University of Castilla-La Mancha

A publication of the

Department of Computer Science

Migration of probability models instead of individuals: an alternative when applying the island model to EDAs.

by

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

Technical Report **#DIAB-03-06-01** January 2004

DEPARTAMENTO DE INFORMÁTICA
ESCUELA POLITÉCNICA SUPERIOR
UNIVERSIDAD DE CASTILLA-LA MANCHA
Campus Universitario s/n
Albacete - 02071 - Spain
Phone +34.967.599200, Fax +34.967.599224

Migration of probability models instead of individuals: an alternative when applying the island model to EDAs.

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

February 2, 2004

Abstract

In this work we experiment with the application of island models to Estimation of Distribution Algorithms (EDAs) in the field of combinatorial optimization. This study is motivated by the success obtained by these models when applied to other meta-heuristics (such as genetic algorithms, simulated annealing or VNS) and by the use of a compact representation of the population that make EDAs through probability distributions. This fact can be exploited during information interchange among islands. Thus, besides discussing the performance of different topologies, in this work we focus our research into three different ways of performing migration: individuals, models and models weighted by fitness. The proposed algorithms are tested by using four well known combinatorial optimization problems as benchmarks.

1 Introduction

Due to the own nature of genetic algorithms and their use to solve problems which became increasingly harder, first attempts to parallelize them came out soon. Thus, searches were intended to perform faster by taking advantage of computing power of parallel machines. These first attempts lead on the develop of new models of algorithms that, in some cases, differed from sequential models. These parallel genetic algorithms not only improve its behavior in execution time (as expected), but also the performance/accuracy of the original (non parallel) version of the algorithm. In general, these techniques lead algorithms to converge faster and to find better values.

Island based models appeared as one of the ways to carry out this parallelization. They basically consist on dividing the population into a group of sub-populations which evolve independently and occasionally exchange individuals among them. Research concerning to these models basically chases up both an optimal distribution for subpopulations and migration policies in order to improve performance. Besides genetic algorithms, island models have been tested successfully with other meta heuristics such as simulated annealing, and variable neighborhood search (VNS). This work concerns to the application of the island model to a recent family of evolutionary algorithms: *Estimation of Distribution Algorithms* (EDAs). Our proposal here is twofold: first, as an alternative to the classical migration of individuals we propose to migrate a compact representation of the population of each island, its probability model; then, we perform an empirical comparison between the two philosophies over a set of well known combinatorial optimization problems.

In order to do that, the paper is organized into five sections besides this introduction. Section 2 briefly describes basic ideas to parallelize evolution algorithms, laying over island models. On next (section 3), EDAs are described, paying more attention to univariate models since they will be used in this work. Section 4 describes application of island models to EDAs, putting forward several options. Section 5 is dedicated to experimentation, going into detail on the cases we have dealt with, their results and their analysis. We finally present some conclusions and possible work lines for future research in Section 6.

2 Evolutionary algorithms parallelization: Island Model.

In the literature, we can find several ways to parallelize evolutionary algorithms, going from a simple distribution of individuals evaluation among different processors to complex mechanisms which make a very intensive use of communication [3]. Island based models falls into what is called coarse grained philosophy. It consists on multiple sub-populations which evolve independently and, occasionally, exchange individuals. Experiments carried out by applying this models to other metaheuristics have given very good results and have speeded up the convergence.

Apart from the parameter setting required to specify the evolutionary algorithm which will be used to define evolution inside each island, it becomes necessary to specify parameters which determine interaction among the islands. These are the main:

- Number of islands. This parameter becomes more important when there is a fixed population size to be splitted off.
- Number of individuals which migrate from and island to another.
- Migration policies. The most common option consist on 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's necessary to define how population inside an island is formed after a migration is carried out. The most common option lies on using elitism, that is, received individuals are added to current population and, afterward, individuals with worst fitness are discarded.

3 Estimation of Distribution Algorithms: EDAs

EDAs [9] are a family of evolutionary algorithms where the transition from a population to another is made by estimating the probability distribution from better individuals and sampling it. In the case of combinatorial optimization (there are extensions to deal with numerical optimization), discrete EDAs are used, that is, each variable x_i can take a finite number of states $\{x_i^1, \ldots, x_i^k\}$ and a multinomial distribution is used to model the joint probability $P(x_1, \ldots, x_n)$.

In EDAs, the transition from the current population(D_t) to the next one (D_{t+1}) is given by the following steps:

1. Select a subset S of individuals from D_t

2. Estimate the joint probability distribution (JPD) P from S.

3. Obtain a new subset S' with $|D_t|$ individuals by sampling P.

4. Obtain a new population D_{t+1} selecting its individuals from $D_t \cup S'$.

One of the advantages of EDAs is their capability to represent the existing interrelations among the variables involved in individuals through the JPD. Since in real problems it is impossible to deal with the JPD, the idea is to estimate a probabilistic model $\mathcal{M}$ which will be a simplification of such distribution. The more complex $\mathcal{M}$ is the richer it becomes, but it also increases the complexity of model estimation.

Three main groups of EDAs are usually considered: univariate models [13, 1], which assume that variables are marginally independent; bivariate models [4, 2], which accept dependences between pairs of variables; and multivariate models [8], where the dependence degree is not limited.

The scheme of a generic EDA (parameters will be described later) is as follows:

1. $D_0 \leftarrow$ Generate initial population (m individuals)
2. Evaluate population D_0
3. $k = 1$
4. Repeat until stop condition
   (a) $D_{k-1}^{S_e} \leftarrow$ Select $n \leq m$ individuals from D_{k-1}
   (b) Estimate a new model $\mathcal{M}$ from $D_{k-1}^{S_e}$
   (c) $D_{k-1}^{m} \leftarrow$ Sample m individuals from $\mathcal{M}$
   (d) Evaluate D_{k-1}^{m}
   (e) $D_k \leftarrow$ Select m individuals from $D_{k-1} \cup D_{k-1}^{m}$
   (f) $k = k + 1$

Finally, in this initial approach we will use the univariate model, described with more detail in the next section.

3.1 Univariate EDAs

They are the simplest kind of EDAs, based on the assumption of marginal independence among variables, which gives rise to the following factorization of the JPD:

$$P(x_1, x_2, \ldots, x_N) = \prod_{i=1}^{N} P(x_i) \tag{1}$$

This fact simplifies the calculus because no structural learning is required and we only have to estimate unidimensional probability distributions. On the other hand, we do not take into account the possible dependences existing among variables of the problem. However, as well as it happens with other techniques that use very strong assumptions (such as Naïve Bayes on classification problems) the results obtained by these algorithms are very competitive.

Below, we detail two well known EDAs based on the univariate approach:

UMDA (*Univariate Marginal Distribution Algorithm*) [13]. Corresponds to simplest estimation model and it consist on estimating marginal probabilities for each variable. Although considering frequencies (maximum likelihood estimation) of each pair (variable, value) would be enough, it is very common to use some smoothing technique such as m-estimation or Laplace correction [6]. Aside from this, the algorithm corresponds to the scheme previously described.

