Hierarchical Distributed Genetic Algorithms*

F. Herrera,[1], †, M. Lozano[1], ‡, C. Moraga[2], §
[1] *Department of Computer Science and Artificial Intelligence, University of Granada, 18071 Granada, Spain*
[2] *Department of Computer Science, University of Dortmund, 44221 Dortmund, Germany*

Genetic algorithm behavior is determined by the exploration/exploitation balance kept throughout the run. When this balance is disproportionate, the premature convergence problem will probably appear, causing a drop in the genetic algorithm's efficacy. One approach presented for dealing with this problem is the distributed genetic algorithm model. Its basic idea is to keep, in parallel, several subpopulations that are processed by genetic algorithms, with each one being independent from the others. Furthermore, a migration operator produces a chromosome exchange between the subpopulations. Making distinctions between the subpopulations of a distributed genetic algorithm by applying genetic algorithms with different configurations, we obtain the so-called heterogeneous distributed genetic algorithms. In this paper, we present a hierarchical model of distributed genetic algorithms in which a higher level distributed genetic algorithm joins different simple distributed genetic algorithms. Furthermore, with the union of the hierarchical structure presented and the idea of the heterogeneous distributed genetic algorithms, we propose a type of heterogeneous hierarchical distributed genetic algorithms, the hierarchical gradual distributed genetic algorithms. Experimental results show that the proposals consistently outperform equivalent sequential genetic algorithms and simple distributed genetic algorithms. © 1999 John Wiley & Sons, Inc.

I. INTRODUCTION

Genetic algorithms (GAs) are general purpose search algorithms that use principles inspired by natural genetic populations to evolve solutions to problems.[1,2] The basic idea is to maintain a population of chromosomes, which

---

* This research has been partially supported by CICYT TIC96-0778 and DGICYT SAB95-0473. Work leading to this Research Report was done while C. Moraga was visiting the University of Granada, Granada, Spain, in 1997, under the Sponsorship of the DGICYT SAB95-0473.

† Author to whom all correspondence should be addressed. e-mail: herrera@decsai.ugr.es
‡ e-mail: lozano@decsai.ugr.es
§ e-mail: moraga@cs.uni-dortmund.de

INTERNATIONAL JOURNAL OF INTELLIGENT SYSTEMS, VOL. 14, 1099–1121 (1999)
CCC 0884-8173/99/111099-23

represent candidate solutions to the concrete problem, that evolves over time through a process of competition and controlled variation. Each chromosome in the population has an associated *fitness* to determine which chromosomes are used to form new ones in the competition process, which is called *selection*. The new ones are created using genetic operators such as *crossover* and *mutation*. GAs have great measure of success in search and optimization problems. They are particularly adequate in large, complex, and poorly understood search spaces, where classical search tools (enumerative, heuristic, etc.) are inappropriate, offering a valid approach to problems requiring efficient and effective search techniques. The following bibliography may be examined for a more detailed discussion about GAs and their applications.[1–6]

The balance between exploration and exploitation or, in other words, between the creation of diversity and its reduction, by focusing on the individuals of higher fitness, is essential in order to achieve a reasonable behavior for GAs in the case of complicated optimization problems.[7] Loss of critical alleles due to selection pressure, selection noise, schemata disruption due to crossover operator, and poor parameter setting may make this exploitation/exploration relationship disproportionate; e.g., the proper balance between a broad search and a sufficient refinement is not established and produce the lack of *diversity* in the population.[8–10] Under these circumstances a preeminent problem appears, *the premature convergence problem*, a premature stagnation of the search caused by a loss of genotypical diversity.[1] Usually the search becomes trapped in a local optimum before the global optimum is found.

Some tools for monitoring the exploration/exploitation relationship have been proposed in order to avoid the premature convergence problem and improve GA performance. These tools include modified selection and crossover operators,[9–12] adaptive techniques,[13] optimization of parameter settings,[14,15] and diversity preservation methods based on *spatial separation*.[16–22]

Distributed GAs (DGAs) are one of the most important representatives of methods based on spatial separation.[21–23] The basic idea of DGAs lies in the partition of the population into several subpopulations (whose sizes are relatively small), each one of them being processed by a GA, independently from the others. Furthermore, an operator, called *migration*, produces a chromosome exchange between the subpopulations. Its principal role is to promote genetic diversity, and in essence, to allow the sharing of possible solutions. DGAs show two determinant advantages: (1) the preservation of the diversity due to the semi-isolation of the subpopulations, which may stand up to the premature convergence problem, and (2) they may be easily implemented on parallel hardware, obtaining, in this way, substantial improvements on computational time. Some authors have highlighted the interest in the distinction between the subpopulations of a DGA by means of the application of GAs with different control parameters, genetic operators, codings, etc.[21,22,24] These types of DGAs are called *heterogeneous DGAs*. A model of these algorithms are the *gradual DGAs* (GDGAs),[25] a class of heterogeneous DGAs based on real coding that applies a different crossover operator to each subpopulation. These operators are differentiated according to their associated exploration and exploitation properties and the degree thereof.

Some DGA models combine the advantages of the DGAs with the ones of other spatial separation methods, such as the cellular GAs,[16–19] reporting good results. The basic idea concerns the organization of each DGA subpopulation as a cellular GA. Instances of these proposals are the mixed parallel GAs[26] for neural network design and training applications, the structured DGAs[27] for function optimization problems, and a DGA model whose subpopulations are based on the ECO framework[17] that was designed and used in Ref. 28 for different problems, such as quadratic assignment problem, dynamic control, and parametric optimization.

In this paper we present a proposal of DGAs, the *hierarchical DGAs* (HDGAs), DGAs whose nodes are other simple DGAs, which are connected with each other. The existence of two types of migrations, the local ones, in every DGA, and the global ones, between the constituent DGAs, is a key feature of HDGAs, since it establishes the proper hierarchy between the simple DGAs and the HDGA. An important feature of the hierarchical structure to be presented is that it may be easily joined with the idea of heterogeneous DGAs. A possible way is the design of heterogeneous HDGAs, which use different basic homogeneous DGAs. In this way, we propose the hierarchical GDGAs, GDGAs that are composed by different homogeneous DGAs whose subpopulations use the same crossover operator, following these crossover operators the underlying idea in the GDGAs.

In order to develop the HDGAs, this paper is set out as follows. In Section II, we describe the DGAs. In Section III, we present the HDGAs. In Section IV, a type of heterogeneous HDGAs based on homogeneous DGAs, the hierarchical GDGAs, are proposed. In Section V, the experiments carried out for determining the efficacy of the algorithms presented are described. And, some concluding remarks are dealt with in Section VI.

II. DISTRIBUTED GENETIC ALGORITHMS

This section is devoted to DGAs. In Subsection A, they shall be presented as a class of parallel GAs called *coarse grained parallel GAs*. In Subsection B, spatial separation, a basic principle of DGAs, is justified from a biological point of view through the *Shifting Balance Theory of Evolution*[29] and the *Theory of Punctuated Equilibria*.[30] In Subsection C, we describe the basic structure of DGAs. And finally in Subsection D, we review the types of DGAs presented previously.

A. Parallel Genetic Algorithms

The availability, over the last few years, of fast and cheap parallel hardware has favored research into possible ways for implementing parallel versions of GAs. GAs are good candidates for effective parallelization, since they are inspired on the principles of evolution, in parallel, for a population of individuals.[24] In general, three ways were followed for implementing the parallelization of GAs.[24,31–34]:

- **Global parallelization**. The evaluation of chromosome fitness and sometimes the genetic operator application are carried out in a parallel form.[35–37]
- **Coarse grained parallelization**. The population is divided into small subpopulations that are assigned to different processors. Each subpopulation evolves independently and simultaneously according to a GA. Periodically, a *migration operator* exchanges individuals between subpopulations, allowing new diversity to be injected into converging subpopulations. The exchange generally takes the form of copying individuals between the populations. GAs based on coarse grained parallel GAs are known as *distributed GAs*, since they are usually implemented in distributed memory MIMD computers. Versions of DGAs appeared in Refs. 20–23 and 38–43.
- **Fine grained parallelization**. In this model, the population is divided into a great number of small subpopulations. Usually, a unique individual is assigned to each processor. The selection mechanism and the crossover operator are applied by considering neighboring chromosomes. For example, every chromosome selects the best neighbor for recombination and the resultant individual shall replace it. These types of GAs, known as *cellular GAs*, are usually implemented on massively parallel computers. Examples of cellular GAs are to be found in Refs. 16–19.

