Available online at www.sciencedirect.com
SCIENCE DIRECT®
J. Parallel Distrib. Comput. 66 (2006) 991–1001
Journal of Parallel and Distributed Computing
www.elsevier.com/locate/jpdc

Initial approaches to the application of islands-based parallel EDAs in continuous domains

Luis delaOssa*, José A. Gámez, José M. Puerta

Departamento de Sistemas Informáticos, Universidad de Castilla-La Mancha, Campus Universitario s/n, Albacete, 02071, Spain

Received 10 May 2005; received in revised form 14 March 2006; accepted 21 March 2006
Available online 11 May 2006

Abstract

Estimation of distribution algorithms (EDAs) are a wide-ranging family of evolutionary algorithms whose common feature is the way they evolve by learning a probability distribution from the best individuals in a population and sampling it to generate the next one. Although they have been widely applied to solve combinatorial optimization problems, there are also extensions that work with continuous variables. In this paper [this paper is an extended version of delaOssa et al. Initial approaches to the application of islands-based parellel EDAs in continuous domains, in: Proceedings of the 34th International Conference on Parallel Processing Workshops (ICPP 2005 Workshops), Oslo, 2005, pp. 580–587] we focus on the solution of the latter by means of island models. Besides evaluating the performance of traditional island models when applied to EDAs, our main goal consists in achieving some insight about the behavior and benefits of the migration of probability models that this framework allow.

Keywords: Parallel evolutionary algorithms; Estimation of distribution algorithms; Numerical optimization

1. Introduction

Early attempts to parallelize genetic algorithms soon went further than the simple distribution of the evaluation of individuals among processors. Although the models that emerged preserve many of the original features from the sequential algorithms, they introduce some basic differences of behavior which improve their performance even when executing on a single processor [2]. Thus, besides reducing execution time by taking advantage of the computational power of parallel machines, they achieve higher quality solutions.

Very frequently, the use of a structured population in the form of islands or demes is responsible for such benefits. These kinds of *parallel evolutionary algorithms* (PEAs) are known as multideme algorithms or island models. They basically work by considering a set of subpopulations that evolve independently and occasionally exchange information (generally a subset of individuals) among each other.

In the family of evolutionary computation, EDAs represent a relatively recent paradigm that explicitly models the population of good solutions by using a probabilistic model, which is used to guide the search. Although EDAs were initially proposed to solve combinatorial optimization problems (discrete domains) several attempts to apply them to numerical optimization (continuous domains) have been made. In literature we can find several approaches to the application of parallel EDAs to combinatorial domains; however, little work has been done in the numerical case. In fact, to the best of our knowledge there are only two (very recent) studies in which this task has been tackled. These works, [7,16], have been developed at the same time but independently. Thus, [7] is a subset of this paper and was presented in June, while [16] is an in-press work.

Madera et al. [16] propose a parallel asynchronous island-based estimation of the distribution algorithm which is applied to combinatorial and numerical problems showing better performance (with respect to the number of evaluations) than the sequential algorithm. Concretely, UMDA (univariate marginal distribution algorithm) is used inside each island. The

* Corresponding author.
E-mail addresses: ldelaossa@info-ab.uclm.es (L. delaOssa), jgamez@info-ab.uclm.es (J.A. Gámez), jpuerta@info-ab.uclm.es (J.M. Puerta).

0743-7315/$ - see front matter © 2006 Elsevier Inc. All rights reserved.
doi:10.1016/j.jpdc.2006.03.005

information (asynchronously) shared between the different islands is a set of individuals (the best individuals are sent from the source and the worst individuals are replaced in the target island).

On the other hand, [7] is completely devoted to dealing with numerical domains. As in [16], UMDA is one of the EDAs used inside each island but, instead of comparing sequential vs parallel models, the target of the paper is to compare migrating individuals or migrating probabilistic models. This paper is an extension of [7] and in our opinion our main contributions are: (1) to design parallel EDAs that share probabilistic models and to compare them with the classical migration of individuals; (2) to enlarge the class of algorithms considered from the *univariate* case [7] to the *multivariate* one, offering a general framework that can be used with any kind of EDAs; and (3) to analyze the impact of different parameters (number of islands, population size, problem size) on the proposed algorithms.

With this purpose in mind, this article, is organized into six sections apart from the introduction. Section 2 briefly describes EDAs and the approaches used in this work: UMDA_g and $\text{EMNA}_{\text{GLOBAL}}$. In Section 3, we give some basic ideas about PEAs based on the island model. In Section 4, we show the way we propose to carry out migrations when working with EDAs and the island-based algorithms designed. In Section 5, we show experiments carried out and the analysis of the results obtained. Finally, in Section 6 we present our conclusions and some possibilities for future research.

2. Estimation of distribution algorithms: EDAs

EDAs [15] belong to the family of population-based evolutionary algorithms. Their main cycle is quite similar to that of genetic algorithms but they do not use local information through crossover or mutation of individuals. Instead, they capture global knowledge from the population in the form of a probabilistic model. Thus, the transition between populations is made by estimating a probability distribution/model from (usually the best individuals of) the population and sampling it to obtain the new population.

The general schema of an EDA is as follows:

1. $D_0 \leftarrow$ Generate initial population (N individuals)
2. Evaluate D_0
3. $i = 0$
4. Repeat
   a. $i = i + 1$
   b. $D_s \leftarrow$ Select $S \leqslant N$ individuals from D_{i-1}
   d. Estimate a new model M from D_s
   e. $D_{\text{new}} \leftarrow$ Sample N individuals from M
   e. Evaluate D_{new}
   g. $D_i \leftarrow$ Select N individuals from $D_{i-1} \cup D_{\text{new}}$

   until stop condition

The main advantage of EDAs with respect to other techniques lies in the capability of the probabilistic model, M, to make the interrelations among variables explicit. Also noteworthy is the small number of parameters that must be specified and the fact that many of them can be set to their (commonly used) default values. On the other hand, as it is very hard to deal with the joint probability distribution, M is taken as a simplification of it. This gives rise to three main groups of EDAs [15]: univariate models, which assume that variables are marginally independent; bivariate models, which accept dependences between pairs of variables; and multivariate models where there is no limitation of dependences.

In order to test our models we are going to use two different algorithms: a univariate one, UMDA_g, and $\text{EMNA}_{\text{GLOBAL}}$ which is an easy-to-learn multivariate model. The second one is more complex and powerful, although it requires a larger population than UMDA_g to evolve effectively. Therefore, important behavioral differences can be expected when applying island models.

2.1. $UMDA_g$

In literature we can find different proposals of univariate EDAs in continuous domains: continuous PBIL [22] and Gaussian UMDA [14,13] which are extensions of the discrete versions to the continuous case by using unidimensional and independent normal densities to model the joint distribution; UMDA_t which models the unidimensional and independent densities by using the TSP-distribution [8]; FHH and FWH [9] which uses marginal histograms to model the population; and the IDEA framework [4] which makes a clear distinction between the algorithm used (univariate, ...) and the distribution considered (uniform, normal, kernels, ...).

In the univariate case we have selected UMDA_g (*univariate marginal distribution algorithm for Gaussian models*) as the algorithm that evolves inside each island. Our choice is due to different reasons: besides usually being used as a baseline for comparison, its simple structure makes it easier to identify the benefits coming from parallelism. Moreover, we have successfully tested the parallelization of the UMDA [17] algorithm in the discrete case.

