Technical Report
EHU-KZAA-TR-2-2011

UNIVERSITY OF THE BASQUE COUNTRY
Department of Computer Science and Artificial Intelligence

Analyzing limits of effectiveness in different implementations of estimation of distribution algorithms

Carlos Echegoyen, Qingfu Zhang, Alexander Mendiburu, Roberto Santana and Jose A. Lozano

January 2011

San Sebastian, Spain
http://www.ccia-kzaa.ehu.es/

Analyzing limits of effectiveness in different implementations of estimation of distribution algorithms

Department of Computer Science and Artificial Intelligence
University of the Basque Country

Carlos Echegoyen*, Qingfu Zhang†, Alexander Mendiburu*, Roberto Santana‡
Jose A. Lozano*
* University of the Basque Country
Paseo Manuel de Lardizábal 1, 20018, San Sebastián - Donostia, Spain.
†University of Essex
Wivenhoe Park, Colchester, CO4 3SQ, UK
‡Universidad Politécnica de Madrid
Campus de Montegacedo sn. 28660. Boadilla del Monte, Madrid, Spain.

January 25, 2011

Abstract

Conducting research in order to know the range of problems in which a search algorithm is effective constitutes a fundamental issue to understand the algorithm and to continue the development of new techniques. In this work, by progressively increasing the degree of interaction in the problem, we study to what extent different EDA implementations are able to reach the optimal solutions. Specifically, we deal with additively decomposable functions whose complexity essentially depends on the number of sub-functions added. With the aim of analyzing the limits of this type of algorithms, we take into account three common EDA implementations that only differ in the complexity of the probabilistic model. The results show that the ability of EDAs to solve problems quickly vanishes after certain degree of interaction with a phase-transition effect. This collapse of performance is closely related with the computational restrictions that this type of algorithms have to impose in the learning step in order to be efficiently applied. Moreover, we show how the use of unrestricted Bayesian networks to solve the problems rapidly becomes inefficient as the number of sub-functions increases. The results suggest that this type of models might not be the most appropriate tool for the the development of new techniques that solve problems with increasing degree of interaction. In general, the experiments proposed in the present work allow us to identify patterns of behavior in EDAs and provide new ideas for the analysis and development of this type of algorithms.

1 Introduction

Estimation of distribution algorithms (EDAs) [23, 29, 31] are a population-based optimization paradigm in the field of evolutionary computation (EA) that has obtained increasing attention during the last decade. Proofs of this popularity are the development of new and more complex EDAs [19, 25, 31], the new applications for them [1, 3, 34] and the works which study fundamental issues in order to better understand how these algorithms perform [10, 18, 36]. However, the range of problems in which this type of algorithms are effective remains practically unknown. Nevertheless, some previous works [7, 9, 15] are related with this topic. We argue that studying the limits of performance of any search algorithm is a crucial task in order to advance the development of more efficient and effective techniques.

The main characteristic of EDAs is the use of probabilistic models to lead the search towards promising areas of the space of solutions throughout a learning and sampling iterative process. In fact, different EDA implementations are usually distinguished by the class of probabilistic model used. Thus, the more complex the model is, the greater the ability of an EDA to capture possible interactions among the variables of the problem. As has been shown in different works [2, 23, 37], this ability is crucial to successfully solve problems that hide interdependences among variables. However, increasing the complexity has a computational cost. Therefore, the order of dependencies in the probabilistic models must be restricted in order to achieve feasible applications. Thus, the most general EDAs, which are able to learn a new multivariate probabilistic model at each generation, use approximate learning techniques of bounded computational cost. Actually, the study and development of this class of algorithms constitutes a main topic in EDA research [4, 18, 24]. Specifically, probabilistic models such as Bayesian networks have become popular tools in this regard.

In this work, we numerically analyze the limits that different EDA implementations encounter to effectively solve problems in relation to the number of interactions between the variables. In these terms, we argue that the limits for this type of algorithms are mainly imposed by the probabilistic model they rely on. Thus, the complexity allowed in the model strongly determines to what degree of interaction in the problem the algorithm is able to successfully face. We believe that, even in the case of using general Bayesian networks, the restrictions imposed by the structural learning will cause strong limitations. In order to systematically study to what extent EDAs are able to solve problems, we use an additive decomposable function (ADF) in which new subfunctions are progressively added. With the aim of achieving a strong interdependence between the variables belonging to each sub-function, we assign deceptive function values [16]. In order to study the impact that increasing the complexity of the probabilistic model entails in the performance of the algorithm, we use three different implementations. Firstly, an EDA that assumes independencies between variables, secondly, an EDA that learns a tree-like structure at each generation and finally an EDA that learns unrestricted Bayesian networks. Since the population size is a critical parameter in EDAs and especially when Bayesian networks are learned [14], we use different population sizes in the experiments. Additionally, we use three different problem sizes to complete the analysis.

According to the results, the ability of EDAs to solve the problems generated steeply decreases after a certain number of sub-functions in the objective function. This threshold of performance shows a marked effect of phase transition that clearly delimits the frontiers of effectiveness. These limits are found by adding some additional subfunctions to a separable deceptive function similar to the Goldberg's Deceptive3 [17].

In this work, we pay special attention to EDAs based on Bayesian networks. For this type of algorithms, we show that the dramatical loss of performance is due to the lack of the structural learning method to build more complex models. Furthermore, the complexity of the networks that the algorithm has to learn in order to solve the problems rises exponentially with the number of sub-functions in the ADF. These results suggest strong computational limitations to overcome the limits of applicability of EDAs based on Bayesian networks. Therefore, the development of approaches such as probabilistic models based on factor graphs [22, 27], effective use of low-order statistics [12] or mixtures between genetic operators and learning [33, 38, 39] could be promising research areas in order to efficiently overcome the limits that the Bayesian networks exhibit in the terms presented in this paper.