B. Spatial Separation

Both distributed GAs and cellular GAs are instances of models based on spatial separation. One of the main advantages of these models is the preservation of diversity. This property caused them to be considered as an important way to research into mechanisms for dealing with the premature convergence problem.[9,16,17,22,31,34,38–40]

Many authors[16,17,19–21] have attempted to justify spatial separation models, starting from the *Shifting Balance Theory of Evolution*, developed by Wright.[29] This theory explains the process of evolution on the genetic composition of individuals in natural populations. According to this, large populations of organisms rarely act as a single well-mixed (panmictic) population, but rather they consist of semi-isolated subpopulations, *demes*, each of which is relatively small in size. Furthermore, the demes communicate with each other through migrations of individuals. For Wright, the evolution process has two phases. During the first one, the allele frequencies drift randomly around a local fitness peak in each deme. One of them might, by chance, drift into a set of gene frequencies that correspond to a higher peak. Then, the second phase starts; this deme produces an excess of offspring, due to its high average fitness, which then emigrate to the other demes, and will tend to displace them until eventually the whole population has the new favorable gene combination. Then, the process starts again. The relatively small size of the demes allows drift to play an important role in the evolution of the population, without driving the whole population towards convergence. Even if drift were to drive every local deme to fixation, each one of them would be fixed on a different genotype, thereby maintaining diversity in the population as a whole.

Another biological theory adopted by people who do work on spatial separation is the theory of Punctuated Equilibria.[30] This theory states that evolution is characterized by long periods of relative stasis, punctuated by periods of rapid change associated with specialization events. In Ref. 39, it is pointed out that GAs also tend towards stasis, or premature convergence, and that isolated species could be formed by separating the global population into subpopulations. By injecting an individual from a different species into a subpopulation after it had converged, new building blocks would become available; furthermore, immigrants would effectively change the fitness landscape in the subpopulations. In this way, premature convergence may be avoided. In this line, in Ref. 20 it was stated that:

> “Subpopulations isolated for a certain time keep the diversity of the population high. After migration new promising areas can be discovered by crossing over.”

which explains that:[20]

> “The creative forces of evolution take place at migration and few generations afterwards. Wright’s argument that better peaks are found just by chance in small subpopulations does not capture the essential facts.”

C. Basic Structure of Distributed GAs

Although there are a lot of different types of DGAs, all of them are variations on the following basic algorithm:

```
• Distributed Genetic Algorithm
  1. Generate at random a population, $P$, of chromosomes.
  2. Divide $P$ into $SP_1, \ldots, SP_{N_S}$ subpopulations.
  3. Define a neighborhood structure for $SP_i$, $i = 1, \ldots, N_S$.
  4. For $SP_i$, $i = 1, \ldots, N_S$, execute in parallel the next steps:
     4.1 Apply, during $f_m$ generations, the selection mechanism and the genetic
         operators.
     4.2 Send $n_m$ chromosomes to neighboring subpopulations.
     4.3 Receive chromosomes from neighboring subpopulations.
  5. If the stop criterion is not fulfilled, return to 4.
```

Some additional issues should be considered:

1. The neighborhood structure, *topology*, to be used. Generally, hypercubic topologies were considered for exploiting the power of DGAs.[21–23,39,41,42]
2. The *migration rate*, n_m, that controls how many chromosomes migrate.
3. The *migration interval*, f_m, the number of generations between each migration.
4. The *selection strategy* of the genetic material to be copied. Two methods were widely used. The first one is to select randomly the element from current subpopulation. The advantage of this approach is the greater mix of genes that will result. A second method is to select the highest performing individual from each subpopulation to be copied to another subpopulation. This would result in more directed evolution than the first case, as the migrant individuals would not be tainted by genes from lower performing individuals. This is not to say that the former method is worse, for the less directed a population is, the greater diversity it will contain.[44]

5. The *replacement strategy* for including the chromosomes to be received. Some approaches are, replace the worst ones, the most similar to the incoming ones, one randomly chosen, etc.
6. The choice of *whether or not to replicate migrating individuals*; i.e., should individuals move to their new home or should a copy of them be sent there? If one does not copy individuals it is possible that a subpopulation could be set back several generations in evolutionary terms by the mass emigration of its best performers. On the other hand, simply copying individuals across could lead to highly fit individuals dominating several populations.[44]

D. Types of Distributed GAs

In Ref. 34, the following three categorizations of DGAs are reported:

- **Regarding the migration method**
    - *Isolated DGAs*. There are no migrations between subpopulations. This model is known as well as *partitioned GAs*.[22,42]
    - *Synchronous DGAs*. Migrations between subpopulations are synchronized, i.e., they are produced at the same time.[21-23]
    - *Asynchronous DGAs*. Migrations are produced when certain events appear, related with the activity of each subpopulation. Asynchronous behavior is typically found in nature, since evolution is produced at different states depending on the environment.[34]
- **Regarding the connection schema**
    - *Static connection scheme*. The connections between the subpopulations are established at the beginning of the run, and they are not modified throughout it.
    - *Dynamic connection scheme*. The connection topology is dynamically changed throughout the run. The reconfigurations in these connections may occur depending on the evolution state of the subpopulations. For example, in Ref. 34, a connection schema called positive-distance topology was proposed in which an individual is passed to another subpopulation only if the Hamming distance between the best individuals in the two subpopulations is less than 24. An analogous connection schema called negative-distance topology was presented as well.

Finally, we point out that some authors[32,45] assumed another division, based on the connection schema: the *island* model and the *stepping stone* model. In the first model, individuals can migrate to any other subpopulation; in the second model, migration is restricted to neighboring subpopulations.

- **Regarding the subpopulation homogeneity**
    - *Homogeneous DGAs*. Every subpopulation uses the same genetic operators, control parameter values, fitness function, coding schema, etc. Most DGAs proposed in the literature are homogeneous.[21-23] Their principal advantage is that they are easily implemented.
    - *Heterogeneous DGAs*. The subpopulations are processed using GAs with either different control parameter values, genetic operators, coding schema, etc. Some interesting heterogeneous DGAs are the models of adaptation by competing subpopulations,[46,47] the GAs based on migration and artificial selection (GAMAS),[10] the injection island GAs (iiGAs),[34] and the GDGAs.[25]

III. HIERARCHICAL DISTRIBUTED GENETIC ALGORITHMS

In this section we present *HDGAs*. The main idea of these algorithms is to connect DGAs with other DGAs, building, in this way, DGAs whose nodes (subpopulations) are simple DGAs which shall be called *basic DGAs*. For this purpose we have chosen the *Cube-Connected Cycles* structure,[48] which has been used until now mostly for parallel processing in multiprocessor architectures; but is quite appropriate to support migrations within a hierarchical approach. Figure 1 shows an example of HDGA, where the basic DGAs are rings with three subpopulations. It may be observed that every subpopulation in a basic DGA is connected with another subpopulation in an adjacent basic DGA.

Two type of migrations are produced in a HDGA: (1) *local migrations*, produced between subpopulations belonging to the same basic DGAs, and (2) *global migrations*, produced between subpopulations belonging to different basic DGAs that connect these ones. The existence of these two different migrations is the key feature of HDGAs, since they establish the real hierarchy between the basic DGAs and the HDGA.

With the HDGA model we attempt to improve the behavior of traditional DGAs in the following ways:

- Better performance for each node. In the HDGAs, every node is a DGA instead of a simple subpopulation. In this way, we raise the efficiency of these nodes.
- The search may be carried out in different levels. Every basic DGA develops an independent search and collaborates, through the global migrations, with others basic DGAs on a global search of the HDGA.