As has been mentioned, UMDA_g uses the normal distribution to model the density of each variable, and the joint (n-dimensional) density is factorized as the product of all the unidimensional and independent normal densities:

$$f(\boldsymbol{x};\boldsymbol{\theta}) = f_N(\boldsymbol{x};\boldsymbol{\mu},\boldsymbol{\Sigma}) = \prod_{i=1}^{n} f_N(x_i;\mu_i,\sigma_i^2) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma_i}} \cdot e^{-\frac{1}{2}\left(\frac{x_i-\mu_i}{\sigma_i}\right)^2}. \quad (1)$$

Thus, model induction is reduced to the estimation of μ and σ^2 for each variable X_i, $i = 1, \ldots, n$. Furthermore, as each variable is independently simulated, any standard method for sampling from a normal distribution can be used (see for example [21]).

However, as mentioned in [22] the performance of a univariate algorithm hardly depends on the way that the variance (σ^2) is estimated. In fact, four different ways of computing the

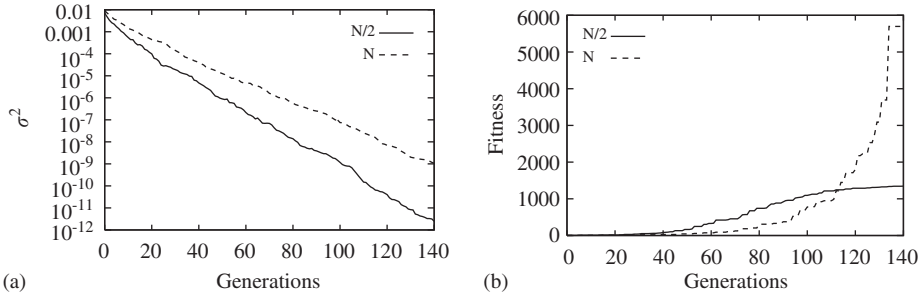

Fig. 1. Evolution of σ^2 for the left-most variable and fitness in a run of the *Summation Cancellation* problem with dimension 10: (a) σ^2_N vs $\sigma^2_{N/2}$ and (b) fitness $\left(\sigma^2_{i(N)}\right)$ vs fitness $\left(\sigma^2_{i(N/2)}\right)$.

new variance are proposed in [22]. The main problem with the variance is that if this parameter decreases too fast, then very similar values are generated which leads to the premature convergence phenomenon. In our case this situation is emphasized by the use of truncation selection, that is, we generate population D_i by selecting the best N individuals from $D_{i-1} \cup D_{new}$. Furthermore, as the model is induced from the best $\frac{N}{2}$ individuals of D_i, the variance decreases too fast.

In order to maintain population diversity during the search progress we have tried different ways of computing the parameters (μ, σ^2) when estimating the probabilistic model. In the end, we have selected the following:

- μ is estimated as usual, that is, from the best $\frac{N}{2}$ individuals taken from D_i.
- σ^2 is computed by using the whole population D_i (N individuals), but considering the mean previously computed by using only the best half of the population:

$$\sigma^2 = \frac{\sum_{j=1}^{N} (x_i^j - \mu_{i(N/2)}(X_i))^2}{N}, \tag{2}$$

where x_i^j is the value of variable X_i in the jth individual and $\mu_{N/2}(X_i)$ is the mean value of variable X_i over the best $\frac{N}{2}$ individuals of the population.

In this way, we try to quickly obtain good values for the mean, but diversity is introduced by avoiding a too fast decrease in variance. Fig. 1(a) shows the evolution of the two different estimations of the variance parameter (in logarithmic scale) for the left-most variable during the first 140 generations in a typical execution of the *Summation Cancellation* problem (dimension 10). As can be seen, Eq. (2) yields a smoother decrease than when using $\sigma_{N/2}$, allowing the evasion of premature convergence to non-optimal values (see fitness evolution in Fig. 1(b)).

2.2. $EMNA_{GLOBAL}$

As mentioned above, in the multivariate approach there is no assumption about independences. Therefore, this kind of model can gather all the information about relations among variables and make use of it to generate the next population. In literature we can find several models of multivariate EDAs for continuous domains: $\text{EMNA}_{\text{GLOBAL}}$, EGNA_{ee}, EGNA_{BGe}, EGNA_{BIC} ([15, Chapter 8]). Whereas the first one, $\text{EMNA}_{\text{GLOBAL}}$, codifies the model by directly learning the covariance matrix, the rest use different techniques to learn a Gaussian network which represents the probabilistic model. These models are richer than univariate or bivariate, but their learning and sampling is rather more complex.

In our work, we are going to use the algorithm $\text{EMNA}_{\text{GLOBAL}}$, which uses a multinormal distribution, $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, to model the joint density:

$$f(\boldsymbol{x}; \boldsymbol{\theta}) = (2\pi)^{\frac{-n}{2}} |\boldsymbol{\Sigma}|^{\frac{-1}{2}} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu})' \boldsymbol{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{\mu})\right), \tag{3}$$

where $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_n)$ is the vector of means, $\boldsymbol{\Sigma}$ is the $(n \times n)$ covariance matrix, prime denotes the transpose operator, $|\boldsymbol{\Sigma}|$ denotes the determinant of $\boldsymbol{\Sigma}$ and $\boldsymbol{\Sigma}^{-1}$ denotes the inverse matrix of $\boldsymbol{\Sigma}$.

As in UMDA_g, in $\text{EMNA}_{\text{GLOBAL}}$ there is no structural learning, only parameter learning. It requires the estimation of $2n + \binom{n-1}{2}$ parameters at each generation: n means, n variances and $\binom{n-1}{2}$ covariances. This can be done very efficiently in a single pass through the population.

With respect to sampling [21], since $\boldsymbol{\Sigma}$ is positive, definite and symmetric, there exists a unique lower triangular matrix $\boldsymbol{L}$ such as $\boldsymbol{\Sigma} = \boldsymbol{L}\boldsymbol{L}'$, called Cholesky factor of $\boldsymbol{\Sigma}$. Samples of the vector $\boldsymbol{x}$ can be represented as $\boldsymbol{x} = \boldsymbol{\mu} + \boldsymbol{L}^{-1}\boldsymbol{z}$, where $\boldsymbol{z} = z_1, \ldots, z_n$ is a vector whose components are marginally distributed to the standard normal distribution $N(0, 1)$.

3. Coarse-grained PEAs: the islands model

The ways to parallelize an evolutionary algorithm go from the distribution of the evaluations of individuals (global parallelization) which does not change its behavior, to new models that make a very intensive use of communication [5], which generally give rise to a completely different algorithm (fine grained parallelization).

Among these, we can find the so-called coarse-grained parallelization or island-based models. They consist of using several populations that evolve independently according to the original

sequential algorithm and occasionally interchange individuals among each other.

These models have been successfully tested with different kinds of algorithms such as genetic algorithms, simulated annealing or variable neighborhood search (VNS) (see [3, Chapters 5, 11 and 12]). However, despite being the most widely used and popular kind of PEA, there are still some questions which are not completely clear since the effects of migration are not fully known and because there are many parameters involved in their design. Besides the parameter setting required for each island to evolve (population size, elitism, etc.) it is necessary to specify the way that interaction among islands will be carried out. The parallel model is mainly specified by using the following parameters:

- *Number of islands*: When real parallelism is used this parameter is constrained by the available hardware, but when parallelism is simulated this is a tunable parameter.
- *Number of individuals* that migrate from one island to another.
- *Migration policies*: The most common option consists in fixing the number of generations elapsed in an island before individuals are received or sent from/to others.
- *Topology*: Islands and migrations must be defined by some interconnection topology (star, ring, etc.).
- *Replacement policies*: It is necessary to define how population inside an island is formed after a migration is carried out. The most common option relies on elitism, that is, received individuals are added to current population and, afterwards, individuals with the worst fitness are discarded.