The remainder of the paper is organized as follows. Section 2 introduces probabilistic models based on Bayesian networks and presents the EDA implementations used in the paper. Section 3 introduces the definition of the fitness function and the particular procedure to progressively increase the number of interactions. Section 4 explains and discusses the results of the experiments. Section 5 draws the conclusions obtained during the study. Finally, Section 6 points out possible future work.

2 Estimation of distribution algorithms

Estimation of distribution algorithms (EDAs) arise as an alternative to genetic algorithms (GAs) [17]. EDAs, instead of exchanging information between individuals through genetic operators, use machine learning methods to extract relevant features of the search space through the selected individuals of the population. The collected information is represented using a probabilistic model which is later employed to generate new solutions. The general scheme of the EDA approach is shown in Algorithm 1.

Algorithm 1: **EDA**

1. $D_0 \leftarrow$ Generate N individuals randomly
2. $t \leftarrow 1$
3. **do** $\{$
4. $\quad D_{t-1} \leftarrow$ Evaluate individuals
5. $\quad D_{t-1}^{Se} \leftarrow$ Select $M < N$ individuals from D_{t-1} according to a selection method
6. $\quad p_t(\boldsymbol{x}) = p(\boldsymbol{x}|D_{t-1}^{Se}) \leftarrow$ Estimate the joint probability by using a probabilistic model
7. $\quad D_t \leftarrow$ Sample M individuals from $p_t(\boldsymbol{x})$ and create the new population
8. $\quad t \leftarrow t + 1$
9. $\}$ **until** Stop criterion is met

2.1 Probabilistic models and factorizations

The term probabilistic model refers to the qualitative and quantitative structure given by a probability function [6]. Although the EDA paradigm could admit any type of probabilistic model, the most popular models are those that express their qualitative component through a graph. In particular, one class of models that has been extensively applied in EDAs are Bayesian networks [13, 28, 32].

Formally, a Bayesian network is a pair $(S, \boldsymbol{\theta})$ representing a graphical factorization of a probability distribution. The structure S is a directed acyclic graph which reflects the set of conditional (in)dependences among the variables. Regarding $\boldsymbol{\theta}$, it is a set of parameters for the local probability distributions associated with each variable.

The factorization of the probability distribution is codified as:

$$p(\boldsymbol{x}) = \prod_{i=1}^{n} p(x_i | \boldsymbol{pa}_i) \tag{1}$$

where $\boldsymbol{pa}_i$ denotes a value of the variables $\boldsymbol{Pa}_i$, the parent set of X_i in the graph S.

With reference to the set of parameters $\boldsymbol{\theta}$, if the variable X_i has r_i possible values, the local distribution $p(x_i | \boldsymbol{pa}_i^j, \boldsymbol{\theta}_i)$ is an unrestricted discrete distribution:

$$p(x_i^k | \boldsymbol{pa}_i^j, \boldsymbol{\theta}_i) \equiv \theta_{ijk} \tag{2}$$

where $\boldsymbol{pa}_i^1, \ldots, \boldsymbol{pa}_i^{q_i}$ denote the q_i possible values of the parent set $\boldsymbol{Pa}_i$. In other words, the parameter θ_{ijk} represents the probability of variable X_i being in its k-th value, knowing that the set of its parents' variables is in its j-th value. Therefore, the local parameters are given by $\boldsymbol{\theta}_i = ((\theta_{ijk})_{k=1}^{r_i})_{j=1}^{q_i}$.

2.2 EDA implementations

In this work, besides studying the learning limits of EDAs based on Bayesian networks, we also show to what extent this type of algorithms outperforms EDAs with simpler models. This is useful to analyze the utility of increasing the complexity of the model.

Specifically, we use the following three EDA implementations that only differ in the structural model. Firstly, we deal with the well known univariate marginal distribution algorithm (UMDA) [26]. The model used to estimate the joint probability distribution from the selected individuals at each generation is as simple as possible. It is factorized as a product of independent univariate marginal distributions. This factorization is given by Equation 1, where the parent sets are empty ($\forall i, \boldsymbol{Pa}_i = \emptyset$). Secondly, we use an EDA that learns a bivariate probabilistic model at each generation which will be called Tree-EDA. In particular, the factorization of the probability distribution is given by a tree structure learned through the algorithm proposed by Chow and Liu [8]. The graph that represents the tree structure is directed and its factorization can be formulated by using Equation 1 where all the parent sets have only one variable ($\forall i, |\boldsymbol{Pa}_i| = 1$). Lastly, we use the estimation of Bayesian network algorithm (EBNA) that considers general models without restrictions. In this case, the structural learning is based on a score+search technique [23]. Specifically, the search is carried out using the algorithm B [5] and the score is the Bayesian Information Criterion (BIC) [35]. Algorithm B is a greedy search procedure which starts with an arcless structure and, at each step, adds the arc with the maximum improvement in the score. The algorithm finishes when there is no arc whose addition improves the score.

Since all the above mentioned factorizations of the probability distribution can be encoded by using Bayesian networks, we use similar approaches both to learn the parameters and sample the new solutions. Thus, the parameters $\boldsymbol{\theta}$ are estimated by maximum likelihood using Laplace correction [23] and the sampling is carried out by a common method known as probabilistic logic sampling (PLS) [20].

3 Fitness functions

In order to investigate the behavior of EDAs as the complexity of the function increases, we deal with additively decomposable functions (ADFs). It is well known that many optimization problems studied over the years can be modelled by using this type of functions. Our model function, in which new sub-functions are progressively added, can be seen as a system that increases its complexity with the time due to the creation of new interactions among the variables.

3.1 Definitions

A fitness function $f : \boldsymbol{X} = \{0,1\}^n \rightarrow \mathbb{R}$ is additive if it can be represented as a sum of sub-functions of lower dimension,

$$f(\boldsymbol{x}) = \sum_{\boldsymbol{C}_i \in \mathcal{C}} f_i(\boldsymbol{c}_i), \tag{3}$$

where $\mathcal{C} = \{\boldsymbol{C}_1, .., \boldsymbol{C}_l\}$ is a collection of distinct sub-sets $\boldsymbol{C}_i \subseteq \boldsymbol{X}$ and $\boldsymbol{c}_i$ denotes the assignment of the variables belonging to $\boldsymbol{C}_i$. This type of functions is also characterized by its order k, which is the size of the largest sub-set in $\mathcal{C}$.