IUMDA (*Incremental UMDA*) [13]. The basic idea which lies on IUMDA algorithm consists on refining the initial model by using the one recently estimated from the last population, instead of just substituting it. To do so, a constant (α) known as learning rate, is used to control the weight of the new model when updating the current one. Thus, if P_{k-1} is the current model, and $\mathcal{M}^e$ is the model more recently learnt, the updating process is as follows:

$$P_k(X_i) = (1 - \alpha)P_{k-1}(X_i) + \alpha\mathcal{M}^e(X_i)$$

Notice that if $\alpha = 1$ then IUMDA reduces to UMDA.

This approach is clearly inspired on PBIL [1], but it uses a model for the updating process instead of just some selected individuals. As a consequence, parameters (population size and α) are quite different from those used in PBIL.

4 Application of island models to EDAs.

In literature we can find some works related to application of parallelism to EDAs. In some of them [10, 11] complex probabilistic models (Bayesian networks) are considered, and the goal is to parallelize the process of model induction. Other works [14] are devoted to solve concrete problems by using classical island models applied to EDAs (basically ring topology and migration of individuals). Finally, in some papers ([15, 5]) migration of probability vectors is outlined, although they replacement policy is based on populations and not in the combination of the probabilistic models.

4.1 Individuals vs models migration

In this work we investigate two different approaches for migration when applying the island model to EDAs:

- Migration of individuals. It consists on applying the well known island models to EDAs. That is to say, we will migrate some percentage of individuals from the population contained in each island. Topologies and other parameters will be described in section 4.2.
- Migration of models. Without any kind of doubt, the main characteristic of an EDA algorithm is the probabilistic model used to codify the best individuals in the population. In this sense, isolated individuals do not play a role as important as in GAs, where they give directly rise to new individuals by means of (selection,) crossover and mutation. In EDAs new individuals are sampled from the probabilistic model, that is, from a compact representation of a subset of the whole population.

We believe that it could be of interest to exploit this different behavior when applying the island models to EDAs. Thus, what we propose is to migrate the probabilistic model instead of a subset of the population individuals. Moreover, this alternative can improve the performance in clusters of processors since it is more efficient to exchange only one vector instead of a large set of them.

The next point to discuss is how to combine the incoming model with the one currently active in a concrete island. As this process can be viewed as a refinement of the inner model in a given island, we propose to use a convex combination (as in PBIL algorithm) controlled by a parameter β. Concretely, if $\mathcal{M}_i$ is the inner model on island i and $\mathcal{M}_r$ is the incoming model, the new model $\mathcal{M}'_i$ is obtained as follows:

$$\mathcal{M}'_i(X_j) = \beta\mathcal{M}_i(X_j) + (1-\beta)\mathcal{M}_r(X_j) \tag{2}$$

In initial experiments we have tried with two different values for β: 0.5 and 0.9. However, an alternative is to take into account the goodness of the incoming model wrt to the inner one when adjusting the value of β. In this way, an island i sends/receives a pair $(\mathcal{M}, f)$ where $\mathcal{M}$ is a probabilistic model and f is its associated fitness, which is computed as the fitness average of the $V\%$ best individuals of population in island i. Then, $\beta = \frac{f_i}{f_i+f_r}$, being f_i the fitness associated to the inner model and f_r the fitness associated to the incoming model.

4.2 Design of algorithms

In this subsection we specify the design criteria we have considered in order to completely specify the proposed islands-based EDAs. Due to the huge number of parameters involved, we have limited them to a couple of values (or three in some cases).

Perhaps the main decisions to take are those to specify the model to be used. By *model* we refer to the topology to interconnect the islands (or processors) plus the kind of migration carried out. We have tested five different (M)odels, two of them dealing with individuals whereas the other three migrate vectors:

(1) The topology used is a star (figure 1.a). A specialized processor or bridge is used in order to receive the information from all the islands, process it and return the result to all of them. In this model, the islands send individuals to the bridge, and it generates a pool with the union of all the received individuals. Afterward, the bridge select (by fitness) the

best subset from the pool and send it to every island.

(2) The topology used is a star (figure 1.a). Each island sends the pair $(\mathcal{M}_i, f_i)$ to the bridge. The bridge return the pair $(\mathcal{M}_j, f_j)$ to all the islands, where $j = \arg\max_i f_i$.

(3) The topology used is a ring (figure 1.b). Each island sends individuals to the next through a unidirectional ring.

(4) The topology used is a ring (figure 1.b). Each island sends the pair $(\mathcal{M}_i, f_i)$ to the next through a unidirectional ring.

(6) The topology used is a completely connected graph (figure 1.c). Each island broadcast the pair $(\mathcal{M}_i, f_i)$ to the rest of islands in the topology. Then, if $\{(\mathcal{M}_k, f_k); k \neq i\}$ is the set of incoming pairs received by island i, the incoming pair to be considered during replacement is computed as the following weighted average:

$$\mathcal{M}_r(X_j) = \sum_{k; f_k > f_i} \frac{f_k}{F} \mathcal{M}_k(X_j) \qquad \text{with} \quad F = \sum_{k; f_k > f_i} f_k$$

Notice that odd numbers correspond to migration of individuals whereas even number represent migration of vectors.

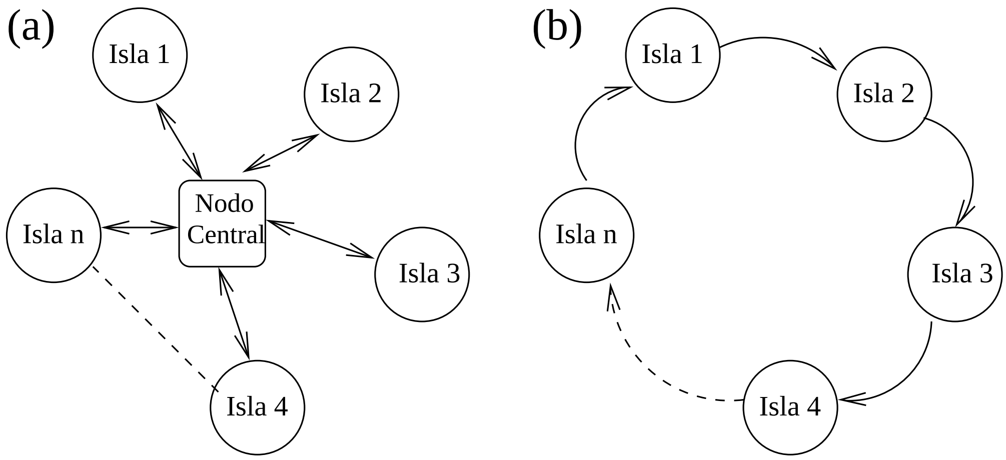

Figure 1: Topologies considered.

The rest of parameters considered are:

• Population Size (T). We have considered a population of 1000 individuals for both the sequential and the parallel version. In the former case, this population is (equally) splitted into the islands.

• Number of individuals which will compound migration (V). It represents the percentage of best individuals that will be sent or received from/to an island. In case of migrating models, they are used to compute the fitness sent together with the model. We have tried values 10 and 50.