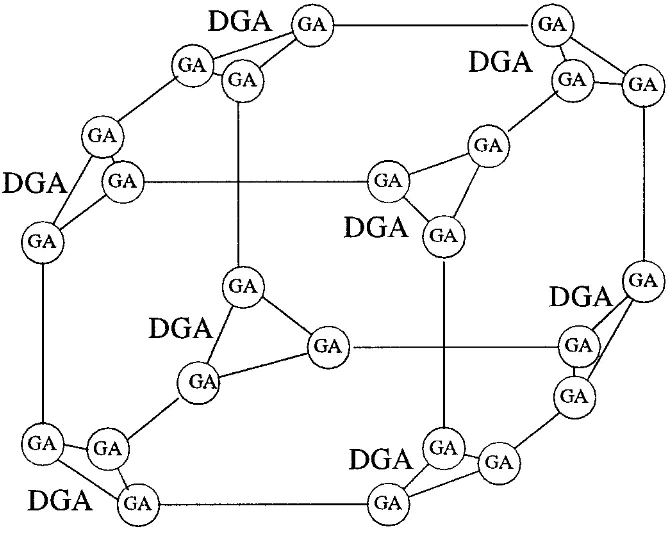

Figure 1. Example of HDGA.

- The good features of the homogeneous and heterogeneous DGAs may be easily joined in two ways:
  - homogeneous HDGAs with heterogeneous basic DGAs or
  - heterogeneous HDGAs with homogeneous basic DGAs.

Finally, we should point out that the HDGAs proposed may be extended using more levels. In this faction, each basic DGA may be, in its turn, a DGA of DGAs, i.e., an HDGA.

IV. HETEROGENEOUS HDGAS BASED ON HOMOGENEOUS BASIC DGAS

In this section we design heterogeneous HDGA based on homogeneous basic DGAs. In order to do so, we use the GDGAs presented in Ref. 25. In Subsection A, we describe the structure of GDGAs. In Subsection B, we build a particular model of them. And finally in Subsection C, we present the heterogeneous HDGAs based on the gradual structure which shall be called hierarchical GDGAs.

A. Gradual Distributed Genetic Algorithms

Heterogenous DGAs have been considered as suitable tools for avoiding the premature convergence problem and for maximizing the exploration and exploitation on the search space. For example, in Ref. 22, Tanese suggested:

> “The distributed genetic algorithm enables different subpopulations to run with different crossover and mutation rates, maintaining a balance between exploration and exploitation in a novel way.”

Gradual DGAs (GDGAs)[25] are a class of heterogeneous DGAs based on real coding in which subpopulations are distinguished by applying crossover operators with different degrees of exploration or exploitation. So, a parallel *multiresolution* is obtained with regard to the crossover operator, which allows a spread search (*reliability*) along with an effective local tuning (*accuracy*) to be simultaneously achieved. Furthermore, subpopulations are adequately connected for exploiting the multiresolution in a gradual way, offering the refinement or the expansion of the best zones emerging. The migrations between subpopulations belonging to different categories produce these final effects.

Figure 2 outlines the basic structure of the GDGAs. They are based on a hypercube topology with three dimensions with two important sides to be differentiated.

- The *front side* is devoted to the exploration. It is made up of four subpopulations, $E_1, \ldots, E_4$, to which exploratory crossover operators are applied. The exploration degree increases clockwise, starting at the lowest, E_1, and ending at the highest, E_4.
- The *rear side* is for exploitation. It is composed of subpopulations $e_1, \ldots, e_4$, that undergo exploitative crossover operators. The exploitation degree increases clockwise, starting at the lowest, e_1, and finishing at the highest e_4.

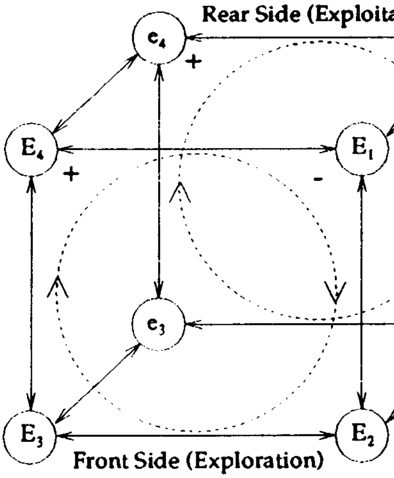

Figure 2. GDGA topology.

The short diameter of the cubic topology may allow multiresolution together with migrations to produce an effective exploration/exploitation balance, since genetic material is quickly exchanged between subpopulations with a wide spectrum of properties and exploration and exploitation degrees, producing a spread search along with an effective local tuning.

In order to allow the refinement and expansion to be carried through to a suitable conclusion, and overcome two possible problems that may appear due to the crossover configuration of GDGAs, the *conquest*[28,34,49] and the *noneffect*[25] problems, the authors proposed to use the following migration schema and selection mechanism:

- **Migration schema**. GDGAs use emigration[50]; individuals leave their subpopulation and migrate to exactly one of the neighboring subpopulations. In particular, every five generations, the best element of each subpopulation is sent towards the corresponding subpopulation, as shown in Figure 3. The sequence of application is from left to right; i.e., first the refinement migrations; second, the refinement/expansion migrations; third, the expansion migrations; and then, the sequence starts again. The place of an emigrant is taken by an immigrant.
- **Selection mechanism**. The selection mechanism of GDGAs is the *linear ranking selection*[51]; the chromosomes are sorted in order of raw fitness, and then the selection probability of each chromosome is computed according to its rank by using a nonincreasing assignment function. Every individual receives an expected number of copies that depends on its rank, independent of the magnitude of its fitness. The selective pressure of this selection mechanism is determined by a parameter, $\eta_{\min} \in [0,1]$, which specifies the expected number of copies for the worst chromosome; the best one has $2 - \eta_{\min}$ expected copies. If $\eta_{\min}$ is low, high pressure is achieved, whereas if it is high, the pressure is low. Different $\eta_{\min}$ values were assigned to the subpopulations, such as shown in Table I.

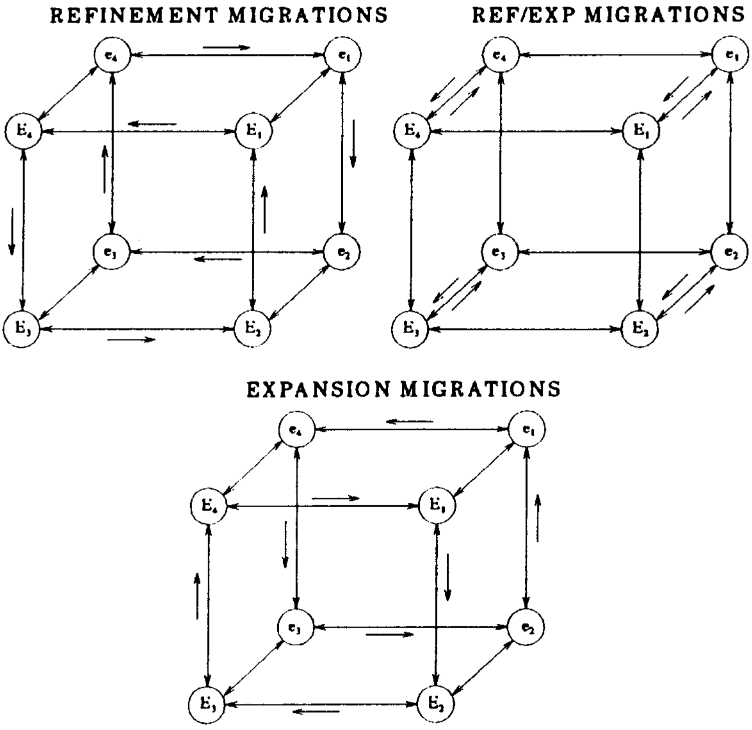

Figure 3. The three types of migration in a GDGA.