4. Islands-based EDAs: migration of probabilistic models

As has been mentioned before, cooperation among islands is carried out by interchanging *information* among each other. This *information* usually takes the form of a subset of individuals that are passed to an island, which in turn incorporates it. There are some works on parallel EDAs that follow this approach in combinatorial (i.e. [1,16,19,20]) and numerical (i.e. [12,16]) optimization. However, because of the intrinsic functioning of EDAs, parallelization can be designed in a different way. In fact, as EDAs resume the properties of the best individuals in the population by using a compact representation, this opens the possibility of sharing only those parameters that define the probabilistic model instead of sharing individuals. This alternative scheme (migration of the probabilistic model) has been explored in the combinatorial case [6], showing better performance than the standard migration of individuals. In general we can expect some benefits when interchanging this kind of information:

- A probabilistic model gathers more information than a subset of individuals because it usually represents and resumes a larger number of them. Besides, it may contain additional information about interrelations among variables.
- This richer information is also easier to analyze since it is more compact and explicitly represented.
- Whereas the options when migrating individuals are reduced to their incorporation or not to the receiver island evolution, the migration of models allows more flexibility.

4.1. Model combination

The next point to discuss is how the receiver island incorporates (or otherwise) the information received. That is, we have to decide how two probabilistic models (the resident and the incoming one) are combined. We refer to this step as *model combination*.

In this initial study, our first choice has been to try a strategy successfully tested in discrete domains with univariate models [6]. It consists in replacing the probabilistic model in an island by *a convex combination* between the resident and the incoming models, yielding a mixed model that belongs to the same family. In the univariate case, given a parameter $\beta \in [0, 1]$ and the parameters for two marginal normal distributions (μ_i,σ_i^2) and (μ_j,σ_j^2) the combined model is obtained as

$$\begin{aligned} \mu_k &= \beta \cdot \mu_i + (1-\beta) \cdot \mu_j, \\ \sigma_k^2 &= \beta^2 \cdot \sigma_i^2 + (1-\beta)^2 \cdot \sigma_j^2, \end{aligned} \tag{4}$$

where (μ_k, σ_k^2) are also the parameters for a normal distribution. In the case of more complex models the combination can be carried out by adding two weighted covariance matrices.

However, preliminary experiments with this type of model combination showed us that the variance of the obtained model decreases at a critical velocity, causing the algorithms to converge prematurely. It seems that as both UMDA_g and $\text{EMNA}_{\text{GLOBAL}}$ use models best suited for unimodal fitness landscapes, this (closed) model combination yields a model again only suited for unimodal fitness. [1] Because of this, we have abandoned this type of model combination and study a more general alternative way.

Our new approach for model combination in continuous domains consists of using a mixture model $M = \sum_i \beta_i M_i$ as a linear convex combination of simpler distributions, where $\{\beta_i\}$ are called the mixture coefficients and satisfy $\sum_i \beta_i = 1$; $\{M_i\}$ are called the components of the mixture model, each one having its own parameters, and can even belong to different types of models or distributions. Mixture models provide a flexible way for modeling complex distributions, because they allow the combination of single distributions (such as normal distributions) into a joint model by using a building-block like scheme.

Mixture models are used, for instance, in clustering problems, where each component in the mixture corresponds to a different cluster from which the data can be generated and the coefficients represent the importance of each cluster. This is very closely related to the island model because each island can be associated to a cluster of solutions in the mixture model and the coefficients could be the relative importance of the

[1] This possible explanation was provided by an anonymous reviewer.

fitness solutions reached in an island with respect to the fitness solutions in the other island. So, this kind of combination is a natural way of mixing models in PEA.

As this type of model combination is suitable for all families of EDAs, not only for univariate models, we can understand this proposal of island-based parallel EDAs with migration of models as a framework that can be instantiated by using probabilistic models of different complexity and does not force the use of the same type of algorithm at each island. Of course, we plan to investigate these topics as future research.

In this work, we consider a mixture of two distributions, so we only have to learn one coefficient $\beta \in [0, 1]$ involved in the mixture, and after that we just sample from the learnt mixture. Thus, if M_r is the resident model on a given island and M_i is the incoming model, the mixture is sampled as follows:

for $i = 1$ to N do
  $u = random(0, 1)$
  if $u < \beta$ $sample(M_r)$
  else $sample(M_i)$

We discuss β in the next section.

4.2. Proposed algorithms

Due to the great number of parameters which need to be specified in a PEA and bearing in mind the goal of this work, we have fixed some of these by using *standard* values or according to our experience (previous work or preliminary experiments). Thus, (1) we use eight islands arranged in an unidimensional ring; (2) we use synchronous communication every five generations; and (3) we use standard parameter setting for the algorithms running inside each island (truncation selection, best half population is used for learning, etc.).

For the sake of comparison we consider the traditional migration-of-individuals scheme and additionally we propose two migration-of-models-based algorithms.

- pEDAInd. In this algorithm we migrate the best 10% of the individuals in the population. The receiver island uses elitism to incorporate these individuals, that is, it enlarges its population with the incoming individuals and then truncates the population by removing the worst 10% individuals.
- pEDAMod(0.9). This algorithm uses migration of models and combines them by using the mixture described above. In this case β is given the value 0.9. The choice of this value conforms to two different requirements: (1) it represents a rather conservative value that avoids strongly degrading the resident model even if the incoming one is of low quality; and (2) in some way allows a fair comparison with pEDAInd because in the mixture around 10% of the individuals are expected to be sampled from the incoming model.
- pEDAMod(Adpt). This algorithm differs from the previous one in the way in which β is estimated. In this case instead of using a fixed value for β we use an adaptive method that takes into account the goodness of the incoming model with respect to the resident one. Thus, an island j sends/receives a pair (M, f) where M is a probabilistic model and f is its associated fitness, which is computed as the fitness average of the 50% best individuals of the island population. Therefore, f_j can be viewed as the goodness of population/model in island j. Finally,

$$\beta = \begin{cases} \dfrac{f_r}{f_i + f_r} & \text{if } f_i \geqslant f_r, \\ 1 & \text{otherwise,} \end{cases}$$

where f_r is the fitness associated to the resident model and f_i is the fitness associated to the incoming model. That is, if an island receives a model which is better than the one it possesses, then a low weight is assigned to the resident model. Otherwise a conservative policy is used and the island does not incorporate any information.

5. Experimental study

In this section, we describe the experimental study carried out and the results obtained. Our purpose is twofold: besides evaluating the behavior of UMDA_g and $\text{EMNA}_{\text{GLOBAL}}$ when applying island models, we will try to compare migration of individuals vs migration of models.