• B. It determines the value of β. We distinguish 2 possibilities:

- $B > 0$.- In this case $\beta = B$.

- $B = 0$.- In this case β is computed by taking into account the fitness associated to the incoming and inner model: $\beta = \frac{f_i}{f_i + f_r}$.

• α. It is the value of learning rate. If equal to 1, then the island evolves by using the UMDA paradigm. The other value we have considered is 0.5. In this case, the island evolves in accordance to IUMDA.

• Number of islands (N). In our experiments we will test with 4 and 8 islands.

• Number of generations(G). Specifies the number of generations evolved inside an island before each migration is done.

• Replacement policy. In case of individuals, elitism is used, that is, incoming individuals substitute those with worst fitness in the island population. When dealing with models, the inner model is updated by using expression 2, but only if $\beta > 0$ or if $f_r \geq f_i$. Then, the algorithm goes on by sampling the new distribution.

• Cardinality of S. In the sequential version as well as on the intra-island evolution in parallel version, 50% of best individuals is selected in order to learn the new model of distribution.

• Stop criterion. Experiments evolve a maximum of 300 generations, stopping if there is no improvement in 100 generations. In our experiments, this second condition is the one that usually stops the search.

• Generation of new population. We have used two ways, both elitists, to select new population from former and sampled population inside an island:

- $e = 0$.- New population is obtained by selecting the m best individuals of $D_{k-1} \cup D_{k-1}^m$.

- $e = 1$.- New population is built by replacing the worst individual from D_{k-1}^m by the best in D_{k-1}.

5 Experimentation.

5.1 Study cases.

For our study we have chosen four different combinatorial optimization problems. The first one is described in [12] and the others have been used in the literature to evaluate different EDA models (e.g. [7]. In all of them we represent the individuals as a binary vector, however, we have tried to find different features:

• Colville function. Consists on the optimization of a numerical function. In our setting, an individual is represented by 60 positions/variables. It has a minimum whose fitness is 0.

• CheckerBoard. Dimension 10 has been considered, so representation length is 100. In this case the problem has a maximum whose fitness is 256.

• SixPeaks. Dimension 100 has been considered. In this case dimension coincides with representation length. This problem is characterized because it has 4 global and 2 local optima. In our setting t has been fixed to 30 and the maximum fitness has a value of 169.

• EqualProducts: Dimension 50 has been considered (again it coincides with representation length). The numbers have been randomly generated from a uniform distribution in [0,4]. In this case the optimum is unknown.

In all the cases our implementation try to maximize, so in Colville and EqualProducts we have transformed the fitness by multiplying by -1.

5.2 Experiments

Taking into account the parameter setting described in subsection 4.2 the number of different algorithms/configurations considered in our experimentation rise to 256. Besides, in order to have a baseline for comparison, we have firstly ran the sequential versions of the algorithms (UMDA and IUMDA $-\ \alpha = 0.5$) over the same problems. To perform a fair comparison, we have considered the same population as in the island model and the same kinds of elitism. Twenty independent runs have been carried out for each algorithm and the averaged results are shown on table 2. First line refers to the fitness (mean ± standard deviation) of the best individual found during the search and second row refers to the number of evaluations (mean ± standard deviation) carried out during the search.

	UMDA -e 1	UMDA -e 0	IUMDA -e 1	IUMDA -e 0
Colville	-6.951± 6.453	-3.152± 2.085	-4.06± 2.84	-3.889± 2.256
	23550± 9859.41	56450± 65391.51	41250± 6504.05	61400± 35504.48
CheckerBoard	231.6± 5.915	243.3± 9.223	228.9± 8.441	239.7± 8.633
	37550± 5771.76	46900± 7745.29	72500± 8696.64	80100± 12468.49
SixPeaks	102.25± 15.821	99.6± 1.789	99.8± 0.894	101.4± 8.617
	100700± 9392.61	68350± 6482.97	180650± 12872.96	98550± 13323.8
EqualProducts	-0.807± 0.829	-0.388± 0.304	-0.443± 0.452	-0.574± 0.567
	67200± 44610.12	153300± 147150.51	118700± 62778.73	150550± 128527.48

Figure 2: Results for sequential versions

Concerning to islands, we allow the same number of evaluations than in the sequential version. The 256 different configurations have been grouped in four sets. Each set is characterized by the use of UMDA/IUMDA and e=0/e=1. Due to space restrictions[1], for each problem we only include here the results for one of the groups, the one achieving (on the average) the best results. It's worth noticing that the two parameters characterizing each group (kind of algorithm and elitism) really have influence over the results because the difference among tables for a certain problem is remarkable.

Tables 3 to 18 show, for each algorithm/configuration, the same information as in the sequential case (table 2). Additionally, the last row (of each problem subtable) shows the average fitness obtained for each kind of migration.

5.3 Results analisys

After analyzing the obtained results we are in a position to drawn the following conclusions:

• As expected, island-based algorithms perform better than the sequential version.

• With respect to the two parameters used to group the configurations/algorithms in four groups, it is clear that, at least in the problems we have tested, IUMDA peforms better than UMDA. However, the kind of elitism seems to be dependent on the problem.

• With respect to the main point considered in this work, migration of individuals or models, it seems that in general migration of models performs better. Less clear is the decison about using a fixed value (0.9) for β or to adjust it using the fitness. In these cases the averaged results are similar, although we can appreciate some differences among particular settings for the remaining parameters. As an example, model (6) seems to perform worse than models (2) and (4).

6 Conclusions and future work.

An experimental study of island-based EDAs has been presented. The attention has been focused on an alternative to the classical island-based algorithms migration of individuals: migration of models. Additionally, many experiments have been conducted by trying with different topologies, replacemente policies, etc ... ¿From the results, we can conclude that migration of models is a competitive alternative to migration of individuals, having the advantage of decreasing the communication cost.

For the future we focus on two lines: first, we understand that a more detailed analysis of the resuls is required; secondly, we plan to extend our study to the bivariate and multivariate

[1]The full set of results can be downloaded from `http://www.artearmonia.com/umdat/exp1.html`

Figure 3: Results for the task Colville, population 1000.