B. Gradual DGA Based on FCB-Crossovers

In Ref. 25, some versions of GDGAs were built based on different crossover operators for real-coded GAs that allow different exploration or exploitation degrees to be obtained. In this work, we use only one of these crossover operators, the *Fuzzy Connectives-Based Crossovers* (FCB-crossovers).[12] Next, we comment on some of the main features for these crossover operators and describe how they were used for building a GDGA.

Let us assume that $X = (x_1 \ldots x_n)$ and $Y = (y_1 \ldots y_n)$ $(x_i, y_i \in [a_i, b_i] \subseteq \Re$, $i = 1 \ldots n)$ are two real-coded chromosomes that have been selected to apply the crossover operator to them. In short, the action interval of the genes x_i and y_i $[a_i, b_i,]$ may be divided into three intervals, $[a_i, x_i]$ $[x_i, y_i]$, and $[y_i, b_i]$, that

Table I. $\eta_{\min}$ values for each subpopulation.

	Exploitation				Exploration		
+	←		−	−	→		+
e_4	e_3	e_2	e_1	E_1	E_2	E_3	E_4
0.9	0.7	0.5	0.1	0.9	0.7	0.5	0.1

bound three regions to which the resultant genes of some combination of x_i and y_i may belong. These intervals could be classified as exploration or exploitation zones. The interval with both genes being the extremes is an exploitation zone; the two remaining intervals are exploration zones.

With regards to these intervals, in Ref. 12, three monotone and nondecreasing functions, F, S, and M, are proposed, defined from $[a, b] \times [a, b]$ into $[a, b]$, $a, b \in \Re$, and which fulfill:

$$\forall c, c' \in [a, b], \quad F(c, c') \leq \min\{c, c'\}, \quad S(c, c') \geq \max\{c, c'\} \quad \text{and}$$
$$\min\{c, c'\} \leq M(c, c') \leq \max\{c, c'\}.$$

Each one of these functions allows us to combine two genes giving results belonging to each one of the aforementioned intervals. Now, if $Q \in \{F, S, M\}$, we may generate the offspring $Z = (z_1 \ldots z_n)$ as

$$z_i = Q(x_i, y_i), \qquad i = 1 \ldots n.$$

This crossover operator applies the same F, S, or M function for all the genes in the chromosomes to crossover. For this reason, they were called F-crossover, S-crossover, and M-crossover, respectively. Moreover, they have different properties: The F- and S-crossover operators show exploration and the M-crossover operators show exploitation. Four families of F-, S-, and M-crossover operators, Logical, Hamacher, Algebraic, and Einstein, were presented in Ref. 12. Their effects, along with their associated exploration or exploitation degrees, may be observed in Figure 4.

We have considered that the maximum degree of exploitation is for the logical M-crossover, which is based on the arithmetical mean since it uses the maximum level of information from both genes; i.e., it is not biased toward either of them.

Using the F-, S-, and M-crossover operators, a GDGA called GDGA-FCB was implemented with the configuration show in Table II.

C. Hierarchical Gradual DGAs

We may obtain heterogeneous HDGAs based on homogeneous basic DGAs by assigning to every node of a GDGA a homogeneous DGA whose subpopulations use the same crossover operator and $\eta_{\min}$ values of the corresponding

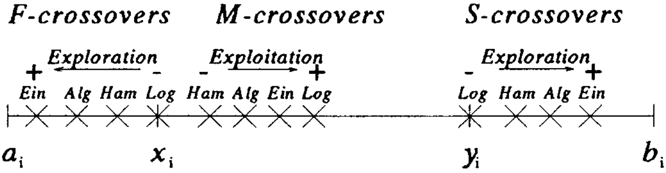

Figure 4. FCB-crossover operators.

Table II. Crossover configuration for GDGA-FCB.

Subpopulations	M-crossover	Subpopulations	F- and S-crossover
e_1	Hamacher	E_1	Logical
e_2	Algebraic	E_2	Hamacher
e_3	Einstein	E_3	Algebraic
e_4	Logical	E_4	Einstein

node of the GDGA (see Tables I and II). These HDGAs shall be called *hierarchical GDGAs*. Figure 5 shows an example of these algorithms where each basic DGA is a ring with three subpopulations. E_i^j with $j = 1, \ldots, 3$ and e_i^j with $j = 1, \ldots, 3$ use the same crossover operator and selective pressure than E_i and e_i in a GDGA, respectively.

With regards to the migration schema of the hierarchical GDGAs, we propose the following:

1. The global migrations, produced between subpopulations belonging to different basic DGAs, are similar to the migration schema of a GDGA. They occur every 10 generations in the sequence of application that is shown in Figure 3, i.e., first, the refinement migrations, second, the refinement/expansion migrations, third, the expansion migrations, and then, the sequence starts again.
2. The local migrations, produced between the subpopulations in the same basic DGA, happen 5 generations after each global migration. These migrations are: $E_i^1 \to E_i^2$, $E_i^2 \to E_i^3$ and $E_i^3 \to E_i^1$, $i = 1, \ldots, 4$, and $e_i^1 \to e_i^2$, $e_i^2 \to e_i^3$ and $e_i^3 \to e_i^1$, $i = 1, \ldots, 4$.

Starting from GDGA-FCB we built a hierarchical GDGA called H-GDGA-FCB.

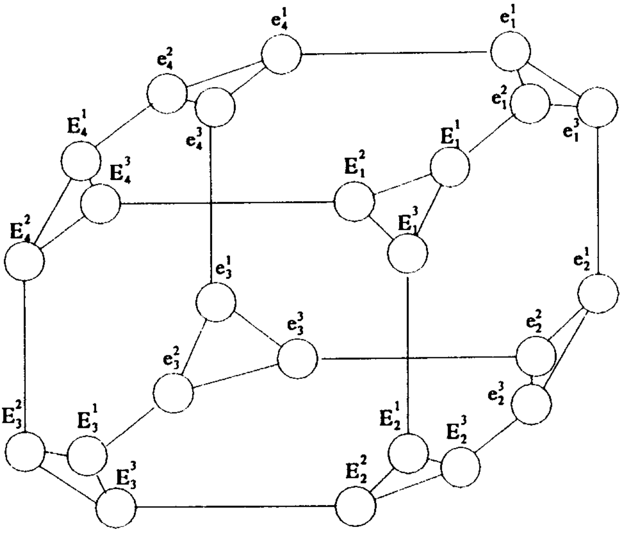

Figure 5. Hierarchical GDGA.

V. EXPERIMENTS

Minimization experiments on the test suite, described below, were carried out in order to study the behavior of the hierarchical GDGA proposed. It was compared with equivalent sequential versions and other implementations of homogeneous DGAs and homogenous HDGAs, in which all the subpopulations have the same configuration.

In Section V A, we describe the test functions considered. And in Section V B, we study the results obtained in the experiments.

A. The Test Suite

For the experiments, we have considered three very complex test functions used in the GA literature: the Generalized Rosenbrock's function,[52,53] the Griewangk's function,[54] and the Expansion of F10.[55] The dimension of the search space is 25.

- *Generalized Rosenbrock's function.*

$$f_{\mathrm{Ros}}(\vec{x}) = \sum_{i=1}^{n-1} \left(100 \cdot \left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2\right),$$

  with $-5.12 \le x_i \le 5.12$ and $f^*_{\mathrm{Ros}} = f_{\mathrm{Ros}}(1, \ldots, 1) = 0$.
  f_{Ros} is a continuous and unimodal function, with the optimum located in a steep parabolic valley with a flat bottom. This feature will probably cause slow progress in many algorithms since they must permanently change their search direction to reach the optimum. This function has been considered by some authors to be a real challenge for any continuous function optimization program.[46] A great part of its difficulty lies in the fact that there are nonlinear interactions between the variables, i.e., it is *nonseparable*.[56]
- *Griewangk's function.*

$$f_{\mathrm{Gri}}(\vec{x}) = \frac{1}{d} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1,$$

  with $d = 4000$, $-600.0 \le x_i \le 600.0$ and $f^*_{\mathrm{Gri}} = f_{\mathrm{Gri}}(0, \ldots, 0) = 0$.
  f_{Gri} is a continuous and multimodal function. This function is difficult to optimize because it is nonseparable[21] and the search algorithm has to climb a hill to reach the next valley. Nevertheless, one undesirable property exhibited is that it becomes easier as the dimensionality is increased.[56]
- *Expansion of $F10$.*

$$E - F10(x_1, \ldots, x_n) = F10(x_1, x_2) + F10(x_2, x_3) + \cdots + F10(x_{n-1}, x_n) + F10(x_n, x_1),$$

  where $F10(x, y) = (x^2 + y^2)^{0.25} \cdot [\sin^2(50 \cdot (x^2 + y^2)^{0.1}) + 1]$, with $x, y \in (-100, 100]$ and $E - F10(x^*) = 0$.
  F10 is a function that has nonlinear interactions between two variables. Its expanded version, $E{-}F10$, is built in such a way that it induces nonlinear interaction across multiple variables. It is nonseparable as well.