In this work, we use a particular instance of this general class of functions. Firstly, all the sub-sets in $\mathcal{C}$ have three variables ($k = 3$). Therefore, $\mathcal{C}$ consists of any collection of distinct sub-sets taken from all the $C(n,k) = \binom{n}{k}$ possible sub-sets of variables. Secondly, all the sub-functions f_i are the same deceptive function [17] denoted by f_{3dec} . By using a unitation function $u(\boldsymbol{y}) = \sum_{i=1}^{k} y_i$ where $\boldsymbol{y} \in \{0,1\}^k$, the deceptive function of three variables is defined as,

$$f_{3dec}(\boldsymbol{c}_i) = \begin{cases} 0.9 & for \quad u(\boldsymbol{c}_i) = 0 \\ 0.8 & for \quad u(\boldsymbol{c}_i) = 1 \\ 0.0 & for \quad u(\boldsymbol{c}_i) = 2 \\ 1.0 & for \quad u(\boldsymbol{c}_i) = 3 \end{cases}$$

From the point of view of the EDA analysis, the benefit of the fitness functions that we propose is twofold. Firstly, independently of the number of sub-functions, we always know the global optimum which is the assignment of all ones for $\boldsymbol{X}$. Secondly, the deceptive approach creates a strong interdependence among the three variables belonging to each sub-function.

Regarding the f_{3dec} functions, when they are disposed without overlapping among the set of variables $\boldsymbol{X}$, the fitness function is called Goldberg's Deceptive3 function [16]. This specific function was proposed in the context of genetic algorithms aimed at finding their limitations. Thus, in the present work, this function constitutes a useful reference in order to analyze the limits of performance in EDAs. Nowadays, deceptive or trap separable functions are widely used to test evolutionary algorithms.

3.2 Implementation of the functions

In order to progressively increase the complexity of the functions, we propose a simple procedure. Basically, we generate a sequence of objective functions in which each function adds one more sub-function than the previous one. This sequence of functions is given by the ordered set $\mathcal{C} = \{\boldsymbol{C}_1, .., \boldsymbol{C}_l\}$. This set is a collection of l distinct

subsets of variables randomly selected according to a uniform distribution from all the $C(n,3)$ possible combinations. Although we could introduce in $\mathcal{C}$ all the $C(n,3)$ distinct subsets, it is not necessary for our purposes. Thus, the s-th objective function in the sequence sums s sub-functions which are applied to the corresponding first s subsets of variables in $\mathcal{C}$. The s-th function can be expressed as,

$$f_s(\boldsymbol{x}) = \sum_{i=1}^{s \leq l} f_{3dec}(\boldsymbol{c}_i) \tag{4}$$

The ordered set $\mathcal{C}$ has a restriction. The first n/k subsets cover the whole set of variables $\boldsymbol{X}$ without overlapping, forming a separable function. Note that in the functions from $s = 1$ to $s = n/k$ some of the variables do not have any sub-function assigned. In order to complete these functions and keep the optimum in the assignment of all ones in this specific stage, we directly apply the unitation function over the set of variables $\boldsymbol{X} \setminus \bigcup_{i=1}^{s} \boldsymbol{C}_i$ without sub-function assignment. This stage is useful to analyze UMDA, Tree-EDA, and even EBNA with insufficient population size to reliably reach the optimum, because the algorithms can fail before reaching the level of the Godberg's Deceptive3 function. Furthermore, this separable function is useful as a reference of problem difficulty.

Finally, due to the random nature of the set $\mathcal{C}$, we create for the experiments 100 different random instances of this type of sets and the results shown are the average among them. We use a problem size of $n = 72$ and the maximum allowed numbers of combinations in $\mathcal{C}$ is $l = 200$. In addition, for each possible function, we carry out 10 independent EDA runs.

3.3 Degree of interaction and problem difficulty

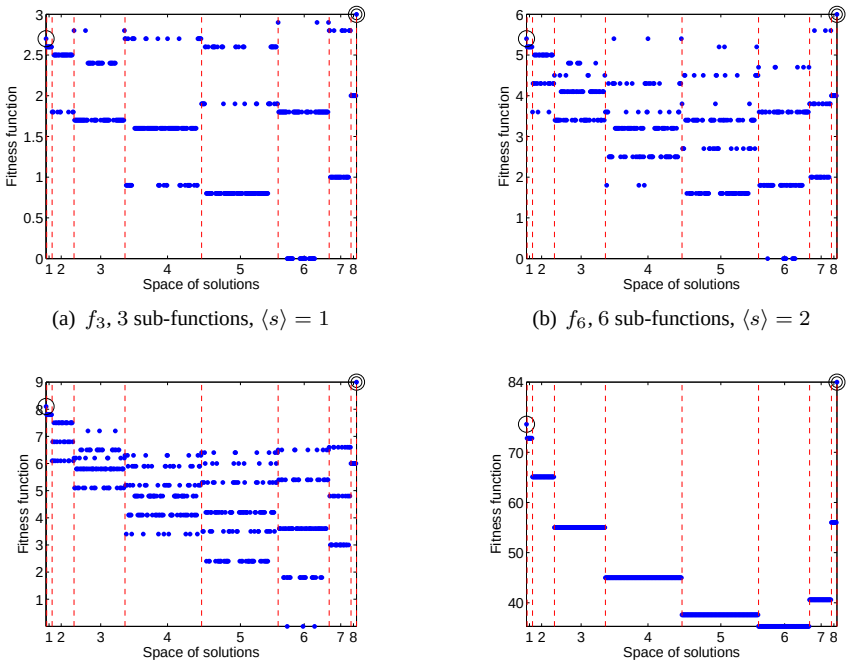
The degree of interaction can be seen as a concept related with the interdependences that emerge among the variables of a problem. Although there could be many different ways to measure this notion, in the context of the present work, we assume that the degree of interaction is essentially given by the number of sub-functions included in the objective function.