Mean and deviation for 20 executions.
Fixed Parameters: -P problemas.binarios.Colville -T 1000 -E 1

	-M 1	-M 3	-A 0 -M 2	-A 0 -M 4	-A 0 -M 6	-A 0.9 -M 2	-A 0.9 -M 4	-A 0.9 -M 6
-a 1 -N 4 -V 10 -G 5	-2.16	-2.48	-2.85	-2.4	-1.58	-1.9	-2.87	-2.38
	35500	30500	37500	34000	35500	35000	27250	35500
-a 1 -N 4 -V 10 -G 10	-2.13	-2.18	-1.98	-2.5	-2.03	-1.88	-2.2	-1.84
	55500	43500	51000	41500	38500	50500	49500	46500
-a 1 -N 4 -V 50 -G 5	-3.76	-2.29	-2.53	-2.28	-4.8	-2.38	-2.78	-4.73
	23250	28500	33000	36250	22500	29000	33000	23000
-a 1 -N 4 -V 50 -G 10	-4.45	-2.3	-1.76	-1.74	-3.62	-2.29	-2.52	-3.46
	31000	33500	35500	61500	35500	34000	30000	31500
-a 1 -N 8 -V 10 -G 5	-0.93	-1.06	-1.37	-1.54	-0.95	-1.42	-1.53	-0.76
	53000	46250	46250	36500	51500	31500	28500	57250
-a 1 -N 8 -V 10 -G 10	-1.02	-1.26	-1.31	-2.22	-1.17	-1.65	-1.86	-1.43
	66000	63500	62000	40500	81000	53500	36000	90000
-a 1 -N 8 -V 50 -G 5	-3.33	-2.76	-1.79	-1.46	-3.53	-1.05	-1.91	-2.44
	40750	31250	40500	41500	36000	33500	37000	38750
-a 1 -N 8 -V 50 -G 10	-4.16	-1.94	-1.47	-2.63	-3.04	-1.57	-1.68	-3.65
	47500	45000	61500	44500	40500	43500	39000	48000
Medias	-2.74	-2.03	-1.88	-2.1	-2.59	-1.77	-2.17	-2.59

Mean and deviation for 20 executions.
Fixed Parameters: -P problemas.binarios.Colville -T 1000 -E 1

	-M 1	-M 3	-A 0 -M 2	-A 0 -M 4	-A 0 -M 6	-A 0.9 -M 2	-A 0.9 -M 4	-A 0.9 -M 6
-a 0.5 -N 4 -V 10 -G 5	-1.39	-1.23	-2.35	-2.17	-1.97	-2.09	-2.01	-1.92
	50000	53000	54000	49500	42500	44750	44500	44750
-a 0.5 -N 4 -V 10 -G 10	-2.12	-1.39	-1.76	-2.38	-1.82	-1.46	-1.66	-1.87
	50500	52000	58500	55000	50500	50000	49000	65000
-a 0.5 -N 4 -V 50 -G 5	-3.55	-2.9	-3.26	-1.95	-4.2	-1.73	-1.37	-3.56
	40250	45250	46500	50500	39750	50000	48500	37500
-a 0.5 -N 4 -V 50 -G 10	-3.14	-2.47	-2.54	-2.18	-1.82	-1.95	-1.36	-2.57
	54000	57500	53500	51000	57500	42000	54000	49500
-a 0.5 -N 8 -V 10 -G 5	-1.3	-1.11	-1.4	-1.18	-1.34	-1.21	-1.28	-0.96
	64500	56750	58250	53500	75750	55250	63500	81000
-a 0.5 -N 8 -V 10 -G 10	-1.27	-1.07	-1.07	-1.25	-1.17	-1.11	-1.02	-0.88
	98500	65000	76000	61500	90000	69000	69000	65000
-a 0.5 -N 8 -V 50 -G 5	-2.56	-1.82	-1.24	-1.42	-4.61	-0.96	-1.19	-4.37
	43500	48750	70500	52000	50750	59750	65750	54000
-a 0.5 -N 8 -V 50 -G 10	-1.04	-1.62	-1.11	-1.21	-1.35	-1.13	-1.12	-1.95
	97500	90000	77000	73500	90500	72500	65000	82000
Medias	-2.05	-1.7	-1.84	-1.72	-2.29	-1.45	-1.38	-2.26

Figure 4: Results for the task Colville, population 1000.

Mean and deviation for 20 executions.
Fixed Parameters: -P problemas.binarios.Colville -T 1000 -E 1000

	-M 1	-M 3	-A 0 -M 2	-A 0 -M 4	-A 0 -M 6	-A 0.9 -M 2	-A 0.9 -M 4	-A 0.9 -M 6
-a 1 -N 4 -V 10 -G 5	-1.34 59250	-1.13 98750	-1.12 60750	-0.61 62000	-2.78 63500	-1.29 62750	-0.49 61750	-2.66 58750
-a 1 -N 4 -V 10 -G 10	-1.65 95000	-1.81 141500	-0.74 76500	-1.35 63500	-1.67 95000	-0.93 64500	-0.46 61500	-1.05 98000
-a 1 -N 4 -V 50 -G 5	-3.9 48750	-1.98 94750	-2.18 61250	-1.23 56500	-3.03 50750	-1.44 58250	-1.08 59000	-2.2 56250
-a 1 -N 4 -V 50 -G 10	-1.81 73500	-1.66 126000	-1.24 62500	-1.12 76000	-2.43 83500	-1.13 66500	-1.43 77000	-2.79 73000
-a 1 -N 8 -V 10 -G 5	-2.54 54000	-1 94250	-0.67 57250	-0.65 59250	-1.36 53750	-0.4 57500	-1.14 50750	-1.89 54000
-a 1 -N 8 -V 10 -G 10	-0.56 87500	-0.82 130500	-0.91 74000	-0.84 56000	-1.18 81500	-0.45 71500	-0.69 50000	-0.92 84500
-a 1 -N 8 -V 50 -G 5	-4.48 36250	-0.92 79250	-0.22 69250	-0.98 50750	-3.95 38500	-0.7 55000	-0.94 56500	-3.29 37250
-a 1 -N 8 -V 50 -G 10	-2.37 55500	-1.13 100500	-0.63 81000	-0.62 66000	-2.78 59000	-0.54 66000	-0.65 66000	-2.33 61000
Medias	-2.33	-1.31	-0.96	-0.93	-2.4	-0.86	-0.86	-2.14

Figure 5: Results for the task Colville, population 1000.

Figure 6: Results for the task Colville, population 1000.

Mean and deviation for 20 executions.
Fixed Parameters: -P problemas.binarios.Colville -T 1000 -E 1000

	-M 1	-M 3	-A 0 -M 2	-A 0 -M 4	-A 0 -M 6	-A 0.9 -M 2	-A 0.9 -M 4	-A 0.9 -M 6
-a 0.5 -N 4 -V 10 -G 5	-1.87 75250	-1.17 118500	-1 83500	-1.4 77000	-2.35 75750	-0.87 93750	-1.09 79750	-1.94 83000
-a 0.5 -N 4 -V 10 -G 10	-1.61 118000	-1.45 174500	-0.84 84000	-1.1 92000	-1.61 124000	-1.38 75500	-1.13 89000	-1.39 112000
-a 0.5 -N 4 -V 50 -G 5	-3.21 67750	-1.43 112750	-1.16 84000	-1.26 93250	-2.38 68000	-1.36 78750	-0.73 96250	-2.96 63000
-a 0.5 -N 4 -V 50 -G 10	-1.73 103500	-2.52 152500	-1.01 98000	-1.36 80500	-2.15 111500	-1.2 81500	-0.7 87500	-2.33 113000
-a 0.5 -N 8 -V 10 -G 5	-2.37 71750	-1.15 119750	-0.2 92500	-0.65 78500	-3.07 71000	-0.82 77250	-0.23 78500	-2.09 68250
-a 0.5 -N 8 -V 10 -G 10	-2.13 99000	-0.64 164000	-0.61 89000	-0.32 92000	-0.74 102000	-0.83 82500	-0.52 87000	-1.45 99000
-a 0.5 -N 8 -V 50 -G 5	-3.14 55500	-1.29 110500	-0.62 96750	-0.97 86500	-3.64 51750	-0.33 83000	-0.69 80250	-2.95 52500
-a 0.5 -N 8 -V 50 -G 10	-2.7 86500	-0.51 138500	-0.26 94500	-0.23 82500	-1.96 89500	-1.06 85000	-0.33 82000	-1.59 88000
Medias	-2.34	-1.27	-0.71	-0.91	-2.24	-0.98	-0.68	-2.09