B. Results

Table III shows the set of algorithms executed in order to study the behavior of the HDGAs proposed. We include hierarchical GDGAs, hierarchical homogeneous DGAs, and real-coded GAs (RCGAs).[57] The table row has the name of the algorithm, its type, the number of subpopulations, the size of them, the maximum number of generations, and the strategy of application of the crossover operators. The algorithms based on the ST2 and ST4[12] strategies use FCB-crossover operators, and the ones based on the DST1 and DST2 use dynamic FCB-crossovers.[11] The values of the control parameters of the algorithms were suitably chosen for requiring the same number of evaluations. In the following, we briefly describe the mentioned strategies.

- *The ST2 strategy*. For each pair of chromosomes from the total population that undergo crossover, four offspring are generated, the result of applying two exploratory crossover operators and two exploitative ones to them. The two most promising offsprings of the four substitute their parents in the population.
- *The ST4 strategy*. For each pair of chromosomes from a total of $\frac{1}{2} \cdot p_c \cdot N$ (p_c is the crossover probability and N is the population size), four offspring are generated, the result of applying two exploratory crossover operators, an exploitative one and an operator with "relaxed" exploitation, which puts together the two properties. All four offspring will form part of the population in such a way that two of them substitute their parents and the other two substitute two chromosomes belonging to the remaining $\frac{1}{2}$ of the population that should undergo crossover.
- *The DST1 and DST2 strategies*. DST1 and DST2 are similar to ST2 and ST4, respectively. They use dynamic FCB-crossover operators, which keep a suitable sequence between the exploration and the exploitation along the GA run: "*to protect the exploration in the initial stages and the exploitation later*."

Table III. Algorithms.

Algorithms	Type	Subs.	Size	Gens.	Strategy
H-GDGA-FCB	Hier. GDGA	24	15	5000	
H-DGA-S4-Log,...	Hier. Hom DGAs	24	15	5000	ST4
GDGA-FCB	GDGA	8	45	5000	
DGA-S4-Log,...	Hom. DGAs	8	45	5000	ST4
RGA-S2-180-Log,...	Seq. RCGAs	1	180	5000	ST2
RGA-DS1-180-Dub	Seq. RCGA	1	180	5000	DST1
RGA-S2-60-Log,...	Seq. RCGAs	1	60	15000	ST2
RGA-DS1-60-Dub	Seq. RCGA	1	60	15000	DST1
RGA-S4-360-Log,...	Seq. RCGAs	1	360	5000	ST4
RGA-DS2-360-Dub	Seq. RCGA	1	360	5000	DST2
RGA-S4-120-Log,...	Seq. RCGAs	1	120	15000	ST4
RGA-DS2-120-Dub	Seq. RCGA	1	120	15000	DST2
H-GDGA-FCB-R6	Hier. GDGA	48	8	5000	
H-DGA-S4-Log-R6,...	Hier. Hom. DGAs	48	8	5000	ST4
GDGA-FCB-R6	GDGA	8	48	5000	
DGA-S4-Log-R6,...	Hom. DGAs	8	48	5000	ST4

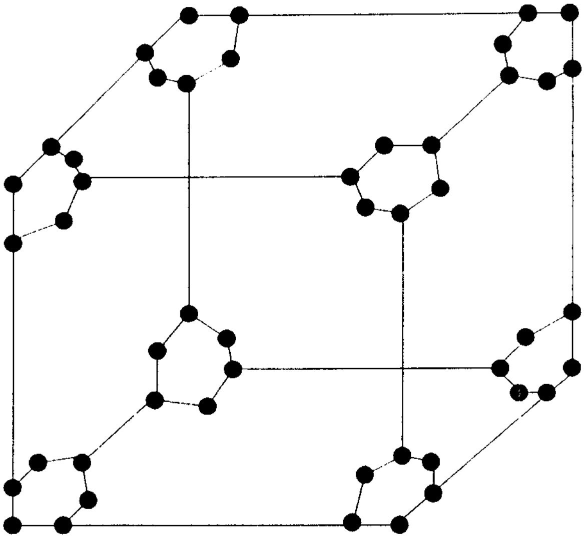

Figure 6. Hierarchical GDGA based on rings with six subpopulations.

For studying the importance of the structure of the basic DGAs in the HDGA behavior, we have implemented a hierarchical GDGA and four hierarchical homogeneous DGAs whose basic DGAs are ring with six subpopulations, such as shown in Figure 6. Their names end with "R6" and their features appear in Table III. All algorithms were executed 15 times.

Table IV shows the average values of the results obtained. For each function, we introduce the medium of the *best* (B) from the last generation, the percentage of *success* (S) with respect to the thresholds shown in Table V, and the final average *online* measure (O)[52] (the average of the fitness of all the elements appearing throughout the GA execution). *Online* is considered here as a population diversity measure.

Next, we point out some important considerations about the results obtained.

About the Hierarchical GDGAs

- The B results of the hierarchical GDGA based on rings with six subpopulations, H-GDGA-FCB-R6, are better than the ones of the hierarchical GDGA based on rings with three subpopulations, H-GDGA-FCB, for all the test functions. Furthermore, we also observe that the S performance of H-GDGA-FCB-R6 is better than the one of H-GDGA-FCB for f_{Ros} and f_{E-F10}, and is equal for f_{Gri}. These results indicate that the hierarchical GDGAs achieve a better behavior when rings with a higher number of subpopulations are introduced.
- We should highlight the good results of H-GDGA-FCB-R6 for the complex f_{Ros}, which has been considered as a very complex test function.[25,46,56] H-GDGA-FCB-R6 obtains the best S value, 93.3, and the second B result, 1.4. Only RGA-S4-120-Ham, a real-coded GA based on the ST4 strategy, returned a better B result,

Table IV. Results.