Additionally, in order to provide a more intuitive measure of the degree of interaction, we also take into account the number of sub-functions that each variable has assigned. For example, in the separable deceptive function each variable belongs to only one sub-function. In general, given s sub-functions of size k over n variables, we calculate the expected number of sub-functions assigned to each variable. It is given by,

$$\langle s \rangle = s\frac{k}{n}. \tag{5}$$

Note that, the maximum number of sub-functions assigned per variable is $k\frac{C(n,k)}{n}$ and therefore depends on the problem size. A measure whose range is independent of this parameter is the density of sub-functions in the objective function, which is given by $d = \frac{s}{C(n,k)}$.

In order to illustrate how the landscapes of the functions change according to the number of sub-functions added, we present a simple example with 9 variables. Fig. 1 shows the function values that four different objective functions assign to all possible solutions of the search space. Due to the properties of the objective functions, we can group these solutions by the number of ones in order to provide more intuitive



(a) f_3, 3 sub-functions, $\langle s \rangle = 1$

(b) f_6, 6 sub-functions, $\langle s \rangle = 2$

(c) f_9, 9 sub-functions, $\langle s \rangle = 3$

(d) f_{84}, all the 84 sub-functions, $\langle s \rangle = 28$

Figure 1: Different snapshots that show how the landscape of the functions with $n = 9$ changes according to the number of deceptive sub-functions. The space of solutions in the x-axis is grouped and labeled by the number of ones. The vertical dashed lines enclose the different groups of solutions that have the same number of ones. The function values of the solutions are in the y-axis. The assignment of all zeros is highlighted with a circle and the optimum (all ones) with two concentric circles. (a) Landscape of the function f_3 that adds 3 sub-functions and has $\langle s \rangle = 1$. (b) Landscape of the function f_6 that adds 6 sub-functions and has $\langle s \rangle = 2$. (c) Landscape of the function f_9 that adds 9 sub-functions and has $\langle s \rangle = 3$. (d) Landscape of the function f_{84} that adds all the 84 possible sub-functions and has $\langle s \rangle = 28$.

plots. The vertical dashed lines enclose the groups of solutions that have the same number of ones. For example, the area that corresponds to the number 4 (x-axis) contains exactly the $C(9,4)$ solutions with 4 ones. Thus, the area that corresponds to the number 0 or number 9 in the x-axis only includes one solution. Nevertheless, both these solutions play a crucial role in the class of functions that we propose, and therefore, the assignment of all zeros is highlighted with a circle and the optimum (all ones) with two concentric circles.

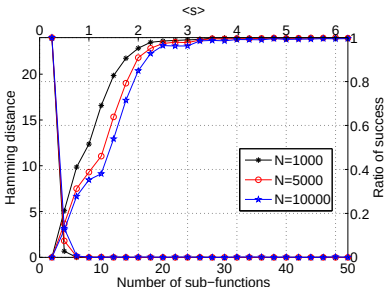
For example, Fig. 1(a) corresponds to the landscape of the Goldberg's separable deceptive function and Fig. 1(d) corresponds to a function that sums all $C(n,k)$ possible sub-functions. In turn, Figs. 1(b) and 1(c) show the landscapes for two intermediate functions in which each variable exactly belongs to 2 and 3 sub-functions. In general, we can observe, without using specific measures of problem difficulty for EAs [21, 30], that the optimum tends to be isolated as the number of sub-functions increases. Therefore, it will be more difficult to find useful information in the populations to guide the algorithm towards the optimum. After the third snapshot (Fig.1(c)), the assignments of all zeros for $\boldsymbol{X}$ is the sub-optimum and it becomes more attractive as more sub-functions are added. The neighborhoods of this solution tend to have higher function values than the neighborhoods of the optimum. We are assuming that the neighborhoods are given by a bit-flip operator [21]. Nevertheless, we can see in Fig.1(c) that there are solutions near to the optimum that keep high quality function values, and therefore, they can contain valuable information about the optimal solution. Even in the last snapshot (Fig.1(d)), some traces of information about the optimum remain in the solutions with more than 6 ones.

4 Experiments

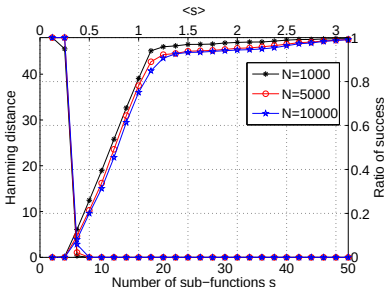
4.1 Descriptors and parameters

In order to measure the performance of EDAs we use two descriptors. Firstly, we calculate the Hamming distance between the best solution given by the algorithm and the optimum. This measure indicates the quality of the solutions that EDAs are able to return. Secondly, we calculate the ratio of successful runs. This measure represents the proportion of problems solved for each level of difficulty that the number of sub-functions provides. In addition, we record the order of the Bayesian networks learned by EBNA at each generation. The order of the Bayesian network is given by the number of variables in the largest factor of the corresponding factorization i.e. the maximum number of parents plus the child. The results that we present only take into account the maximum order among the networks learned during each run. This provides a measure of the computational cost of the search. All the results are shown both in relation to the number of sub-functions and the average number of sub-functions assigned to the variables $\langle s \rangle$.

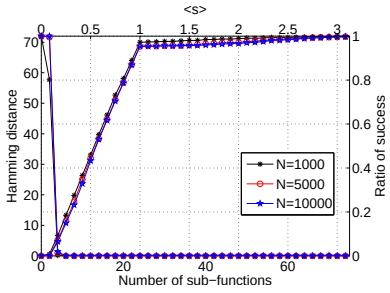
Besides the three problems sizes ($n \in \{24, 48, 72\}$) that we consider for the experiments, different population sizes are also used in order to show the impact of this parameter. Particularly, we use five different population sizes $N \in \{1000, 5000, 10000, 15000, 20000\}$. The results confirm that the population size has a higher influence on EDAs based on Bayesian networks. Therefore, in order to avoid unnecessary experiments, the last two sizes will be used only for EBNA. Finally, the stopping criterion of the algorithms is a fixed number of n generations, that is, as many as the number of problem variables.