Mean and deviation for 20 executions..
Fixed Parameters: -P problemas.binarios.CheckerBoard -T 1000 -E 1 – -i 10

	-M 1	-M 3	-A 0 -M 2	-A 0 -M 4	-A 0 -M 6	-A 0.9 -M 2	-A 0.9 -M 4	-A 0.9 -M 6
-a 1 -N 4 -V 10 -G 5	241.2	250.55	239.5	238.4	245.35	246.05	240.4	244.15
	54000	49250	34500	50750	40750	55250	45750	48750
-a 1 -N 4 -V 10 -G 10	248.8	245.65	243.85	247.2	246.95	244.4	241.6	245.1
	69000	79000	39500	52500	99000	66500	61500	86000
-a 1 -N 4 -V 50 -G 5	237.95	241.4	233.7	241.9	234.05	244.05	242.45	232.45
	31000	34000	34000	42750	31000	58000	48500	32000
-a 1 -N 4 -V 50 -G 10	242.25	240.35	243.15	246.25	238.75	242.95	246.15	244.35
	37000	40000	43000	55500	38000	75500	56500	39000
-a 1 -N 8 -V 10 -G 5	249.4	251	235.2	249.7	248.05	251.7	242.55	246.35
	46250	64500	33750	60000	42750	74500	39000	44750
-a 1 -N 8 -V 10 -G 10	252.85	248.75	244.75	251.75	252.25	248.25	243.95	253.2
	90500	90000	41500	78000	76000	85000	47000	89000
-a 1 -N 8 -V 50 -G 5	238.4	245.35	242.35	245	240.15	249.6	244.05	235.4
	29750	38500	32500	54250	28750	57750	53500	30000
-a 1 -N 8 -V 50 -G 10	239.5	245.2	243.95	249.95	239.4	249.45	241.85	240.05
	38500	44000	41500	95000	35500	94000	45500	37000
Medias	243.79	246.03	240.81	246.27	243.12	247.06	242.88	242.63

Figure 7: Results for the task CheckerBoard, population 1000.

Mean and deviation for 20 executions.
Fixed Parameters: -P problemas.binarios.CheckerBoard -T 1000 -E 1 – -i 10

	-M 1	-M 3	-A 0 -M 2	-A 0 -M 4	-A 0 -M 6	-A 0.9 -M 2	-A 0.9 -M 4	-A 0.9 -M 6
-a 0.5 -N 4 -V 10 -G 5	243.45 75750	245.05 78250	239.05 68000	244.6 93000	242.35 74000	245.5 86000	244.2 76250	240.55 73000
-a 0.5 -N 4 -V 10 -G 10	245.4 111500	246.8 102500	244.35 81500	251.25 119000	246.7 87000	247.4 93500	244.6 87000	248.95 94500
-a 0.5 -N 4 -V 50 -G 5	237.8 60750	237.9 70000	237.7 68000	246.45 76000	240.3 58750	247.35 77750	243.3 73000	238.95 60750
-a 0.5 -N 4 -V 50 -G 10	242.15 71000	244.2 90500	242.25 83000	244.25 106500	241.45 69500	246.75 97500	247.6 89500	240.7 70500
-a 0.5 -N 8 -V 10 -G 5	243.5 67250	251.4 83000	239.05 63750	246.9 87250	246.15 67250	251.85 83500	247.3 67250	248.95 67000
-a 0.5 -N 8 -V 10 -G 10	250.55 109000	250.75 108000	244.75 87500	248.45 105500	250.8 93000	248.8 94000	245.5 72000	252.95 102500
-a 0.5 -N 8 -V 50 -G 5	236.75 54000	247.1 79750	240.65 61250	248.15 89750	240.1 55250	250.8 85500	247.25 76000	240.3 50500
-a 0.5 -N 8 -V 50 -G 10	243.2 67000	252.55 103500	246.45 85500	252.75 116500	240.35 66000	251.65 85500	248.35 68500	244.4 66000
Medias	242.85	246.97	241.78	247.85	243.52	248.76	246.01	244.47

Figure 8: Results for the task CheckerBoard, population 1000.

Mean and deviation for 20 executions.
Fixed Parameters: -P problemas.binarios.CheckerBoard -T 1000 -E 1000 – -i 10

	-M 1	-M 3	-A 0 -M 2	-A 0 -M 4	-A 0 -M 6	-A 0.9 -M 2	-A 0.9 -M 4	-A 0.9 -M 6
-a 1 -N 4 -V 10 -G 5	239.15	240.85	244.35	247.35	239.5	248.05	246.9	237.95
	45250	54000	39500	42250	43500	45750	41250	41500
-a 1 -N 4 -V 10 -G 10	245.4	243.7	245.8	245.8	240.25	246.6	246.6	244.8
	45500	58500	51500	45500	50500	45500	45000	45000
-a 1 -N 4 -V 50 -G 5	232.85	241.25	245.1	247.15	227.75	246.85	243.25	231.35
	41000	53000	36500	40500	39250	47750	45250	40000
-a 1 -N 4 -V 50 -G 10	235.8	245.3	249.8	247	239.95	249.15	243.8	240.15
	52000	44000	50500	42000	52500	47000	46000	57500
-a 1 -N 8 -V 10 -G 5	240.3	251	246.15	250.25	239.3	250.75	248.5	239.6
	38750	44500	35750	42500	37000	39000	37500	37750
-a 1 -N 8 -V 10 -G 10	245.1	251.1	249.4	248.1	245.95	248.1	248.95	247.45
	44000	67000	41000	43000	41500	54500	37500	42500
-a 1 -N 8 -V 50 -G 5	220	248.75	246.55	250.3	218.95	251.25	247.45	219.4
	28000	37000	35750	39750	29000	50500	37250	30000
-a 1 -N 8 -V 50 -G 10	236.1	247	251	250.15	238.9	248.55	247.9	231.25
	40000	40000	49500	38500	41000	49500	38000	41500
Medias	236.84	246.12	247.27	248.26	236.32	248.66	246.67	236.49

Figure 9: Results for the task CheckerBoard, population 1000.