		f_{Ros}			f_{Gri}			f_{H-F10}	
Algorithms	B	S	O	B	S	O	B	S	O
H-GDGA-FCB	2.8e + 00	86.7	9.2e + 0.4	1.1e − 02	46.7	2.5e + 02	2.6e − 01	0.0	1.2e + 02
H-DGA-S4-Log	2.2e + 01	0.0	8.0e + 02	8.8e − 03	53.3	3.9e + 00	1.4e − 01	0.0	2.2e + 01
H-DGA-S4-Ham	1.9e + 00	6.7	6.5e + 04	2.3e − 01	0.0	2.9e + 0.2	1.7e + 01	0.0	2.4e + 02
H-DGA-S4-Alb	9.2e + 01	0.0	3.5e + 05	1.0e + 00	0.0	6.9e + 02	4.4e + 01	0.0	3.0e + 02
H-DGA-S4-Ein	9.1e + 01	0.0	5.2e + 05	1.0e + 00	0.0	9.2e + 02	3.3e + 01	0.0	3.3e + 02
GDGA-FCB	4.3e + 00	80.0	1.4e + 05	6.6e − 03	53.3	3.1e + 02	3.4e − 03	0.0	1.3e + 02
DGA-S4-Log	2.2e + 01	0.0	6.9e + 02	1.8e − 02	33.3	3.4e + 00	2.1e − 01	0.0	2.4e + 01
DGA-S4-Ham	6.7e + 00	0.0	8.2e + 04	2.1e − 01	0.0	3.3e + 02	2.4e + 01	0.0	2.5e + 02
DGA-S4-Alb	9.2e + 01	0.0	1.7e + 06	1.0e + 00	0.0	1.9e + 03	8.9e + 00	0.0	3.5e + 02
DGA-S4-Ein	1.1e + 02	0.0	1.8e + 06	5.3e − 01	0.0	2.0e + 03	9.9e + 00	0.0	3.5e + 02
RGA-S2-180-Log	2.0e + 01	0.0	5.1e + 02	5.4e − 03	40.0	1.9e + 00	3.6e − 03	100.0	1.0e + 01
RGA-S2-180-Ham	2.2e + 01	0.0	7.5e + 02	7.7e − 03	60.0	5.7e + 00	3.3e + 01	0.0	7.8e + 01
RGA-S2-180-Alg	2.2e + 01	0.0	6.8e + 02	3.0e − 03	73.3	4.3e − 01	2.8e − 01	0.0	3.6e + 01
RGA-S2-180-Ein	2.2e + 01	0.0	7.0e + 02	8.4e − 03	53.3	4.2e + 00	2.6e − 01	0.0	3.4e + 01
RGA-DS1-180-Dub	2.1e + 01	0.0	6.1e + 02	6.7e − 03	53.3	2.4e + 00	1.3e − 02	100.0	1.3e + 01
RGA-S2-60-Log	1.8e + 01	0.0	3.9e + 02	1.2e − 02	46.7	1.6e + 00	3.0e − 05	100.0	6.1e + 00
RGA-S2-60-Ham	1.9e + 01	0.0	5.6e + 02	3.0e − 02	6.7	4.5e + 00	4.0e − 02	80.0	4.7e + 01
RGA-S2-60-Alg	2.1e + 01	0.0	5.2e + 02	2.4e − 02	6.7	3.1e + 00	4.5e − 02	73.3	3.1e + 01
RGA-S2-60-Ein	2.1e + 01	0.0	5.4e + 02	2.0e − 02	20.0	3.0e + 00	3.8e − 02	86.7	2.8e + 01
RGA-DS1-60-Dub	1.6e + 01	0.0	4.3e + 02	1.2e − 02	26.7	1.8e + 00	2.4e − 04	100.0	7.5e + 00
RGA-S4-360-Log	2.2e + 01	0.0	8.6e + 02	2.3e − 03	73.3	3.7e + 00	5.3e − 01	0.0	2.5e + 01
RGA-S4-360-Ham	2.4e + 01	0.0	9.3e + 04	1.4e + 00	0.0	3.6e + 02	5.0e + 01	0.0	2.5e + 02

RGA-S4-360-Alg	2.5*e* + 02	0.0	2.2*e* + 06	4.6*e* + 00	0.0	2.1*e* + 03	5.4*e* + 01	0.0	3.6*e* + 02
RGA-S4-360-Ein	3.6*e* + 02	0.0	2.2*e* + 06	3.8*e* + 00	0.0	2.2*e* + 03	5.8*e* + 01	0.0	3.6*e* + 02
RGA-DS2-360-Dub	2.2*e* + 01	0.0	1.3*e* + 03	3.3*e* − 03	66.7	4.7*e* + 00	5.8*e* − 01	0.0	2.7*e* + 01
RGA-S4-120-Log	2.1*e* + 01	0.0	6.6*e* + 02	3.3*e* − 02	13.3	3.2*e* + 00	3.3*e* − 02	93.3	2.5*e* + 01
RGA-S4-120-Ham	9.3*e* − 01	0.0	8.7*e* + 04	4.4*e* − 01	0.0	3.5*e* + 02	3.2*e* + 01	0.0	2.5*e* + 02
RGA-S4-120-Alg	1.4*e* + 02	0.0	2.1*e* + 06	5.1*e* − 01	0.0	2.1*e* + 03	1.5*e* + 01	0.0	3.6*e* + 02
RGA-S4-120-Ein	1.0*e* + 02	0.0	2.2*e* + 06	7.7*e* − 02	6.7	2.2*e* + 03	1.2*e* + 01	0.0	3.6*e* + 02
RGA-DS2-120-Dub	2.0*e* + 01	0.0	8.6*e* + 02	1.7*e* − 02	26.7	3.7*e* + 00	4.2*e* − 02	73.3	2.5*e* + 01
H-GDGA-FCB-R6	1.4*e* + 00	93.3	7.9*e* + 04	6.1*e* − 03	46.7	2.3*e* + 02	7.2*e* − 02	13.3	1.1*e* + 02
H-DGA-S4-Log-R6	2.2*e* + 01	0.0	9.0*e* + 02	7.9*e* − 03	33.3	4.2*e* + 00	2.3*e* − 01	0.0	2.3*e* + 01
H-DGA-S4-Ham-R6	2.2*e* + 01	0.0	1.3*e* + 04	1.7*e* − 02	26.7	7.5*e* + 01	1.8*e* + 00	0.0	1.6*e* + 02
H-DGA-S4-Alg-R6	2.3*e* + 01	0.0	2.4*e* + 04	1.1*e* − 02	33.3	1.1*e* + 02	2.3*e* + 00	0.0	1.7*e* + 02
H-DGA-S4-Ein-R6	2.3*e* + 01	0.0	3.4*e* + 04	8.6*e* − 03	40.0	1.4*e* + 02	1.7*e* + 00	0.0	1.8*e* + 02
GDGA-FCB-R6	8.7*e* + 00	60.0	1.4*e* + 05	6.4*e* − 03	46.7	3.2*e* + 02	4.0*e* − 01	0.0	1.3*e* + 02
DGA-S4-Log-R6	2.2*e* + 01	0.0	7.2*e* + 0	9.4*e* − 03	40.0	3.5*e* + 00	2.7*e* − 01	0.0	2.5*e* + 01
DGA-S4-Ham-R6	1.9*e* + 00	0.0	7.2*e* + 04	1.7*e* − 01	0.0	3.1*e* + 02	1.9*e* + 01	0.0	2.4*e* + 02
DGA-S4-Alg-R6	7.1*e* + 01	0.0	1.5*e* + 06	1.0*e* + 00	0.0	1.7*e* + 03	1.4*e* + 01	0.0	3.5*e* + 02
DGA-S4-Ein-R6	8.6*e* + 01	0.0	1.7*e* + 06	8.7*e* − 01	0.0	1.9*e* + 03	8.8*e* + 00	0.0	3.5*e* + 02

Table V. Thresholds for the test functions.

Test Functions	Thresholds
f_{Ros}	0.1
f_{Gri}	0.005
$f_{E\text{-}F10}$	0.05

0.93; however, its S performance is 0. This algorithm, RGA-S4-120-Ham, uses the ST4 strategy with the Hamacher FCB-crossover operators, which induces a high diversity level into the population such as is indicated in Ref. 12. Its O value shows this fact. The diversity allowed good results to be reached; however, an efficient final refinement is not achieved and therefore S was not good. In H-GDGA-FCB-R6, the joined effects of the hierarchical structure and the gradual features produce high levels of diversity (see its O value) and a suitable final local refinement. In this way, reliability and accuracy were simultaneously improved and so the better result was obtained. Finally, we point out that the B and S results of the hierarchical GDGA based on rings with three subpopulations, H-GDGA-FCB, for f_{Ros} are very good as well.

- With regard to f_{Gri} and f_{E-F10}, we observe that although H-GDGA-FCB-R6 does not return the best B results, it combines two notable qualities due to the union of the gradual features and the hierarchical structure:
  1. High O values, which indicates that it generated high diversity levels.
  2. Good final approximations, B values, which shows that the level of convergence was suitable as well.

