selection policy.

This is probably one of the avenues of research we should use in the future. Instead of using selection by the emitter, it should probably be the receiver the one choosing who is coming in, so that it can choose an individual different *and* fit enough. However, there is a trade-off between the amount of information that can be sent between nodes and what performance increases can be obtained from it, so algorithmic performance will have to be analyzed along with (simulated or real) network traffic to obtain the best results.

Past experiments by the authors [2] also show that starting populations at different time will have a positive impact on performance by decreasing the number of evaluations; we will study how late start and migration policies interact with each other, and which combination yields the best results.

We also intend to develop a parallel implementation of the system, which will allow us to measure execution times too. We are also working on alternative mechanisms to characterize the destination population, and thus select the most appropriate immigrants. We will also test results obtained by changing other algorithm parameter such as number of immigrants or the number of nodes.

ACKNOWLEDGEMENTS

This paper has been funded in part by the Spanish MICINN project NoHNES (Spanish Ministerio de Educación y Ciencia - TIN2007-68083-C02-01) and the Junta de Andalucía P06-TIC-02025.

REFERENCES

[1] E. Alba, A. J. Nebro, and J. M. Troya, "Heterogeneous computing and parallel genetic algorithms," *J. Parallel Distrib. Comput.*, vol. 62, no. 9, pp. 1362–1385, 2002.

[2] J. J. Merelo, A. M. Mora, P. A. Castillo, J. L. J. Laredo, L. Araujo, K. C. Sharman, A. I. Esparcia-Alc'azar, E. Alfaro-Cid, and C. Cotta, "Testing the intermediate disturbance hypothesis: Effect of asynchronous population incorporation on multi-deme evolutionary algorithms," in *Parallel Problem Solving from Nature - PPSN X*, 2008, pp. 266–275.

[3] F. Herrera, M. Lozano, and C. Moraga, "Hierarchical distributed genetic algorithms," *nternational journal of intelligent systems*, vol. 14, no. 11, pp. 1099–1121, 1999.

[4] E. Alba and J. M. Troya, "Analyzing synchronous and asynchronous parallel distributed genetic algorithms," *Future Generation Comp. Syst.*, vol. 17, no. 4, pp. 451–465, 2001.

[5] E. Cantú-Paz, "Migration policies and takeover times in genetic algorithms," in *GECCO*, 1999, p. 775.

[6] E. Cantú-Paz, "Migration policies, selection pressure, and parallel evolutionary algorithms," *Journal of Heuristics*, vol. 7, no. 4, pp. 311–334, 2001.

[7] E. Alba and J. M. Troya, "Influence of the migration policy in parallel distributed gas with structured and panmictic populations," *Appl. Intell.*, vol. 12, no. 3, pp. 163–181, 2000.

[8] E. Noda, A. Coelho, I. Ricarte, A. Yamakami, and A. Freitas, "Devising adaptive migration policies for cooperative distributed genetic algorithms," *Systems, Man and Cybernetics, 2002 IEEE International Conference on*, vol. 6, pp. 6 pp. vol.6–, 6-9 Oct. 2002.

[9] M. Affenzeller and S. Wagner, "Sasegasa: A new generic parallel evolutionary algorithm for achieving highest quality results," *Journal of Heuristics*, vol. 10, no. 3, pp. 243–267, 2004.

[10] S. Yang and R. Tinós, "A hybrid immigrants scheme for genetic algorithms in dynamic environments," *International Journal of Automation and Computing*, vol. 4, no. 3, pp. 243–254, 2007.

[11] L. Araujo, J. Guervos, C. Cotta, and F. de Vega, "MultiKulti Algorithm: Migrating the Most Different Genotypes in an Island Model," *Arxiv preprint arXiv:0806.2843*, 2008.