Mean and deviation for 20 executions.
Fixed Parameters: -P problemas.binarios.CheckerBoard -T 1000 -E 1000 – -i 10

	-M 1	-M 3	-A 0 -M 2	-A 0 -M 4	-A 0 -M 6	-A 0.9 -M 2	-A 0.9 -M 4	-A 0.9 -M 6
-a 0.5 -N 4 -V 10 -G 5	235.05	240.15	244.35	250.3	233.1	248.7	250.75	236.75
	63250	85000	71250	68000	62750	71750	65000	59750
-a 0.5 -N 4 -V 10 -G 10	243.15	243.9	248.5	249.3	245.4	251.25	249.35	242.4
	81500	90000	76500	72000	74000	61000	61500	77500
-a 0.5 -N 4 -V 50 -G 5	233.8	238.35	247.45	249	236.45	249.5	248.3	229.5
	54250	77250	70500	63500	55250	66500	60750	56750
-a 0.5 -N 4 -V 50 -G 10	231.2	244	252.5	250.75	234.65	245.1	249.7	238.65
	78000	94000	69500	76500	77000	84000	70500	75500
-a 0.5 -N 8 -V 10 -G 5	237.25	246.45	250.55	249.95	239.3	252.25	250.7	234.9
	56500	71500	67250	65750	54750	58000	55500	56000
-a 0.5 -N 8 -V 10 -G 10	243.5	250.5	253.55	252.1	243.8	251.3	250.2	243.75
	67500	89500	66500	70000	72500	67000	56500	68000
-a 0.5 -N 8 -V 50 -G 5	231.2	241.75	249.7	251.3	229.85	251.55	251.25	234.55
	47500	68500	62750	60000	46500	69250	62250	46750
-a 0.5 -N 8 -V 50 -G 10	235.65	247.9	254.7	249.6	236.4	251.85	253.65	236.45
	69500	71500	68000	67000	65000	67000	63500	70500
Medias	236.35	244.13	250.16	250.29	237.37	250.19	250.49	237.12

Figure 10: Results for the task CheckerBoard, population 1000.

Mean and deviation for 20 executions.

Fixed Parameters: -P problemas.binarios.SixPeaks -T 1000 -E 1 – -i 100

	-M 1	-M 3	-A 0 -M 2	-A 0 -M 4	-A 0 -M 6	-A 0.9 -M 2	-A 0.9 -M 4	-A 0.9 -M 6
-a 1 -N 4 -V 10 -G 5	106.15	119.15	86.65	94.6	110.5	121.4	142.6	105.75
	81750	92000	84500	95000	76500	94500	108500	78500
-a 1 -N 4 -V 10 -G 10	133.8	137.55	95.9	108.85	121.6	136.65	124.4	135.45
	97000	100000	91000	104500	91000	130000	130000	97500
-a 1 -N 4 -V 50 -G 5	98.65	91.75	93.4	93.75	98.55	130.05	120.3	98.25
	72000	82000	87500	87500	72750	98750	97250	71500
-a 1 -N 4 -V 50 -G 10	99.95	88.7	92.2	97.5	100.05	136.4	117.35	100.75
	81500	88500	96500	100000	85000	132000	131000	85500
-a 1 -N 8 -V 10 -G 5	101	131.85	73	97.55	101.7	145.25	126.45	101.5
	79250	107500	61500	105750	72250	103750	98250	75000
-a 1 -N 8 -V 10 -G 10	128.15	156.65	79.55	113	140.3	150.9	101.85	131.1
	90000	175000	77000	125000	92000	161000	142500	92000
-a 1 -N 8 -V 50 -G 5	83.75	76.6	75.4	92.9	99.3	144.3	100.95	82.05
	56250	66250	70000	97000	59500	98750	90750	55500
-a 1 -N 8 -V 50 -G 10	82.9	62.65	85.4	115.55	90.5	148.85	111.9	78.95
	68000	63500	83500	126500	68500	149500	124000	66500
Medias	104.29	108.11	85.19	101.71	107.81	139.23	118.23	104.23

Figure 11: Results for the task SixPeaks, population 1000.

Mean and deviation for 20 executions.
Fixed Parameters: -P problemas.binarios.SixPeaks -T 1000 -E 1 – -i 100

	-M 1	-M 3	-A 0 -M 2	-A 0 -M 4	-A 0 -M 6	-A 0.9 -M 2	-A 0.9 -M 4	-A 0.9 -M 6
-a 0.5 -N 4 -V 10 -G 5	103.45 136250	107.05 143250	95.5 145500	92.9 149750	110.2 130250	139.4 147250	112.65 147250	108.95 131000
-a 0.5 -N 4 -V 10 -G 10	138.35 159500	136.9 176500	101.6 165500	106.45 162500	127.95 169500	140.15 230500	132.8 215000	128 157000
-a 0.5 -N 4 -V 50 -G 5	99.9 120000	99.4 144000	93.2 150000	91.75 149250	99.5 119250	120.2 146750	97.5 148250	99.8 118000
-a 0.5 -N 4 -V 50 -G 10	103.45 142500	100 160500	99.6 170000	106.2 166000	100 143000	138.25 210000	125 201000	100 141000
-a 0.5 -N 8 -V 10 -G 5	113.8 127250	120.65 137500	97.7 148250	95.45 145500	113.8 128000	136.85 145250	112.8 143750	110.2 130750
-a 0.5 -N 8 -V 10 -G 10	137.65 148500	147.45 193000	102.65 159000	122.2 160000	154.55 159500	149.65 224500	141.8 217500	151.45 150500
-a 0.5 -N 8 -V 50 -G 5	102.85 103000	99.1 136750	95.6 147000	98.55 147000	99.3 105000	130.7 145750	127.2 146750	99.35 102750
-a 0.5 -N 8 -V 50 -G 10	106.9 130500	106.9 149000	100 160000	119.7 163500	100 132000	149.9 228000	129.45 216000	103.05 130000
Medias	113.29	114.68	98.23	104.15	113.16	138.14	122.4	112.6

Figure 12: Results for the task SixPeaks, population 1000.

Mean and deviation for 20 executions.
Fixed Parameters: -P problemas.binarios.SixPeaks -T 1000 -E 1000 – -i 100

	-M 1	-M 3	-A 0 -M 2	-A 0 -M 4	-A 0 -M 6	-A 0.9 -M 2	-A 0.9 -M 4	-A 0.9 -M 6
-a 1 -N 4 -V 10 -G 5	90.6 61250	99.65 72250	102.4 69000	101.4 77000	86.35 61500	100 74750	97.7 74500	84.45 63250
-a 1 -N 4 -V 10 -G 10	102.5 76500	101.25 85000	105.2 78000	96.7 82500	101.95 78000	98.75 78000	99.25 78000	100.05 73500
-a 1 -N 4 -V 50 -G 5	68.4 52250	99.2 63750	99.5 67500	100 77250	87 58500	101.6 73500	99.15 74500	87.5 56250
-a 1 -N 4 -V 50 -G 10	99.5 71000	95.45 69000	99.45 77500	99.65 79000	88.7 64500	98.65 80000	98.3 77500	101.1 69500
-a 1 -N 8 -V 10 -G 5	85.6 57500	95.1 78250	96.5 66000	81.6 65750	84.7 61000	98.55 80500	83.3 80750	76.85 51000
-a 1 -N 8 -V 10 -G 10	102.85 77500	93.2 111500	96.5 79500	83.8 75500	91.5 71500	98.2 102500	76 71500	93.25 74000
-a 1 -N 8 -V 50 -G 5	64.15 42000	87.2 62500	93.4 70750	89.3 67500	52.25 34750	99.9 83250	83.1 84250	59.3 39250
-a 1 -N 8 -V 50 -G 10	83.65 57000	78.05 60000	100.15 85000	86.55 67000	63 52000	102.7 105500	79.3 92000	67 47500
Medias	87.16	93.64	99.14	92.37	81.93	99.79	89.51	83.69

Figure 13: Results for the task SixPeaks, population 1000.