(a) $n = 24$



(b) $n = 48$



(c) $n = 72$

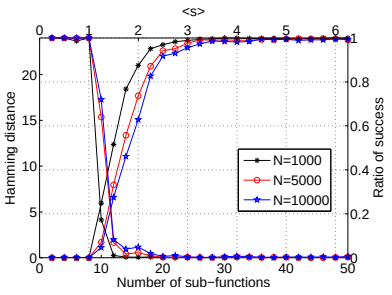
Figure 2: Hamming distance and ratio of success for UMDA with different population sizes and problem sizes. The description of the plots is as follows. The x-axis at the bottom shows the number of sub-functions. The x-axis at the top shows the average number of sub-functions assigned to each variable given by $\langle s \rangle$. The y-axis on the left of the charts shows the Hamming distance between the best solution given by the algorithm and the optimum. The curves that increase correspond to this label. The y-axis on the right shows the ratio of successful runs which correspond to the curves that decrease.

4.2 Results

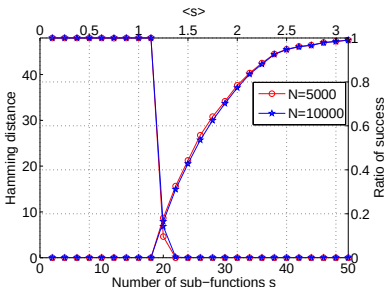
In this section, we summarize the results that we have obtained in the analysis of the limits that EDAs encounter as the number of sub-functions increases in the objective function.

Figs. 2, 3 and 4 show the performance in terms of Hamming distance and ratio of successful runs for UMDA, Tree-EDA and EBNA respectively. The different elements of the charts are explained in the labels of these figures. For the sake of simplicity and clarity in the plots, and due to the constant patterns observed in the results, some curves are ignored.

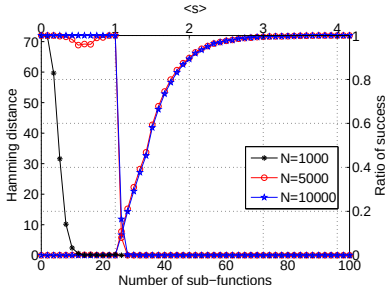
Through the two descriptors used in these three Figures, we clearly see the manner in which the performance of different EDAs collapses. The curves show an effect of phase transition after a certain level of difficulty. This effect is particularly noticeable in the curves of ratio of successful runs which fall off from 1 to 0 by only adding a low number of sub-functions. In UMDA and Tree-EDA, the difference between total success and complete failure is in approximately two sub-functions. When EDA learns Bayesian networks, the transition from 1 to 0 in ratio of successful runs is slightly more progressive, it approximately happens in 6 or 8 sub-functions. In this case, the shape



(a) $n = 24$

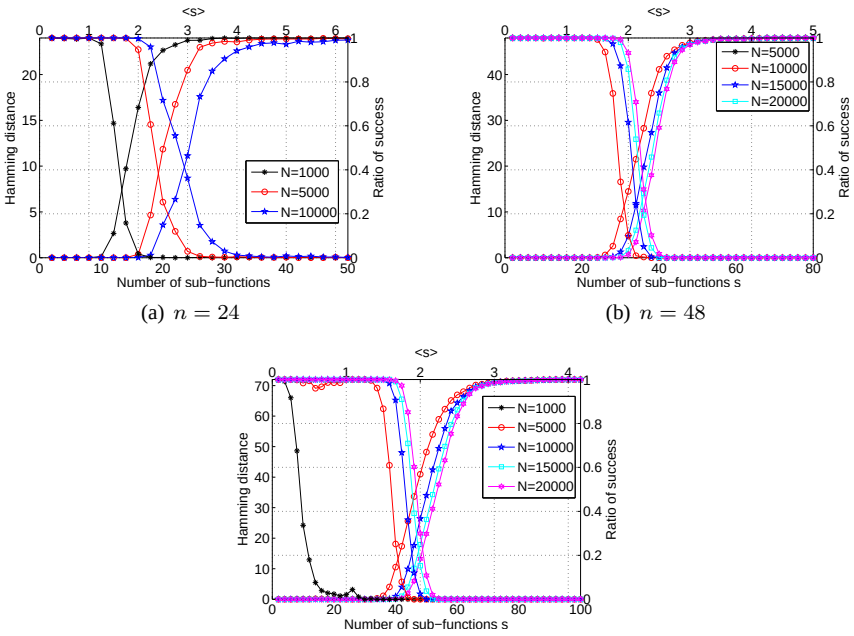


(b) $n = 48$



(c) $n = 72$

Figure 3: Hamming distance and ratio of success for UMDA with different population sizes and problem sizes. The description of the plots is as follows. The x-axis at the bottom shows the number of sub-functions. The x-axis at the top shows the average number of sub-functions assigned to each variable given by $\langle s \rangle$. The y-axis on the left of the charts shows the Hamming distance between the best solution given by the algorithm and the optimum. The curves that increase correspond to this label. The y-axis on the right shows the ratio of successful runs which correspond to the curves that decrease.



(a) $n = 24$

(b) $n = 48$

(c) $n = 72$

Figure 4: Hamming distance and ratio of success for UMDA with different population sizes and problem sizes. The description of the plots is as follows. The x-axis at the bottom shows the number of sub-functions. The x-axis at the top shows the average number of sub-functions assigned to each variable given by $\langle s \rangle$. The y-axis on the left of the charts shows the Hamming distance between the best solution given by the algorithm and the optimum. The curves that increase correspond to this label. The y-axis on the right shows the ratio of successful runs which correspond to the curves that decrease.