5.1. Study cases

In order to test the proposed algorithms we have chosen four standard functions broadly used in the related literature. In addition, we have used two more functions that exhibit different features. The first is a rotation of the *Cigar* function, whereas the other (*Test* [7]) is a deceptive function with different properties. In all cases the function takes a n-dimensional individual $\boldsymbol{x} = (x_1, \ldots, x_n)$ as input. The mathematical description of the six functions is

- *Summation Cancellation*:

$$f_{\text{sum}}(\boldsymbol{x}) = \frac{1}{(10^{-5} + \sum_{i=1}^{n} |y_i|)},$$
$$y_1 = x_1,$$
$$y_i = x_i + y_{i-1}; \quad i = 2, \ldots, n,$$
$$-0.16 \leqslant x_i \leqslant 0.16; \quad i = 1, \ldots, n.$$

- *Griewangk*:

$$f_{\text{gri}}(\boldsymbol{x}) = 1 + \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right)$$
$$-600 \leqslant x_i \leqslant 600; \quad i = 1, \ldots, n.$$

- *Schwefel*:

$$f_{sch}(\boldsymbol{x}) = \sum_{i=1}^{n} \left[(x_i - x_i^2)^2 + (x_i - 1)^2\right]$$
$$-10 \leqslant x_i \leqslant 10; \quad i = 1, \ldots, n.$$

- *Rosenbrock generalized*:

$$f_{\text{ros}}(\boldsymbol{x}) = \sum_{i=1}^{n-1} \left[100 \cdot (x_{i+1} - x_i^2)^2 + (1 - x_i)^2\right]$$
$$-10 \leqslant x_i \leqslant 10;\ i = 1, \ldots, n.$$

- *RotatedCigar*:

$$f_{\text{cigar}}(\boldsymbol{x}) = y_1^2 + 10^4 \sum_{i=1}^{n} y_i^2,$$

with

$$\boldsymbol{y} = [\boldsymbol{o}_1, \ldots, \boldsymbol{o}_n]^T \boldsymbol{x}, \quad \text{i.e. } y_i = \boldsymbol{o}_i^T \boldsymbol{x} (\text{see text}),$$
$$-3 \leqslant x_i \leqslant 7; \quad i = 1, \ldots, n.$$

- *Test*:

$$f_{\text{test}}(\boldsymbol{x}) = \sum_{j=1}^{m} f_{\text{dec}}^5(\boldsymbol{s_j}),$$

$$\boldsymbol{s_j} = (x_{5j-4}, x_{5j-3}, x_{5j-2}, x_{5j-1}, x_{5j}); \quad n = 5m,$$

$$f_{\text{dec}}^5(x_1, x_2, \ldots, x_5) = \left(\sum_{i=1}^{4} x_i\right) \cdot f^1(x_5) + 4 \cdot \max\{abs(f^1(-1)), 1\}$$

$$f^1(x) = (1 - (x - p)^2),$$
$$-1 \leqslant x_i \leqslant 1; \quad i = 1, \ldots, n.$$

In the first case the goal is to maximize the function and the optimum value is 100 000.

In the next three cases the goal is to minimize, and the optimum value is 0. In the four problems the optimum is achieved by the configuration $(0, 0, \ldots, 0)$. The *RotatedCigar* function is a minimization one with optimum 0. As described in [10] we have carried out a rotation of the function *Cigar*, $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$, where $\boldsymbol{A} = [\boldsymbol{o}_1, \ldots, \boldsymbol{o}_n]^T$ implements an orthonormal linear transformation of $\boldsymbol{x}$ obtained through *Gram Schmidt* process. Finally, the *Test* problem [7] is a deceptive function with a parameter, $0 \leqslant p \leqslant 1$, that changes the properties of the function. Thus, for $f_{\text{dec}}^5(\boldsymbol{x})$, values of $f^1(x_5)$ below 0 leads x_1 to x_4 to -1, since values above 0 lead these values to 1. The role of p has to do with the probability of $f^1(x_5)$ being above 0 and the difference between values of local and global optimum (Fig. 2).

We have considered an instance of this function which corresponds to $p = 0.75$. Thus, the a priori probability of $f^1(x_5)$ being above 0 is 0.625, so it induces a trend for the blocks to reach the configuration $(1, 1, 1, 1, 0.75)$ and the local optima $8m$. Values of $f^1(x_5)$ below 0, with an a priori probability of 0.375 leads to $(-1, -1, -1, -1, -1)$ and a value of $16.5m$.

In order to have only problems of maximization, we have multiplied the fitness of the functions which had to be minimized by -1.

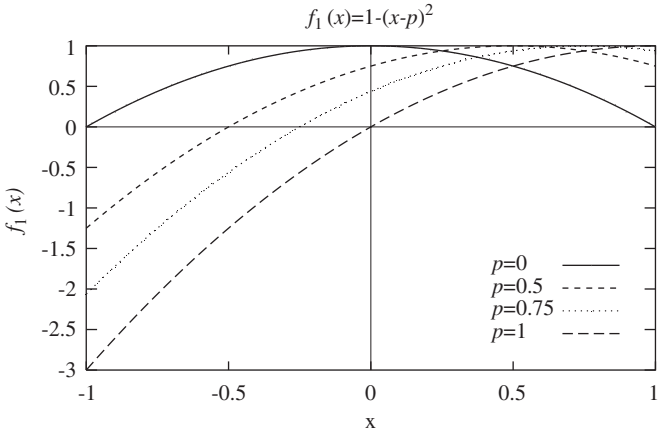

Fig. 2. $f^1(x)$ for different values of p.

5.2. Experiments

In the experiments, we have considered the algorithms and parameters described in Section 4. In order to provide a baseline for comparison, we have replaced the search method (UMDA$_g$ and EMNA$_{\text{GLOBAL}}$) by a genetic algorithm. This algorithm uses *arithmetic* crossover (with probability 0.6 and $\alpha = 0.25$) and *Mühlenbein* mutation [18] (with probability 0.05). Both operators have been successfully tested and compared with others in [11]. Furthermore, we have also considered sequential versions of UMDA$_g$ and EMNA$_{\text{GLOBAL}}$.

We have divided our experiments into two different groups. In the first of these, we analyze the convergence of the algorithms whereas with the second we try to achieve some insight into scalability issues.

5.2.1. Convergence

As the relation between population and quality of the solutions is not linear, we have tested all the proposed algorithms with population size 256 and 2048, splitting the population equally among the eight islands in the parallel algorithms. We have tested the behavior of each configuration problem-algorithm-population by running it 50 times, allowing a maximum of 512 000 evaluations. Table 1 shows the obtained results, mean ± deviation of fitness and number of evaluations. The problem size considered is $n = 10$ in all cases except for *Test* where we have used $n = 20$.