Mean and deviation for 20 executions.
Fixed Parameters: -P problemas.binarios.SixPeaks -T 1000 -E 1000 – -i 100

	-M 1	-M 3	-A 0 -M 2	-A 0 -M 4	-A 0 -M 6	-A 0.9 -M 2	-A 0.9 -M 4	-A 0.9 -M 6
-a 0.5 -N 4 -V 10 -G 5	87.15 93250	100 101000	118.45 105000	112.85 102000	90.65 89750	100 99250	100 101000	89.45 89750
-a 0.5 -N 4 -V 10 -G 10	100 98500	99.75 102000	100 105500	100 105000	100 99000	100 105000	100 102500	99.4 101500
-a 0.5 -N 4 -V 50 -G 5	56.35 58500	98.8 102250	115.5 108750	118.55 112500	59.3 61500	100 104250	100 99000	62.3 69500
-a 0.5 -N 4 -V 50 -G 10	94.2 96500	100 101500	100 114500	100 103500	92.05 95000	101.75 100500	100 103000	82.95 94500
-a 0.5 -N 8 -V 10 -G 5	84.1 78750	100 99500	116.6 104750	106.6 109500	81.7 83750	103.45 101750	101.7 103500	81.95 80250
-a 0.5 -N 8 -V 10 -G 10	103.45 101500	100 107500	103.45 106000	102.55 110500	97.95 99500	100 110500	98.15 112500	93.8 97000
-a 0.5 -N 8 -V 50 -G 5	48.8 50750	99.55 97750	120.1 109750	110.95 109250	58.25 58000	100 102000	102.2 108750	54.8 52000
-a 0.5 -N 8 -V 50 -G 10	88.9 90500	100 104500	100 103500	103.45 107500	78.05 88000	103.45 108000	101.9 112500	89.9 97500
Medias	82.87	99.76	109.26	106.87	82.24	101.08	100.49	81.82

Figure 14: Results for the task SixPeaks, population 1000.

Figure 15: Results for the task EqualProducts, population 1000.

Mean and deviation for 20 executions.
Fixed Parameters: -P problemas.binarios.EqualProducts -T 1000 -E 1 – -i 50

	-M 1	-M 3	-A 0 -M 2	-A 0 -M 4	-A 0 -M 6	-A 0.9 -M 2	-A 0.9 -M 4	-A 0.9 -M 6
-a 1 -N 4 -V 10 -G 5	-0.97	-1.97	-1.82	-1.45	-1.1	-1.15	-0.79	-1.54
	61000	41250	38500	45750	40500	49250	56500	56500
-a 1 -N 4 -V 10 -G 10	-1.4	-0.77	-1.5	-0.83	-1.37	-1.08	-0.93	-1.08
	63500	88500	63500	93500	62500	67000	44000	64500
-a 1 -N 4 -V 50 -G 5	-2.04	-1.42	-0.84	-1.28	-0.65	-0.96	-1.64	-1.37
	42250	46750	56000	31500	52000	50250	45750	49250
-a 1 -N 4 -V 50 -G 10	-0.95	-0.53	-1.07	-0.76	-1.46	-1.1	-1.17	-1.1
	57500	70000	63000	67000	51500	72500	63000	46500
-a 1 -N 8 -V 10 -G 5	-1.92	-1.87	-0.86	-1.19	-1.91	-1.11	-0.97	-2.2
	44500	43750	42250	47500	31750	56250	45000	31750
-a 1 -N 8 -V 10 -G 10	-0.94	-0.94	-0.54	-0.8	-1.1	-1	-1.06	-1.14
	54000	88000	72000	79000	49500	60500	47000	49500
-a 1 -N 8 -V 50 -G 5	-2.18	-1.26	-0.78	-1.88	-2.95	-0.95	-1.43	-2.74
	31750	51750	47750	47500	28250	54250	40750	31500
-a 1 -N 8 -V 50 -G 10	-2.72	-0.7	-1.01	-0.96	-1.81	-0.79	-1.14	-1.98
	43500	59000	52000	74500	49000	62500	56500	44500
Medias	-1.64	-1.18	-1.05	-1.14	-1.54	-1.02	-1.14	-1.64

Mean and deviation for 20 executions.
Fixed Parameters: -P problemas.binarios.EqualProducts -T 1000 -E 1 – -i 50

	-M 1	-M 3	-A 0 -M 2	-A 0 -M 4	-A 0 -M 6	-A 0.9 -M 2	-A 0.9 -M 4	-A 0.9 -M 6
-a 0.5 -N 4 -V 10 -G 5	-1.27 46250	-1.6 36750	-2.26 43250	-1.49 43250	-1.34 44000	-1.34 43750	-1.12 42500	-1.29 42750
-a 0.5 -N 4 -V 10 -G 10	-0.61 107000	-0.66 103000	-0.76 112000	-0.53 96500	-1.26 75000	-0.56 107500	-0.83 99500	-0.79 71000
-a 0.5 -N 4 -V 50 -G 5	-1.79 45250	-1.31 60500	-1.89 48250	-1.42 52500	-1.43 55500	-1.87 50750	-2.43 51000	-1.58 40750
-a 0.5 -N 4 -V 50 -G 10	-0.75 92500	-0.47 115000	-0.68 76000	-0.71 114000	-0.77 86000	-0.54 87000	-0.48 91000	-0.68 87500
-a 0.5 -N 8 -V 10 -G 5	-1.04 52250	-1.29 43250	-0.99 53750	-1.74 48000	-1.27 48500	-1.25 48000	-1.55 42250	-1.3 56000
-a 0.5 -N 8 -V 10 -G 10	-0.84 65000	-0.64 92000	-0.43 92000	-0.47 103000	-1.21 71500	-0.41 126500	-0.5 98500	-0.84 78000
-a 0.5 -N 8 -V 50 -G 5	-1.76 47000	-1.04 38500	-1.2 52000	-1.58 47750	-1.82 40750	-1.79 41500	-1.94 43750	-1 45250
-a 0.5 -N 8 -V 50 -G 10	-0.8 79000	-1.05 89000	-0.33 110000	-0.75 90000	-1.05 62000	-0.45 82500	-0.58 83000	-1.11 75500
Medias	-1.11	-1.01	-1.07	-1.09	-1.27	-1.03	-1.18	-1.07

Figure 16: Results for the task EqualProducts, population 1000.