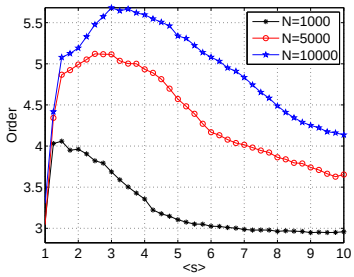
of the curves is more flexible and has a greater dependence both on the population size and problem size. In fact, according to our results, the more complex the model is, the greater the influence of these two elements. On the one hand, whereas UMDA is hardly influenced by the population and problem size, we can see that Tree-EDA and EBNA need a minimum population size to have a competent performance and it becomes evident as the problem size increases. In Figs. 3(c) and 4(c), the curves of ratio of success show that the lowest population size is clearly insufficient to reach the optimum. Nevertheless, the population size in the curves for EBNA clearly has a greater impact than for Tree-EDA. On the other hand, the benefits that learning Bayesian networks or increasing the population size could have to achieve a better performance in EDAs tend to vanish as the problem size increases. This fact can be observed in Fig. 4. Firstly, the curves for different population sizes are closer for the problems with 48 and 72 variables. Secondly, as the problem size increases, the moment in which the algorithm starts to fail is closer to the separable deceptive function in $\langle s \rangle = 1$.

The Hamming distance shows a more progressive change that provides complementary information about the quality of the solutions. For example, although the ratio of success can be equal to 0 with only four sub-functions in Fig. 2, UMDA returns high quality solutions which are close to the optimum. Similar situations happen in other scenarios except for the smallest problem size in Fig. 4(a), where both curves are more balanced. Finally, the curves of Hamming distance indicate that, after a certain level of difficulty, all EDAs return the same solution which is in the assignment of all zeros. If we take into account the combinatorial number of sub-functions that we can introduce in the ADF, selecting UMDA would be the best option, in terms of efficiency and efficacy, to solve the great majority of the problems. Only in a small range of our problems does EBNA provides better results.

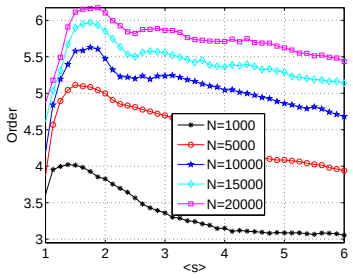
Table 1: Number of sub-functions to which EDAs have reliability of the 95% (they reach the optimum in the 95% of the runs) for each type of learning, problem size and population size. Below the number of variables, the total number of sub-functions that can be added to the ADF is shown.

Problem size	EDAs	Population size				
		1000	*5000*	*10000*	*15000*	*20000*
	UMDA	2	2	2		
$n = 24$	*Tree-EDA*	($\langle s \rangle = 1$) 8	8	8		
$C(24, 3) = 2024$	*EBNA*	10	($\langle s \rangle = 2$) 16	18	18	20
	UMDA	2	2	2		
$n = 48$	*Tree-EDA*	4	($\langle s \rangle = 1$) 16	16		
$C(48, 3) = 17296$	*EBNA*	4	22	26	28	28
	UMDA	0	2	2		
$n = 72$	*Tree-EDA*	2	($\langle s \rangle = 1$) 24	24		
$C(72, 3) = 59640$	*EBNA*	2	32	36	38	40

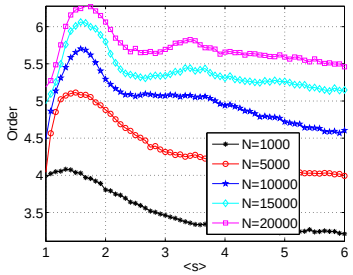
In Table 1, we show the number of sub-functions for which the different EDAs are able to keep a reliability of 95%. In addition, in this table we show the total number of sub-functions that can be added to the objective function. The specific study of these thresholds is useful to better know where the limits of this type of algorithms are in relation to the complexity of the probabilistic model and the population size.



(a) $n = 24$



(b) $n = 48$



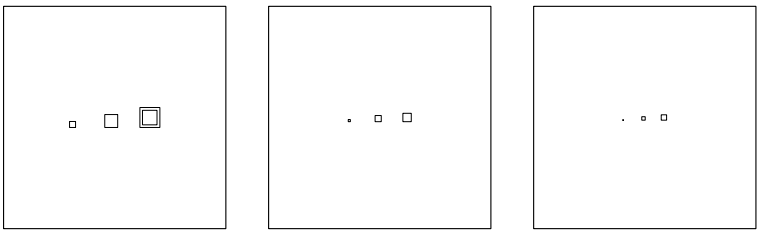
(c) $n = 72$

Figure 5: Order of the Bayesian networks learned by EBNA. For this chart, we only take into account the Bayesian network with the maximum order during each run. The results are shown in relation to the average number of sub-functions assigned to the variables.

As was expected, we can observe that the probabilistic model has an important influence to solve more difficult problems. The best results are obtained with the smallest problem size. In this case, EBNA is able to solve problems which approximately double the number of sub-functions than the separable deceptive function. However, this proportion gets lower as the problem size increases. The balance between solving a wider range of problems and the computational cost that more complex probabilistic models entails, deserves a specific analysis.

Regarding the population size, we see that, although this parameter is critical to obtain a competent behavior of EDAs based on Bayesian networks, it shows a limited utility to overcome certain thresholds of problem difficulty. This indicates that increasing the population size is not the solution to solve more difficult problems. In fact, as the size of this parameter increases, it has a diminishing impact and the level of difficulty that EBNA is able to reach tends to stabilize. In addition, Table 1 shows the results for population sizes up to $20,000$ for the problem size of $n = 24$. This is a huge population size for this number of variables because it represents an important proportion of the search space. Even in this case, the threshold of the number of sub-functions hardly varies for EBNA.