In order to determine whether there is some algorithm or type of migration which outperforms the rest, we compared, for each problem, the group of 4 algorithms that use the same probabilistic model. These comparisons were made as follows:

1. We chose the algorithm with best mean fitness as the reference (we use number of evaluations followed by random decision for tie-breaking). Then, we compared this algorithm with the others in its group by using an unpaired t-test (significance level $\alpha = 0.05$). The fitness of the reference algorithm and all those which do not present a significant difference with respect to it are marked with •.
2. Once we had identified the algorithms with best fitness, we proceeded in the same way for the number of evaluations,

Table 1
Experimental results for the six problems with population size 256 and 2048

Population 256			Population 2048	
SummationCancellation(10)				
Genetic 10%	4621.76087 ± 9665.27592	300 467.2 ± 89 179.86537	4089.53718 ± 2990.67859	510 156.8±4480.26122
UMDAg	• 97 405.02053±14 367.96418	194 763.66±26 111.64882	91 299.59464±13 212.52913	498 625.46±65 551.70533
pEDAInd—UMDA—10%	• 100 000±0.00002	• 171 084.8±12 601.52408	98 500.88305±306.91794	511 590.4±2026.99432
pEDAMod(0.9)—UMDA	• 100000±0	232 704±23 734.8289	94 250.27981±1124.03892	510 361.6±4789.60071
pEDAMod(Adpt)—UMDA	• 99 971.28644±163.23752	273 920±80 456.72727	95 414.51467±959.92388	509 952±5067.48579
EMNA	• 100 000±0	• 37 675.22±315.327	• 100 000±0	307 419.94±1039.68094
pEDAInd-EMNA—10%	131.89238±121.93095	20 812.8±2295.71299	• 100 000±0	286 720±0
pEDAMod(0.9)— EMNA	75.99395±35.35615	20 428.8±2636.69602	• 100 000±0	303 923.2±4825.21161
pEDAMod(Adpt)—EMNA	53.73989±25.3624	15 692.8±2251.59448	• 100 000±0	296 960±0
Rosenbrock(10)				
Genetic 10%	−3.92687 ± 1.71642	512 000±0	−6.42791 ± 1.0945	512 000±0
UMDAg	−7.79648 ± 0.08518	38 114.74±2725.31401	7.77431±0.03037	316 755.74±11 294.86208
pEDAInd—UMDA—10%	−5.70504 ± 0.17617	512 000±0	−7.23648 ± 0.09418	512 000±0
pEDAMod(0.9)—UMDA	−6.55944 ± 0.54114	512 000±0	−7.58713 ± 0.06361	512 000±0
pEDAMod(Adpt)—UMDA	• −0.28805 ± 0.06104	•512 000±0	−5.89412 ± 0.30222	511 590.4±2896.30938
EMNA	−7.66572 ± 0.54625	17 356.8±679.20304	−7.40217 ± 0.18108	143 794.66±17588.25225
pEDAInd—EMNA—10%	−16.85461 ± 8.70958	22 348.8±4139.28153	• −6.82922 ± 0.28296	511 385.6±2456.55488
pEDAMod(0.9)—EMNA	−22.83298 ± 11.88463	21 555.2±3492.14059	• −6.85737 ± 0.35934	49 2134.4±39 816.05851
pEDAMod(Adpt)—EMNA	−44.67464 ± 33.18334	18304±2056.63486	• −6.92935 ± 0.35556	• 33 4028.8±110 518.63931
Schwefel (10)				
Genetic 10%	0±0	212 480±39 000.47514	−0.00216 ± 0.00144	512 000±0
UMDAg	−0.02896 ± 0.01304	25 775.04±1321.24074	−0.02479 ± 0.00398	202 697.72±9073.52838
pEDAInd—UMDA—10%	• 0±0	167 603.2±31 841.735	−0.00082 ± 0.00081	512 000±0
pEDAMod(0.9)—UMDA	• 0±0.00001	219 110.4±78 001.24606	−0.00364 ± 0.00265	512 000±0
pEDAMod(Adpt)—UMDA	• 0±0	• 48 307.2±10 140.94128	• 0±0	334 643.2±51 113.82789
EMNA	−1.91042 ± 1.09402	16 085.62±864.54846	−0.87415 ± 0.20443	126 513.36±23 256.80303
pEDAInd—EMNA—10%	−3.65674 ± 1.73999	20 454.4±2362.89356	• −0.24985 ± 0.18448	• 510 566.4±7743.47645
pEDAMod(0.9)—EMNA	−4.99876 ± 2.09143	20 249.6±2365.72202	−0.63161 ± 0.3714	468 992±73 957.52029
pEDAMod(Adpt)—EMNA	−5.1184 ± 2.73176	17 075.2±2 203.56135	−0.69711 ± 0.32355	291 430.4±99 672.21988
Griewangk (10)				
Genetic 10%	−0.07444 ± 0.02816	79 232±17 440.10485	−0.03478 ± 0.02031	420 659.2±70 849.41997
UMDAg	−0.35167 ± 0.16933	16 313.8±9605.22245	−0.27945 ± 0.09238	117 946±57 619.52701
pEDAInd—UMDA—10%	−0.02649 ± 0.0238	53 145.6±15 321.29	−0.0304 ± 0.02345	428 441.6±77 803.09923
pEDAMod(0.9)—UMDA	• −0.00172 ± 0.00393	• 39 910.4±13 103.82916	• −0.00472 ± 0.01224	425 574.4±75 082.06025
pEDAMod(Adpt)—UMDA	−0.00505 ± 0.01047	33920±10 172.01924	• −0.00507 ± 0.013	419 020.8±79 878.96568
EMNA	−0.0623 ± 0.14189	21 721.12±8194.75666	−0.19092 ± 0.13308	146 097.3±109 398.70613
pEDAInd—EMNA—10%	−0.31505 ± 0.1622	33 100.8±15 290.20332	−0.01927 ± 0.03775	321 126.4±115 533.4899
pEDAMod(0.9)—EMNA	−0.5174 ± 0.20408	24 780.8±4199.82694	• 0±0	163 430.4±16 152.48676
pEDAMod(Adpt)—EMNA	−0.86217 ± 0.28036	18 816±2951.31655	• 0±0	• 160 768±14 810.29978
RotatedCigar (10)				
Genetic 10%	−1.30513 ± 1.99197	358 041.6±204 952.76588	−0.38898 ± 0.47926	487 014.4±46 635.55533
UMDAg	−0.63085 ± 0.77231	45 011.48±7708.3914	−0.30033 ± 0.23005	363 600.88±32 719.10891
pEDAInd—UMDA—10%	−0.4955 ± 0.66971	511 795.2±979.95052	−0.20009 ± 0.27923	493 772.8±66 339.88596
pEDAMod(0.9)—UMDA	−0.29689 ± 0.60245	388 736±194 399.80516	• −0.05935 ± 0.07984	• 469 811.2±97 115.2065
pEDAMod(Adpt)—UMDA	−0.40035 ± 0.67451	482 432±116 925.45457	• −0.08531 ± 0.13762	• 491 520±62 783.67447
EMNA	−52.2593 ± 215.55052	15 068.44±3018.02367	• 0±0	104 100.66±1525.89624
pEDAInd—EMNA—10%	−4276.03436 ± 3009.29172	19 814.4±3211.62948	• 0±0	• 102 400±0
pEDAMod(0.9)—EMNA	−7945.42954 ± 6469.66892	21 376±3304.10258	• 0±0	110 182.4±4417.73169
pEDAMod(Adpt)—EMNA	−11 467.80758 ± 6181.33831	18 201.6±2776.66028	• 0±0	104 857.6±4417.73169
Test (20)				
Genetic 10%	59.54±3.35196	81 817.6±5086.19041	60.72969±3.04274	511 180.8±4551.3433
UMDAg	54.44±4.46487	34 460.42±2001.8494	50.14257±3.50006	248 321.1±84 113.16713
pEDAInd—UMDAvv—10%	• 60.05±3.32546	• 35 635.2±1889.01253	• 60.645±3.52038	300 646.4±33 534.86336
pEDAMod(0.9)—UMDA	• 60.74744±2.79943	37 862.4±4524.30108	58.59561±2.65642	293 683.2±43 809.57967
pEDAMod(Adpt)—UMDA	• −61.0076 ± 3.12438	• 36 454.4±3471.78256	58.35±3.54022	282 828.8±12 565.25313
EMNA	58.87835±2.07826	7971.72±14 591.88229	61.71569±1.16702	27 143.56±9453.81368
pEDAInd—EMNA—10%	39.66867±1.11555	1280±0	• 62.9435±1.63767	275 251.2±206 761.70032
pEDAMod(0.9)—EMNA	39.95651±1.36175	1280±0	• 62.51193±1.30228	• 179 404.8±180 344.12323
pEDAMod(Adpt)—EMNA	39.66872±0.98991	1280±0	62.0844±1.27567	204 185.6±222 050.05352