Mean and deviation for 20 executions.
Fixed Parameters: -P problemas.binarios.EqualProducts -T 1000 -E 1000 – -i 50

	-M 1	-M 3	-A 0 -M 2	-A 0 -M 4	-A 0 -M 6	-A 0.9 -M 2	-A 0.9 -M 4	-A 0.9 -M 6
-a 1 -N 4 -V 10 -G 5	-3.09	-1.44	-2.97	-2.37	-2.92	-1.85	-1.61	-3.01
	22000	40000	39750	50500	17000	53750	44500	28750
-a 1 -N 4 -V 10 -G 10	-1.66	-0.9	-0.89	-0.77	-1.43	-1.44	-0.86	-1.44
	51500	62000	92500	87000	47500	63500	92500	50500
-a 1 -N 4 -V 50 -G 5	-4.31	-1.72	-2.05	-0.86	-8.43	-2.18	-1.03	-3.61
	18250	32000	40750	50750	17750	32000	47250	14250
-a 1 -N 4 -V 50 -G 10	-2.59	-1.38	-1.01	-1.4	-3.36	-1.09	-0.63	-2.05
	41000	95500	81500	72500	42500	82500	106000	26500
-a 1 -N 8 -V 10 -G 5	-6.04	-1.33	-1.35	-1.86	-3.46	-2.11	-2.48	-6.5
	20250	50000	35000	42500	25250	40250	37500	18500
-a 1 -N 8 -V 10 -G 10	-1.71	-1.35	-0.62	-1.31	-2.89	-0.99	-1.01	-2.11
	41000	80500	83000	87000	47000	60000	94500	45500
-a 1 -N 8 -V 50 -G 5	-4.88	-3.58	-1.9	-2.47	-8.59	-1.53	-1.9	-7.86
	9750	34000	45500	41000	11250	51000	38250	17250
-a 1 -N 8 -V 50 -G 10	-4.4	-1.36	-1.09	-0.71	-2.39	-0.91	-0.77	-5.39
	25500	77000	84500	108000	30000	115000	99500	26500
Medias	-3.59	-1.63	-1.49	-1.47	-4.18	-1.51	-1.29	-4

Figure 17: Results for the task EqualProducts, population 1000.

Mean and deviation for 20 executions.
Fixed Parameters: -P problemas.binarios.EqualProducts -T 1000 -E 1000 – -i 50

	-M 1	-M 3	-A 0 -M 2	-A 0 -M 4	-A 0 -M 6	-A 0.9 -M 2	-A 0.9 -M 4	-A 0.9 -M 6
-a 0.5 -N 4 -V 10 -G 5	-3.45	-1.29	-1.33	-1.76	-2.94	-1.52	-2.07	-2.22
	30750	44000	57000	44250	32000	36750	51250	30000
-a 0.5 -N 4 -V 10 -G 10	-1.41	-1.63	-0.84	-0.94	-1.68	-1.1	-1.43	-1.37
	42000	97500	75500	94000	66500	90500	76500	45000
-a 0.5 -N 4 -V 50 -G 5	-4.04	-3.1	-1.96	-1.77	-3.99	-1.15	-1.81	-3.68
	18750	31250	29750	34250	22750	40500	46250	24250
-a 0.5 -N 4 -V 50 -G 10	-2.33	-1.1	-0.93	-0.98	-2.64	-0.85	-1.32	-2.08
	42500	71000	89000	80500	35000	77500	52000	41000
-a 0.5 -N 8 -V 10 -G 5	-2.98	-1.08	-1.82	-1.79	-2.41	-2.63	-2.68	-3.65
	25250	47250	22750	34750	23750	36000	41750	27250
-a 0.5 -N 8 -V 10 -G 10	-1.83	-0.8	-0.79	-1.24	-2.34	-0.85	-0.8	-1.87
	46500	67000	90500	87500	53500	64500	93500	44000
-a 0.5 -N 8 -V 50 -G 5	-7.91	-2.47	-1.85	-2.77	-6.29	-1.39	-1.73	-3.52
	13750	33250	43000	45500	17250	49000	41000	18500
-a 0.5 -N 8 -V 50 -G 10	-4.3	-1.34	-0.73	-0.6	-4.34	-1.06	-1.77	-2.53
	24000	77000	71000	113500	24000	83000	67500	29000
Medias	-3.53	-1.6	-1.28	-1.48	-3.33	-1.32	-1.7	-2.62

Figure 18: Results for the task EqualProducts, population 1000.

cases.

Acknowledgments

This work has been partially supported by the Consejería de Ciencia y Tecnología (JCCM) under project PBC-02-002.

References

[1] S. Baluja. Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning. Tech. Rep. CMU-CS-94-163, Computer Science Dpt., CMU, 1994.

[2] S. Baluja and S. Davies. Combining multiple optimization runs with optimal dependecy trees. Tech. Rep. CMU-CS-97-157, Computer Science Dpt., CMU, 1997.

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

[4] J.S. de Bonet, C. L. Isbell, and P. Viola. MIMIC: Finding optima by estimating probability densities. *Advances in Neural Inf. Processing Systems*, Vol. 9, 1997.

[5] D.E. Goldberg. *Genetic algorithms in search, optimization, and machine learning*. Addison-Wesley, New York, 1989.

[6] I.J. Good. *The Estimation of Probabilities*. MIT Press, Cambridge, 1965.

[7] P. Larrañaga, R. Etxeberria, J.A. Lozano, and J.M. Peña. Optimization by learning and simulation of Bayesian and Gaussian networks. Tech. Rep. EHU-KZAA-IK-4-99, EHU, 1999.

[8] P. Larrañaga, R. Etxeberria, J.A. Lozano, and J.M. Peña. Combinatorial optimization by learning and simulation of Bayesian networks. In *Proc. of the 16th Conference on Uncertainty in Artificial Intelligence*, pages 343–352. 2000.

[9] P. Larrañaga and J.A. Lozano. *Estimation of distribution algorithms. A new tool for evolutionary computation*. Kluwer Academic Publishers, 2001.

[10] J.A. Lozano, R. Sagarna, and P. Larrañaga. *Estimation of Distribution Algorithms. A new tool for evolutionary computation*, chapter Paralell estimation of distribution algorithms, pages 0–1. Kluwer Academic Publishers, 2001.

[11] A. Mendiburu, E. Bengoetxea, and J. Miguel. Paralelización de algoritmos de estimación de distribuciones. In *XIII Jornadas de Paralelismo*, pages 37–41, 2002.

[12] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, 1996.

[13] H. Mühlenbein. The equation for response to selection and its use for prediction. *Evolutionary Computation*, 5:303–346, 1998.

[14] V. Robles, M.S. Pérez, J.M. Peña, V. Herves, and P. Larrañaga. Parallel interval estimation naïve Bayes. In *XIV Jornadas de Paralelismo*, pages 349–353, 2003.

[15] M. Schmidt, K. Kristensen, and T.R. Jensen. Adding genetics to the standard PBIL algorithm. In *Congress of Evolutionary Computation*, pages 1527–1534, 1999.