In Fig. 5, we analyze the complexity of the Bayesian networks learned by EBNA during the searches. Specifically, we focus on the order of these probabilistic models. As aforementioned, the order is given by the number of variables in the largest factor of the factorization. For this figure, we only take into account the maximum order during



(a) $n = 24$ (b) $n = 48$ (c) $n = 72$

Figure 6: Spaces of problems that the different EDAs have solved. We assume that two functions with the same number of sub-functions represent the same level of difficulty and, therefore, the same allocation in the space of problems. The area of the biggest square represents the number of all possible levels of difficulty from 0 sub-functions to $C(n, k)$. The three squares inside the big squares (from left to right are UMDA, Tree-EDA and EBNA) represent the space of problems that each EDA was able to solve with a reliability of 95% for each problem size.

each run.

The phase-transition effect observed in Figs. 2, 3 and 4 is closely related with Fig. 5. Thus, the algorithm starts to dramatically fail after the peaks in the curves shown in Fig. 5. Note that, as the problem size increases, this effect is more marked. This figure shows that the complexity of the Bayesian networks increases exponentially by only adding a few sub-functions to the ADFs. After this first stage of rising, the learning is unable to obtain adequate structures to solve the problems and the complexity of the networks decreases.

Finally, we show in Fig. 6 a more intuitive result. This figure represents the proportion of problems that different EDAs were able to solve with reliability. The big squares represent the whole space of problems that we can create for each number of variables with sub-functions of size 3. We can observe how the space of problem that EBNA is able to solve (small squares on the right) dramatically decreases with the problem size. Only for $n = 24$ the influence of the population size in EBNA can be observed (double square). This is an intuitive result that also is useful to suggest the wide scope for improvement that can exists for this type of algorithms.

5 Conclusions

In this work, we have analyzed the limits of performance that different EDA implementations encounter as the degree of interaction among the variables of the problem increases. We base the analysis on the use of additive decomposable functions in which new sub-functions with the same deceptive values are progressively added. Thus, the degree of interaction can be directly measured by the number sub-functions that the objective function includes. Moreover, we use the separable deceptive function as a reference of problem difficulty in order to provide more intuitive results. In the experiments, we have dealt with three different EDA implementations. Since these algorithms only differ in the probabilistic model used, the results show the impact that introducing more complex models has in order to solve a wider range of problems. We have also used different population sizes. This parameter has been critical in order to achieve

a competent behavior in EDAs based on Bayesian networks. However, the results suggest that, in general, increasing this parameter is not the solution to efficiently solve more complex problems. The limits seem to strongly depend on the probabilistic model itself.

We have discovered that, when the problem has strong interactions among the variables, the performance of the algorithm collapses with a phase-transition effect as the number of sub-functions in the objective function increases. The threshold in which EDAs based on Bayesian networks fails is between the separable deceptive functions and the objective functions with $2n/k$ sub-functions. We have also shown that the complexity of the networks learned by the algorithm has to exponentially rise with the number of sub-functions in the objective function in order reach the optimum. Therefore, after a certain threshold, the learning collapses and the algorithm dramatically fails. It suggests that, after a critical degree of interaction, the learning of Bayesian networks might not be able to recover the information needed to reach the optimum from the population. According to the growth that the curves of order show, the use of Bayesian networks in EDAs does not seem an appropriate tool for the development of new techniques capable of solving problems with an increasing degree of interaction among the variables. In order to make a qualitative step forward in this regard, we believe that the development of new approaches such as probabilistic models based on factor graphs [22, 27], effective use of low-order statistics [12] or mixtures between genetic operators and learning [33, 38, 39] constitutes a promising research area.

In essence, we have explored the general concept of boundaries of effectiveness in EDAs in relation to the degree of interaction of the problem. In this regard, we have exposed specific information with clear patterns of behavior that offer the possibility of conjecturing on more general issues about the limitations of this class of evolutionary algorithm.

6 Future work

There are a number of trends which are worth extending the results presented in this paper.

The first extension of this work is to test the most sophisticated EDAs that actually exist. They can include additional techniques such as niching or local searches. Finding the level of problem difficulty that this type of algorithms is able to reach is important to confirm the hypothesis of learning limits in EDAs based on Bayesian networks. In addition, the analysis of exact methods to learn Bayesian networks in EDAs [11] could provide valuable information about this issues but they are only feasible for small problems. Similarly, we should propose and test, under the same conditions, alternative approaches such as mixture of simple evolutionary algorithms instead of sophisticated EDAs.

In order to estimate to what extent the introduction of more complex models and additional techniques benefit EDAs, we should carry out a study in order to estimate the balance between computational cost and performance in the terms proposed in this work.

In order to contrast the results that we have obtained with this particular model of function, we should conduct similar experiments with other types of functions. An appropriate candidate could be the Ising problem whose difficulty could be increased with the number of interactions between couples of variables. However, the disadvantages

of this problem are 1) the calculation of the optimum and 2) the need to analyze different instances not only with different structures but also with different parameters.

Due to the constant patterns observed in the results, we think it could be possible to theoretically model the curves of ratio of success and Hamming distance. This would be useful in order to estimate the performance of EDAs based on Bayesian networks in relation to the number of sub-functions in the objective function and the problem size.

Acknowledgment

This work has been partially supported by the Saiotek and Research Groups 2007-2012 (IT-242-07) programs (Basque Government), TIN2008-06815-C02-01, TIN2010-14931 and Consolider Ingenio 2010 - CSD2007-00018 projects (Spanish Ministry of Science and Innovation) and COMBIOMED network in computational biomedicine (Carlos III Health Institute). Carlos Echegoyen holds a grant from UPV-EHU. The work of Roberto Santana has been partially supported by the TIN2010-20900-C04-04 and Cajal Blue Brain project (Spanish Ministry of Science and Innovation).

References

[1] R. Armañanzas, I. Inza, R. Santana, Y. Saeys, J. L. Flores, J. A. Lozano, Y. Van de Peer, R. Blanco, V. Robles, C. Bielza, and P. Larrañaga. A review of estimation of distribution algorithms in bioinformatics. *BioData Mining*, 1(6):1–12, 2008.