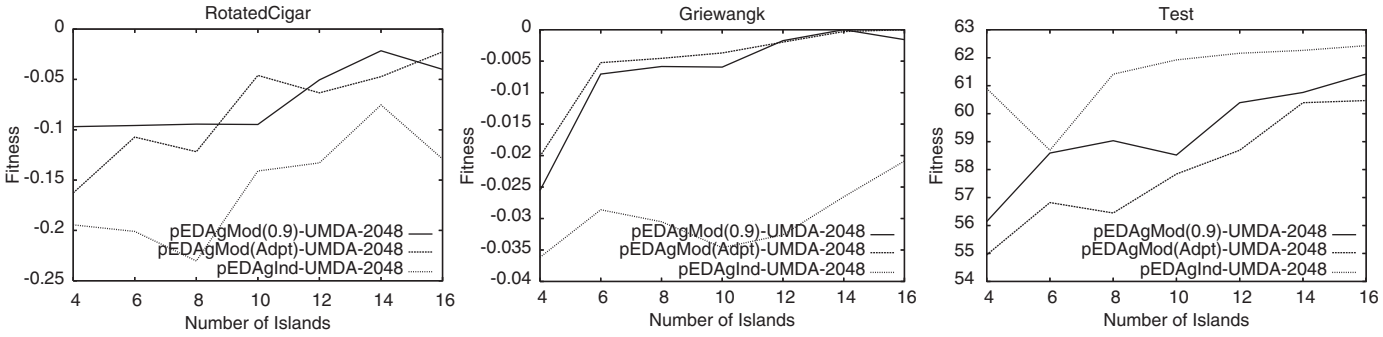

Fig. 3. Fitness with respect to number of islands in algorithms with population 2048.

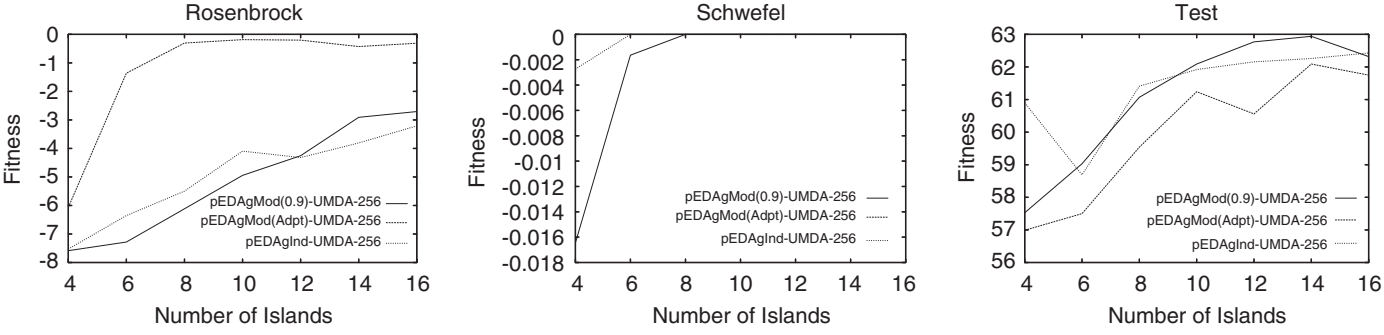

Fig. 4. Fitness with respect to number of islands in algorithms with population 256.

but only taking into account those algorithms previously selected by using fitness. Therefore, those algorithms marked with • in the number of evaluations are those which were found *non-different* with respect to the algorithm having best average fitness and also *non-different* with respect to the algorithm carrying out fewer evaluations (on average). Cells containing the reference algorithm (and those with the same mean) have been marked in italics.

3. Finally, we made a similar comparison between the best algorithm(s) that use $UMDA_g$ and the best one(s) using $EMNA_g$. If any of them outperformed the other(s) then their cell(s) are marked in bold.

From the results of the table and the comparisons carried out we have observed that concerning univariate models, it can be seen that islands always outperform sequential versions. Moreover, except in the *RotatedCigar* problem, the results obtained by these algorithms are always much better when running with a population of 256 individuals. We could expect that this is due to the fact that they need enough generations to evolve through, and a bigger population reaches the stop condition earlier. However, in most cases the number of evaluations is lower than 512 000, which means that apart from generations to evolve, algorithms based on $UMDA_g$ can benefit from small population islands.

For a deeper insight within this phenomenon, we ran another experiment, in which the number of islands changed from 4 to 16. Fig. 3 shows the results where we can appreciate that, although population is the highest we have tested (2048) and the maximum number of generations is the same (512 000), the benefits of augmenting the number of islands are clear in all problems without exception. Moreover, in some problems such as *Rosenbrock* or *Schwefel* this trend can even be observed when using a population of size 256 (Fig. 4).

A possible explanation for this could be that since $UMDA_g$ only computes marginal probabilities, the mean value of a given variable will evolve towards the region with more frequent good values. A different region (for this variable) where the probability of reaching good values is very small (although they can be better) will not be reached even if a few individuals taking values in such a region appear throughout the search. However, if these few individuals appear in an island where the population is smaller, their influence when computing the probabilistic model is bigger and can lead the mean of the variable to the right region. Furthermore, this information is shared with the rest of the islands through communication.

On the other hand, concerning $EMNA_{\mathrm{GLOBAL}}$-based algorithms, it is clear that they need a larger population to determine the model parameters. To get more insight into this situation we have conducted a new experiment in which different population sizes have been tried. As can be seen in Fig. 5, the increase in population leads to better performance, although there is a point at which this trend stops.

The other key point of our analysis is the comparison between migration of models and migration of individuals. As can be seen in Table 1, with $UMDA_g$, the migration of models

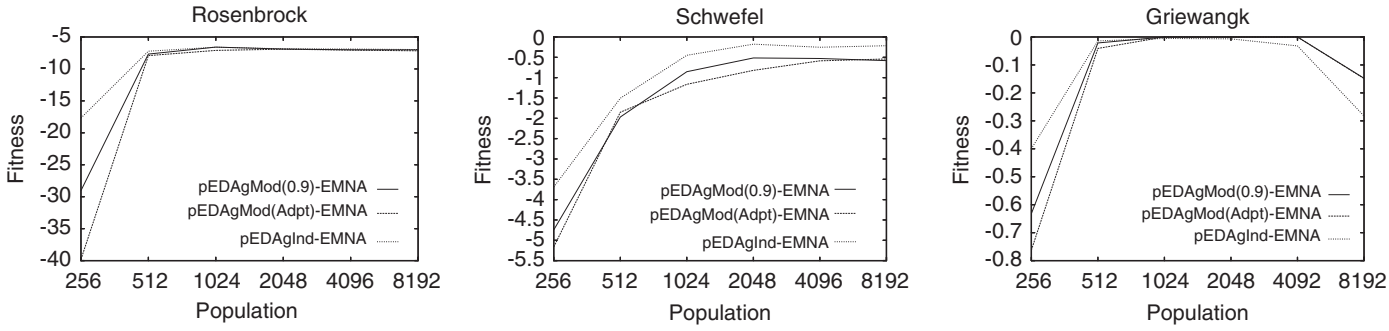

Fig. 5. Population vs fitness within island algorithms that use EMNA.