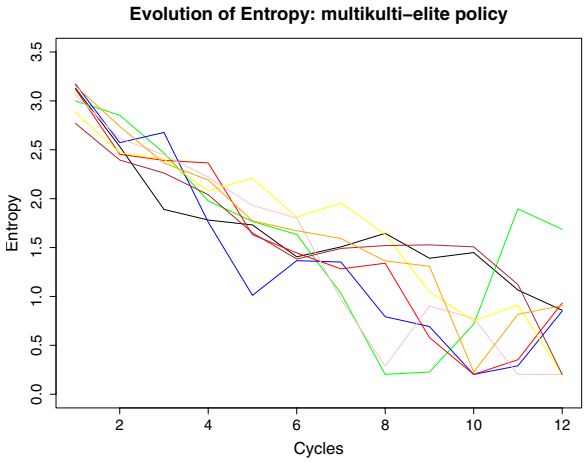
[12] J. Watson, T. Baker, S. Bell, A. Gann, M. Levine, and R. Losick, *Molecular Biology of the Gene*. San Francisco: Benjamin Cummings, 2004.

[13] M. Giacobini, M. Preuss, and M. Tomassini, "Effects of scale-free and small-world topologies on binary coded self-adaptive CEA," in *Evolutionary Computation in Combinatorial Optimization – EvoCOP 2006*, ser. LNCS, J. Gottlieb and G. R. Raidl, Eds., vol. 3906. Budapest: Springer Verlag, 10-12 April 2006, pp. 85–96.

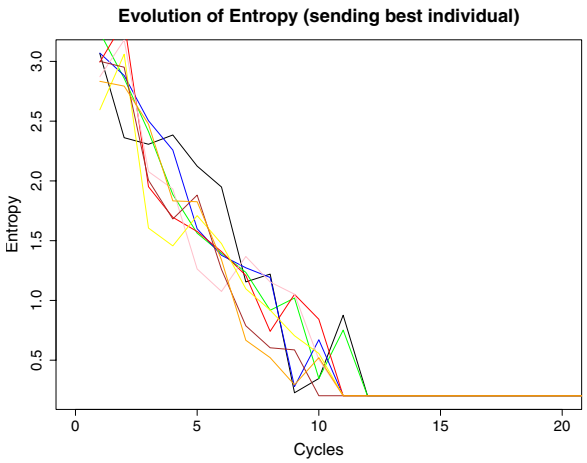
[14] J. J. M. Guervos, "Evolutionary computation in Perl," in *YAPC::Europe::2002*, M. Perl Mongers, Ed., 2002, pp. 2–22.

[15] D. E. Goldberg, K. Deb, and J. Horn, "Massive multimodality, deception, and genetic algorithms," in *Parallel Problem Solving from Nature, 2*, R. Männer and B. Manderick, Eds. Amsterdam: Elsevier Science Publishers, B. V., 1992. [Online]. Available: citeseer.ist.psu.edu/goldberg92massive.html

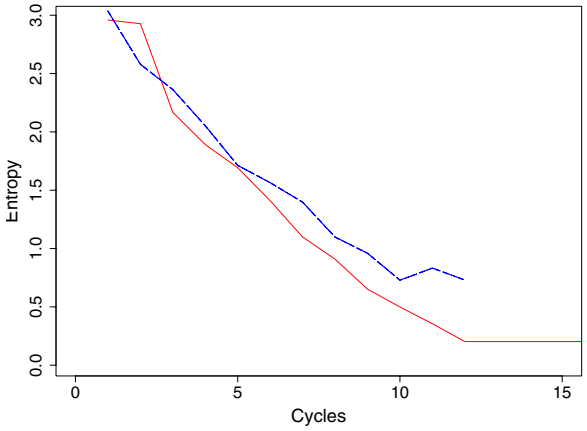
[16] K. A. D. Jong, M. A. Potter, and W. M. Spears, "Using problem generators to explore the effects of epistasis," in *Proceedings of the Seventh International Conference on Genetic Algorithms (ICGA97)*, T. Bäck, Ed. San Francisco, CA: Morgan Kaufmann, 1997. [Online]. Available: citeseer.ist.psu.edu/dejong97using.html



mke



best



average

Fig. 6. Entropy (computed using the Shannon formula $H(P) = -\sum_{g \in P} p(f(g)) log_b p(f(g))$, where g is a member of the population, $f(g)$ its fitness, and $p(f(g))$ the frequency of that fitness across the whole population) in a typical run of the MMDP problem, with the multikulti-elite migration policy (left) and the best migration policy (center). Every line corresponds to a different population, of the eight running in parallel. The figure at the bottom compares average values, with the dashed line corresponding to the multikulti-elite experiment and the other to the experiment that sends the best individual.