  There is not another algorithm showing simultaneously these two desirable properties. For example, the best algorithm for f_{Gri}, with respect to B, RGA-S4-360-Log, obtained 2.3e-03 with an O value of 3.7; however, H-GDGA-FCB-R6 obtained $6.1\mathrm{e}-03$ for B with a higher O value, $2.3\mathrm{e}+0.2$. So, it may be considered that H-GDGA-FCB-R6 established a good relationship between exploration and exploitation for avoiding the premature convergence problem without sacrificing the obtaining of good approximations.

- In general, the hierarchical GDGA based on rings with six subpopulations, H-GDGA-FCB-R6, improves the behavior of the GDGAs executed, GDGA-FCB and GDGA-FCB-R6. For f_{Ros}, H-GDGA-FCB-R6 reaches better B and S results than these algorithms; for f_{Gri}, it achieves a better B result; and finally for f_{E-F10}, it returns better B and S results than GDGA-FCB-R6 and a better S value than GDGA-FCB.

We may conclude that the inclusion of a hierarchical structure in the GDGAs improves the behavior of these algorithms in such a way that both reliability and accuracy are carried through to a suitable conclusion. This fact allowed good results to be reported for complex test functions.

About the Hierarchical Homogeneous DGAs

- Comparing the B results of the hierarchical homogeneous H-DGA-S4-Log, ..., H-DGA-S4-Ein, which are based on rings with three subpopulations, with the ones of their corresponding homogeneous DGAs, DGA-S4-Log, ..., DGA-S4-Ein, we observe that, in general, no improvements are produced. So, the hierarchical structure based on rings with three subpopulations does not seem to introduce a suitable way for improving clearly the results for the complex test functions used.

- Hierarchical homogeneous H-DGA-S4-Log − R6,...,H-DGA-S4-Ein-R6, which are based on rings with six subpopulations, improve the B performance of their corresponding homogeneous DGAs, DGA-S4-Log-R6,...,DGA-S4-Ein-R6, and the one of the homogeneous DGAs DGA-S4-Log,...,DGA-S4-Ein. So, we observe again that the use of ring with six subpopulation introduces a good behavior of the hierarchical structure on the test functions.
- In general, we may observe that the results of the GDGAs, GDGA-FCB and GDGA-FCB-R6, are better than the ones of homogeneous DGAs, DGA-S4-Log,...,DGA-S4-Ein and DGA-S4-Log − R6,...,DGA-S4-Ein-R6, respectively. Furthermore, the results of the hierarchical GDGAs, H-GDGA-FCB and H-GDGA-FCB-R6, are better than the ones of the hierarchical homogeneous DGAs, H-DGA-S4-Log,...,H-DGA-S4-Ein and H-DGA-S4-Log − R6,...,H-DGA-S4-Ein-R6, respectively. These results confirm the suitability of the multiresolution based on the crossover operator, as already claimed in Ref. 12.

The two previous considerations have highlighted, separately, that the hierarchical structure and a multiresolution based on crossover operators are promising ways for empowering DGAs to have a great measure of success on complex problems.

General Remarks About the Hierarchical Structure

The hierarchical structure proposed has a better behavior using rings with a higher number of subpopulations. Next, we attempt to explain this fact.

In Ref. 32, Cantú-Paz wrote:

> “The topology is an important factor in the performance of the parallel GA, because it determines how fast (or how slow) a good solution spreads to other demes. If the topology has a dense connectivity (or a short diameter, or both), good solutions will spread fast to all the demes and may quickly take over the population. On the other hand, if the topology is sparsely connected (or has a long diameter), solutions will spread slower and the demes will be more isolated from each other, permitting the appearance of different solution. These solutions may come together at a later time and recombine to form potentially better individuals.”

With these words, Cantú-Paz pointed out that the topology has important effects in the determination of the exploitation versus exploration tradeoff of DGAs. If it has a dense connectivity, exploitation comes into force, whereas if it is sparsely connected diversity is kept and so exploration takes effect.

The hierarchical structure based on rings with six subpopulations combines sparsely connected topologies, the rings of the basic DGAs, with cubic topology of the HDGA, which has a short diameter. In this way, the basic DGAs develop exploration and the topology that joins them produces exploitation.

So, the hierarchical structure together with a multiresolution based on the crossover operator and the global and local migrations produce an effective exploration/exploitation balance, and therefore results are improved.

VI. CONCLUDING REMARKS

In this paper, we have presented HDGAs, DGAs whose nodes are simple DGAs, in which two types of migrations, local and global, are produced. We also presented a model that mixes the hierarchical structure and the idea of the GDGAs, the hierarchical GDGAs. The results of the experiment carried out with HDGAs and hierarchical GDGAs have shown that:

- HDGAs have a better behavior using rings with a higher number of subpopulations, improving the results of traditional DGAs.
- An effective exploitation/exploration balance may be induced by means of the combination of the two types of topologies underlying in the hierarchical structure: one consisting of the basic DGAs and one that connects them.
- The good results obtained on the test function used and, in particular, on the complex generalized Rosenbrock's function allow us to point out that HDGAs are promixing for dealing with complex problems.
- The union of the gradual features and the hierarchical structure allows reliability and accuracy to be improved.

So, all these conclusions indicate that the introduction of a hierarchy between DGAs is a promising way for increasing the efficacy of DGAs. This paper is a first investigation on a basic implementation of this idea, the connection of different DGAs establishing a higher level DGA. Finally, we should point out that future research needs to be done on the following topics related with HDGAs:

- Study other topologies for the basic DGAs and the ones that connect them. This study should consider the most appropriate multiresolution based models as basic DGAs and the most suitable types of local and global migrations associated with these topologies. A part of this study should tackle the determination of a correct balance between the behavior of the basic DGAs and the global behavior of the HDGA.
- Study the impact of the control parameters that affect the behavior of the basic DGAs and the higher level DGA, such as migration rates, migration intervals, subpopulation sizes, etc.
- Find efficient models for the selection of the chromosomes that migrate, the selection of the chromosomes in the destination subpopulations that are replaced by the ones that arrive and deal with the choice of whether or not to replicate migrating individuals.
- Design homogeneous HDGAs with heterogeneous basic DGAs, such as an homogeneous hierarchical GDGA composed by the same type of basic GDGAs.

References

1. Goldberg, D. E. Genetic Algorithms in Search, Optimization, and Machine Learning; Addison-Wesley: New York, 1989.
2. Holland, J. H. Adaptation in Natural and Artificial Systems; The MIT Press: London, 1992.
3. Bäck, T. Evolutionary Algorithms in Theory and Practice; Oxford, 1996.
4. Davis, L. Handbook of Genetic Algorithms; Van-Nostrand Reinhold: New York, 1991.