Table 2
Results when increasing the complexity of the problems (population sizes 256 and 2048)

Population 256			Population 2048	
Summation Cancellation (50)				
UMDAg	0.43836±0.05086	9108.22±6185.10313	0.51231±0.05769	94 625.68±52 185.1973
pEDAInd—UMDA—10%	2.28311±0.77034	937 318.4±84 999.22861	2.88527±0.73994	1 022 361.6±4839.2352
pEDAMod(0.9)—UMDA	• 4.46237±1.39961	• 1 023 744±739.00834	1.06616±0.16696	831 078.4±292 134.47599
pEDAMod(Adpt)—UMDA	• 5.27216±1.46708	• 1 023 846.4±424.52797	1.08491 ± 0.17086	820 428.8±284 199.51197
Rosenbrock (50)				
UMDAg	−47.58985 ± 0.07711	69 043.62±2411.24206	−47.57181 ± 0.02161	556 667.48±10 868.52866
pEDAInd—UMDA—10%	−44.2741 ± 0.13457	1 024 000±0	−46.72235 ± 0.09526	1 024 000±0
pEDAMod(0.9)—UMDA	−45.4253 ± 0.55201	1 024 000±0	−47.38645 ± 0.07439	1 024 000±0
pEDAMod(Adpt)—UMDA	• −30.73386 ± 0.4261	• 1 024 000±0	−44.3419 ± 0.2754	1 024 000±0
Schwefel (50)				
UMDAg	−0.23512 ± 0.22688	57 073.92±2020.34504	−0.28175 ± 0.10116	460 617.52±11 436.63328
pEDAInd—UMDA—10%	• 0±0	588 134.4±59 284.54647	−0.04008 ±0.01589	1 024 000±0
pEDAMod(0.9)—UMDA	• −0.00002 ± 0.00005	691 046.4±221 984.16991	−0.0701 ± 0.04207	1 024 000±0
pEDAMod(Adpt)—UMDA	• 0±0	• 298 496±16 029.83884	−0.0039 ± 0.00526	1 024 000±0
Griewangk (50)				
UMDAg	−0.23512 ± 0.22688	57 073.92±2020.34504	−0.28175 ± 0.10116	460 617.52±11 436.63328
pEDAInd—UMDA—10%	• 0±0	588 134.4±59 284.54647	−0.04008 ± 0.01589	1 024 000±0
pEDAMod(0.9)—UMDA	• −0.00002 ± 0.00005	691 046.4±221 984.16991	−0.0701 ± 0.04207	1 024 000±0
pEDAMod(Adpt)—UMDA	• 0±0	• 298 496±16 029.83884	−0.0039 ± 0.00526	1 024 000±0
Griewangk (50)				
UMDAg	• 0±0	• 49 528.54± 453.96688	• 0±0	393 089.4±1674.50968
pEDAInd—UMDA—10%	• 0±0	• 49 254.4±652.6745	• 0±0	374 374.4±5187.81752
pEDAMod(0.9)—UMDA	• 0±0	52 531.2±449.52123	• 0±0	398 950.4±2048
pEDAMod(Adpt)—UMDA	• 0±0	51 865.6± 652.6745	• 0±0	396 902.4± 4463.51252
RotatedCigar (50)				
UMDAg	−2.95162 ± 2.21312	152 510.24±25 021.95314	−2.64521 ± 0.951	1 020 572.42±2360.17203
pEDAInd—UMDA—10%	−2.04732 ± 1.55568	1 023 897.6±512	−2.15689 ± 0.87874	1 024 000±0
pEDAMod(0.9)—UMDA	• −1.51311 ± 1.47151	• 985 497.6±192 245.50349	• −1.40967 ± 0.69845	• 1 024 000±0
pEDAMod(Adpt)—UMDA	−3.13416 ± 3.43959	1 023 948.8±256	• −1.34577 ± 1.03632	• 1 016 217.6±50 698.50066
Test (100)				
UMDAg	251.65733±35.64567	82 438.32±41 644.68788	194.80968±25.73027	150 442.08±308 816.34329
pEDAInd—UMDA—10%	• 284.95 ± 8.32103	• 121 702.4±5934.19559	• 283.93 ±8.56	896 614.4±28 055.85591
pEDAMod(0.9)—UMDA	• 284.1±6.94022	170 547.2±26 417.04177	272.37±6.38444	878 182.4±27 137.61002
pEDAMod(Adpt)—UMDA	• 283.08±7.0989	• 121 600±3544.15951	268.8±8.58809	849 100.8±19 590.2735

that use adaptive β is the best island algorithm (or is included in best group) in four of the six problems. Moreover, in the *Schwefel* problem, where the best is the migration of models with adaptive β, the result of this scheme is far better than the others. Algorithms that migrate individuals are the best in two problems, although it is worth mentioning that the difference with model migration is not as great as it is in the remaining cases.

$EMNA_{\mathrm{GLOBAL}}$-based algorithms behave differently since the best kind of migration hardly depends on the problem. Thus, the migration of individuals is the best option in two problems; the migration of models with $\beta = 0.9$ is the best option in one

case; and migration of models with adaptive β is the best option in the rest. When using small populations we have obtained worse results mixing models than migrating individuals. It is clear that learning a multivariate model from a small population reduces the confidence in the learnt model and therefore in its combinations. On the other hand, in the migration of individuals scheme if the received individuals are bad they can be discarded very quickly by the resident island.

In general, it could be said that there is no clear trend in $EMNA_{\text{GLOBAL}}$-based algorithms, while migration of models with adaptive β performs better in the univariate models-based island algorithms.

5.2.2. Increasing the problem complexity

In order to test the robustness of the proposed models with respect to the problem complexity, we have repeated the experiments for instances of size $n = 50$ ($n = 100$ for *Test*). As the complexity increases so much we have duplicated the maximum number of allowed evaluations (1 024 000). For the reasons previously explained in this experiment, we have considered only the univariate case. The results are shown in Table 5.2.1.

The algorithms have been compared by using the methodology previously described. In all cases, but *RotatedCigar* the use of a small population leads to better solutions. Moreover, the number of evaluations required to reach these solutions is considerably smaller.

Concerning the kind of migration the differences are not so clear in some cases. However, in general migration of models with adaptive β estimation seems to be the best option, because in all problems it is either the best or similar to the best. Moreover, in two of the problems where this scheme outperforms the others (*Rosenbrock* and *Schwefel*) the differences are fairly substantial.

6. Conclusions

In this work, the problem of applying migration-of-models-based parallel EDAs to continuous problems has been addressed. The approach can be viewed as a general framework which can be instantiated for different algorithms. The experiments carried out here show that islands-based parallelization is very beneficial when applied to univariate EDAs since it helps, with little computational effort, to overcome some of the shortcomings that sequential versions present. In general, when using a small population they offer a good tradeoff between quality of solution and (low) number of required evaluations. However, in the case of multivariate algorithms, such as $EMNA_{\text{GLOBAL}}$ the requirements of a huge population to (correctly) determine model parameters causes a decrease in their performance when using multiple demes. This problem is more notable when the complexity (problem size) of the task increases.

The other aim of our work was to compare migration of models against migration of individuals. As the results reflect, migration of models plus mixture-based model combination offers the best performance particularly when adaptive estimation of β is used.

As future work we plan to center our efforts on the improvement of model combination (i.e. by using memetic algorithms or trying to take advantage of the knowledge acquired during the search process). Additionally, we plan to experiment with asynchronous communication and with heterogeneous memes.

Acknowledgments

This work has been partially supported by the Consejería de Ciencia y Tecnología (JCCM) and Consejería de Educación y Ciencia (JCCM) under projects PBC-02-002 and PBI-05-022. We would also like to thank the three reviewers for their suggestions.

References

[1] C.W. Ahn, D. Goldberg, R.S. Ramakhrishna, Multiple-deme parallel estimation of distribution algorithms: basic framework and application, in: Proceedings of Parallel Processing and Applied Mathematics (PPAM 2003), Lecture Notes in Computer Science, vol. 2774, Springer-Verlag, 2004, pp. 544–551.

[2] E. Alba, J.M. Troya, A survey of parallel distributed genetic algorithms, Complexity 4 (1990) 303–346.

[3] E. Alba (Ed.), Parallel Metaheuristics: A New Class of Algorithms, Wiley-Interscience, New York, 2005.

[4] P.A.N. Bosman, D. Thierens, IDEAs bases on the normal kernels probability density function, Technical Report UU-CS-2000-11, Utrech University, 2000.

[5] E. Cantú-Paz, Efficient and Accurate Parallel Genetic Algorithms, Kluwer, Dordrecht, 2001.

[6] L. delaOssa, J.A. Gámez, J.M. Puerta, Migration of probability models instead of individuals: an alternative when applying the island model to edas, in: Proceedings of Parallel Problem Solving from Nature (PPSN VIII), Lecture Notes in Computer Science, Springer-Verlag, New York, NY, 2004, pp. 242–252.

[7] L. delaOssa, J.A. Gámez, J.M. Puerta, Initial approaches to the application of islands-based parallel EDAs in continuous domains, in: Proceedings of the 34th International Conference on Parallel Processing Workshops (ICPP 2005 Workshops), Oslo, 2005, pp. 580–587.

[8] M.J. Flores, J.A. Gámez, J.M. Puerta, UMDA$_t$: a univariate EDA in continuous domains based on the triangular and TSP distributions, in: 10th Information Processing and Management of Uncertainty in Knowledge-Based System, vol. 2, Perugia, 2004, pp. 781–788.

[9] D.E. Goldberg, M. Pelikan, S. Tsutsui, Evolutionary algorithm using marginal histogram models in continuous domains, Technical Report 2001019, IlliGAL. University of Illinois at Urbana-Champaign, 2001.

[10] N. Hansen, A. Ostermaier, Completely derandomized self-adaption in evolutionary strategies, Evol. Comput. 9 (2) (2001) 159–195.

[11] F. Herrera, M. Lozano, J. Verdegay, Tackling real-coded genetic algorithms: operators and tools for the behaviour analysis, Artificial Intelligence Rev. 12 (1998) 265–319.

[12] T. Hiroyasu, M. Miki, M. Sano, H. Shimosaka, S. Tsutsui, J. Dongarra, Distributed probabilistic model-building genetic algorithm, in: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2003), Lecture Notes in Computer Science, vol. 3242, Springer-Verlag, New York, NY, 2003, pp. 1015–1028.

[13] P. Larrañaga, R. Etxeberria, J.A. Lozano, J.M. Peña, Optimization by learning and simulation of Bayesian and Gaussian networks, Technical Report. EHU-KZAA-IK-4-99, University of the Basque Country, 1999.

[14] P. Larrañaga, R. Etxeberria, J.A. Lozano, J.M. Peña, Optimization in continuous domains by learning and simulation of gaussian networks, in: Proceedings of the 2000 Genetic and Evolutionary Computation Conference Workshop Program, Las Vegas, 2000, pp. 201–204.

[15] P. Larrañaga, J.A. Lozano, Estimation of distribution algorithms. A New Tool for Evolutionary Computation, Kluwer, Dordrecht, 2002.
[16] J. Madera, E. Alba, A. Ochoa, A parallel island model for estimation of distribution algorithms, in: J.A. Lozano, P. Larrañaga (Eds.), Towards a New Evolutionary Computation. Advances in the Estimation of Distribution Algorithms, Studies in Fuzziness and Soft Computing, Springer, Verlag, New York, NY, 2005.
[17] H. Mühlenbein, The equation for response to selection and its use for prediction, Evol. Comput. 5 (1998) 303–346.
[18] H. Mühlenbein, D. Schlierkamp-Voosen, Predictive models for the breeder genetic algorithm I. Continuous parameter optimization, Evol. Comput. 1 (1993) 25–49.
[19] J. Ocenasek, Parallel estimation of distribution algorithms, Ph.D. Thesis, Faculty of Information Technology, Brno University of Technology, 2002.
[20] J. Ocenasek, J. Schwartz, M. Pelikan, Design of multithreaded estimation of distribution algorithms, in: Proceedings of Genetic and Evolutionary Computation (GECCO 2003), Lecture Notes in Computer Science, vol. 2723, Springer-Verlag, New York, NY, 2003, pp. 1247–1258.
[21] R. Rubinstein, B. Melamed, Modern Simulation and Modelling, Wiley-Interscience, New York, 1998.
[22] M. Sebag, A. Ducoulombier, Extending population-based incremental learning to continuous search spaces, in: Proceedings of Parallel Problem Solving from Nature—PPSN V, Lecture Notes in Computer Science, vol. 1498, Springer-Verlag, New York, NY, 1998, pp. 418–427.

Luis delaOssa received the M.S. degree in computer science in 2002 from the University of Castilla-La Mancha. In 2003, he joined the Department of Computer Systems of this university as teaching assistant and he is currently doing his Ph.D. His research activity is developed at the Laboratory of Intelligent Systems and Data Mining (SIMD) and is mainly focused on evolutionary algorithms, machine learning, data mining and fuzzy logic.

José A. Gámez received the M.S. degree in computer science in 1991, and the Ph.D. degree in computer science in 1998, both from the University of Granada, Spain. He joined the Department of Computer Science at the University of Castilla-La Mancha (UCLM) in 1991, where he is currently an Associate Professor. He has served as Vice-Dean of the Escuela Politécnica Superior de Albacete (UCLM) from 1998 to 2004 and currently he serves as Chair of the Department of Computer Systems (UCLM). His research activity is developed at the Laboratory of Intelligent Systems and Data Mining (SIMD) and his main research interest include probabilistic reasoning, Bayesian networks, evolutionary algorithms, machine learning and data mining. Dr. Gámez has edited four books and published more than 50 papers over these topics.

José M. Puerta received the M.S. degree in computer science in 1991, and the Ph.D. degree in computer science in 2001, both from the University of Granada, Spain. He joined the Department of Computer Systems at the University of Castilla-La Mancha (UCLM) in 1991, where he is currently an Associate Professor. His research activity is developed at the Laboratory of Intelligent Systems and Data Mining (SIMD) and his main research interest include probabilistic graphical models, Bayesian networks, evolutionary algorithms, machine learning and data mining. Dr. Puerta has published more than 20 papers over these topics.