5. Michalewicz, Z. Genetic Algorithms + Data Structures = Evolution Programs; Springer-Verlag: New York, 1992.
6. Mitchell, M. An Introduction to Genetic Algorithms; The MIT Press: London, 1996.
7. Bäck, T. In Proceedings of the First IEEE Conference on Evolutionary Computation; IEEE Press, New York, 1994, pp 57–62.
8. Li, T.-H.; Lucasius, C. B.; Kateman, G. Anal Chim Acta 1992, 2768, 123–134.
9. Mahfoud, S. W. Niching Methods for Genetic Algorithms; IlliGAL Rep 95001; Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign, IL, 1995.
10. Potts, J. C.; Giddens, T. D.; Yadav, S. B. IEEE Trans Syst Man Cybernet 1994, 24, 73–86.
11. Herrera, F.; Lozano, M.; Verdegay, J. L. Int J Intell Sys 1996, 11, 1013–1041.
12. Herrera, F.; Lozano, M.; Verdegay, J. L. Fuzzy Sets and Systems 1997, 92, 21–30.
13. Herrera, F.; Lozano, M. In Genetic Algorithms and Soft Computing; Herrera, F.; Verdega, J. L., Eds.; Studies in Fuzziness and Soft Computing, Vol. 8; Physica-Verlag: Berlin, Germany, 1996, pp 95–125.
14. Grefenstette, J. J. IEEE Trans Syst Man Cybern 1986, 16, 122–128.
15. Schaffer, J. D.; Caruana, R. A.; Eshelman, L. J.; Das, R. In Proceedings of the Third International Conference on Genetic Algorithms; Schaffer, J. D., Ed.; Morgan Kauffman: San Mateo, CA, 1989, pp 51–60.
16. Collins, R. J. Studies in Artificial Evolution; Doct Diss., University of California: Los Angeles, CA, 1992.
17. Davidor, Y. In Proceedings of the Fourth International Conference on Genetic Algorithms; Belew, R.; Booker, L. B., Eds.; Morgan Kaufmann: San Mateo, 1991, pp 257–263.
18. Manderick, B.; Spiessens, P. In Proceedings of the Third International Conference on Genetic Algorithms; Schaffer, J. D., Ed.; Morgan Kaufmann: San Mateo, CA, 1989, pp 428–433.
19. Mühlenbein, H. In Proceedings of the Third International Conference on Genetic Algorithms; Schaffer, J. D., Ed.; Morgan Kaufmann: San Mateo, CA, 1989, pp 416–421.
20. Mühlenbein, In Foundations of Genetic Algorithms 1; Rawlin, G. J. E., Ed.; Morgan Kaufmann: San Mateo, CA, 1991, pp 316–337.
21. Mühlenbein, H.; Schomisch, M.; Born, J. In Fourth International Conference on Genetic Algorithms, Belew, R.; Booker, L. B., Eds.; Morgan Kaufmann: San Mateo, CA, 1991, pp 271–278.
22. Tanese, R. In Proceedings of the Third International Conference on Genetic Algorithms; Schafer, J. D., Ed.; Morgan Kaufmann: San Mateo, CA, 1989, pp 434–439.
23. Cohoon, J. P.; Martin, W. N.; Richards, D. S. In Parallel Problem Solving from Nature 1; Schwefel, H.-P.; Männer, R., Eds.; Springer-Verlag: Berlin, Germany, 1990, pp 134–144.
24. Dorigo, M.; Maniezzo, V. In Parallel Genetic Algorithms: Theory and Applications; IOS: Amsterdam, The Netherlands, 1993, pp 5–42.
25. Herrera, F.; Lozano, M. IEEE Transactions on Evolutionary Computation; 1999. To appear.
26. Gruau, F. In Parallel Computing: Trends and Application; Joubert, G. R.; Trystram, D.; Peters, F. J.; Evans, D. J., Eds.; Elsevier: Amsterdam, 1994.
27. Voigt, H. M.; Born, J. In Parallel Problem Solving from Nature 2; Männer, R.; Manderick, B., Eds.; Elsevier Science: Amsterdam, 1992, pp 199–208.
28. Maresky, J. On Efficient Communication in Distributed Algorithms; M.S. Dissertation; Institute of Computer Science, The Hebrew University of Jerusalem, 1994.

29. Wright, S. In Proceedings of the Sixth International Congress of Genetics; Vol. 1, 1934, pp 356–366.
30. Eldredge, N.; Gould, S. J. In Models of Paleobiology, Schopf, T. J. M., Ed.; Freeman, Cooper: San Francisco, CA, 1972, pp 82–115.
31. Adamidis, P. Review of Parallel Genetic Algorithms Bibliography; Tech. Rep., Aristotle University of Thessaloniki, Thessaloniki, Greece, 1994.
32. Cantú-Paz, E. IlliGAL Rep 95007; Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign, IL, 1995.
33. Gordon, V. S.; Whitley, D.; Böhm, A. P. W. In Parallel Problem Solving from Nature 2; Männer, R.; Manderick, B., Eds.; Elsevier Science: Amsterdam, 1992, pp 533–542.
34. Lin, S.-C.; Punch III, W. F.; Goodman, E. D. In Proceedings of the Sixth IEEE Parallel and Distributed Processing, IEEE Press: New York, 1993, pp 28–37.
35. Bäck, T.; Beielstein, T.; Naujoks, B.; Heistermann, J. In Proc. EuroPVM'95: Second European PVM. User's Group Meeting, Lyon; Dongarra, J.; Gengler, M.; Tourancheau, B.; Vigouroux, X., Eds.; Hermes: Paris, 1995, pp 277–282.
36. Grefenstette, J. J. Parallel Adaptive Algorithms for Function Optimization; Tech Rep CS-81-19; Vanderbilt University: Nashville, TN, 1981.
37. Punch, W.; Goodman, E.; Pei, M.; Chai-Shun, L.; Hovland, P.; Enbody, R. In Proceedings of the Fifth International Conference on Genetic Algorithms; Forrest, S., Ed.; Morgan Kaufmann: San Mateo, CA, 1993, pp 557–564.
38. Belding, T. C. In Proceedings of the Sixth International Conference on Genetic Algorithms; Eshelman, L., Ed.; Morgan Kaufmann: San Francisco, 1995, pp 114–121.
39. Cohoon, J.; Hedge, S.; Martin, W.; Richards, D. In Proceedings of the Second International Conference on Genetic Algorithms and their Applications; Grefenstette, J. J., Ed.; L. Erlbraum Associates: Hillsdale, MA, 1987, pp 148–154.
40. Loraschi, A.; Tettamanzi, A.; Tomassini, M.; Verda, P. In Artificial Neural Networks and Genetic Algorithms; Pearson, D. W.; Steele, N. C.; Albrecht, R. F., Eds.; Springer-Verlag: Berlin, Germany, 1995.
41. Tanese, R. In Proceedings of the Second International Conference on Genetic Algorithms and their Applications, Grefenstette, J. J., Ed.; L. Erlbraum Associates: Hillsdale, MA, 1987, pp 177–183.
42. Tanese, R. Doct Dissertation; University of Michigan, Ann Arbor, MI, 1989.
43. Whitley, D.; Starkweather, D. J Expt Theory Artif Int 1990, 2, 189–214.
44. Ryan, C. In Practical Handbook of Genetic Algorithms: Applications; Chambers, L., Ed.; CRC Press: Boca Raton, Florida, 1995, pp 57–74.
45. Kohlmorgen, U. In Proceedings of the First Nordic Workshop of Genetic Algorithms; Vaasa: Finland, 1995, pp 135–145.
46. Schlierkamp-Voosen, D.; Mühlenbein, H. In Parallel Problem Solving from Nature 3, Davidor, Y.; Schwefel, H.-P.; Männer, R., Eds.; Springer-Verlag: Berlin, 1994, pp 199–208.
47. Schnecke, V.; Vornberger, O. In 4th International Conference on Parallel Problem Solving from Nature; Lecture Notes on Computer Science Vol. 1141; Springer-Verlag: Berlin, 1996, pp 859–868.
48. Preparata, J. F.; Vuillemin, J. E. Commun ACM 1981, 24(4), 300–309.
49. Maresky, J.; Davidor, Y.; Gitler, D.; Aharoni, G.; Barak, A. In Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence; Morgan Kaufmann: San Mateo, CA, 1995, pp 961–966.
50. Kröger, B.; Schwenderling, P.; Vornberger, O. In Parallel Genetic Algorithms: Theory and Applications; IOS: Amsterdam, The Netherlands, 1993, pp 151–186.
51. Baker, J. E. In Proceedings of the First International Conference on Genetic Algorithms and their Applications; Grefenstette, J. J., Ed.; L. Erlbaum Associates: Hillsdale, MA, 1985, pp 101–111.

52. De Jong, K. A. An Analysis of the Behavior of a Class of Genetic Adaptive Systems, Doct Diss, University of Michigan: MI, 1975.
53. Schwefel, H.-P. Numerical Optimization of Computer Models; Wiley: Chicester, 1981.
54. Griewangk, A. O. Journal of Optimization Theory and Application 1981, 34, 11–39.
55. Whitley, D.; Beveridge, R.; Graves, C.; Mathias, K. J Heurist 1995, 1, 77–104.
56. Whitley, D.; Rana, S.; Dzubera, J.; Mathias, E. Artif Intell 1996, 85, 245–276.
57. Herrera, F.; Lozano, M.; Verdegay, J. L. Artif Intell Rev 1998, 12, 265–319.