[2] S. Baluja and S. Davies. Using optimal dependency-trees for combinatorial optimization: Learning the structure of the search space. In *Proceedings of the 14th International Conference on Machine Learning*, pages 30–38. Morgan Kaufmann, 1997.

[3] A. Brownlee, M. Pelikan, J. McCall, and A. Petrovski. An application of a multivariate estimation of distribution algorithm to cancer chemotherapy. In *Proceedings of the Genetic and Evolutionary Computation Conference GECCO-2008*, pages 1033–1040. ACM Press, 2008.

[4] A. E. I. Brownlee, J. A. W. McCall, S. K. Shakya, and Q. Zhang. Structure learning and optimisation in a Markov-network based estimation of distribution algorithm. In *CEC'09: Proceedings of the Eleventh conference on Congress on Evolutionary Computation*, pages 447–454, Piscataway, NJ, USA, 2009. IEEE Press.

[5] W. Buntine. Theory refinement on Bayesian networks. In *Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence*, pages 52–60, 1991.

[6] E. Castillo, J. M. Gutierrez, and A. S. Hadi. *Expert Systems and Probabilistic Network Models*. Springer-Verlag, 1997.

[7] S.-C. Chen and T.-L. Yu. Difficulty of linkage learning in estimation of distribution algorithms. In *GECCO '09: Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, pages 397–404, New York, NY, USA, 2009. ACM.

[8] C. K. Chow and C. N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3):462–467, 1968.

[9] D. J. Coffin and R. E. Smith. The limitations of distribution sampling for linkage learning. In *Proceedings of the 2007 Congress on Evolutionary Computation CEC-2007*, pages 364–369. IEEE Press, 2007.

[10] C. Echegoyen, A. Mendiburu, R. Santana, and J. Lozano. Towards understanding EDAs based on Bayesian networks through a quantitative analysis. *IEEE Transactions on Evolutionary Computation*, 2010. Accepted for publication.

[11] C. Echegoyen, R. Santana, J. Lozano, and P. Larrañaga. *Linkage in Evolutionary Computation*, chapter The Impact of Exact Probabilistic Learning Algorithms in EDAs Based on Bayesian Networks, pages 109–139. Springer Berlin / Heidelberg, 2008.

[12] L. R. Emmendorfer and A. T. R. Pozo. Effective linkage learning using low-order statistics and clustering. *Trans. Evol. Comp*, 13(6):1233–1246, 2009.

[13] R. Etxeberria and P. Larrañaga. Global optimization using Bayesian networks. In A. Ochoa, M. R. Soto, and R. Santana, editors, *Proceedings of the Second Symposium on Artificial Intelligence (CIMAF-99)*, pages 151–173, Havana, Cuba, 1999.

[14] N. Friedman and Z. Yakhini. On the sample complexity of learning Bayesian networks. In *Proceedings of the 12th Conference on Uncertainty in Artificial Intelligence (UAI-96)*, pages 274–282. Morgan Kaufmann, 1996.

[15] Y. Gao and J. C. Culberson. Space complexity of estimation of distribution algorithms. *Evolutionary Computation*, 13(1):125–143, 2005.

[16] D. E. Goldberg. Simple genetic algorithms and the minimal deceptive problem. In L. Davis, editor, *Genetic Algorithms and Simulated Annealing*, pages 74–88. Pitman Publishing, London, UK, 1987.

[17] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, Reading, MA, 1989.

[18] M. Hauschild, M. Pelikan, K. Sastry, and C. Lima. Analyzing Probabilistic Models in Hierarchical BOA. *IEEE Transactions on Evolutionary Computation*, 13(6):1199–1217, December 2009.

[19] M. W. Hauschild, M. Pelikan, K. Sastry, and D. E. Goldberg. Using previous models to bias structural learning in the hierarchical boa. In *Proceedings of the 10th annual conference on Genetic and evolutionary computation*, GECCO '08, pages 415–422, New York, NY, USA, 2008. ACM.

[20] M. Henrion. Propagating uncertainty in Bayesian networks by probabilistic logic sampling. In J. F. Lemmer and L. N. Kanal, editors, *Proceedings of the Second Annual Conference on Uncertainty in Artificial Intelligence*, pages 149–164. Elsevier, 1988.

[21] L. Kallel, B. Naudts, and R. Reeves. Properties of fitness functions and search landscapes. In L. Kallel, B. Naudts, and A. Rogers, editors, *Theoretical Aspects of Evolutionary Computing*, pages 177–208. Springer Verlag, 2000.

[22] F. R. Kschischang, B. J. Frey, and H. A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, 2001.

[23] P. Larrañaga and J. A. Lozano, editors. *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*. Kluwer Academic Publishers, Boston/Dordrecht/London, 2002.

[24] C. F. Lima, F. G. Lobo, M. Pelikan, and D. E. Goldberg. Model accuracy in the Bayesian optimization algorithm. MEDAL Report No. 2010004, Missouri Estimation of Distribution Algorithms Laboratory (MEDAL), 2010.

[25] C. F. Lima, M. Pelikan, F. G. Lobo, and D. E. Goldberg. *Engineering Stochastic Local Search Algorithms. Designing, Implementing and Analyzing Effective Heuristics*, chapter Loopy Substructural Local Search for the Bayesian Optimization Algorithm, pages 61–75. Springer, Berlin Heidelberg, 2009.

[26] H. Mühlenbein. The equation for response to selection and its use for prediction. *Evolutionary Computation*, 5(3):303–346, 1997.

[27] H. Mühlenbein. Convergence of estimation of distribution algorithms for finite samples. Unpublished manuscript, 2007.

[28] H. Mühlenbein and T. Mahnig. FDA – a scalable evolutionary algorithm for the optimization of additively decomposed functions. *Evolutionary Computation*, 7(4):353–376, 1999.

[29] H. Mühlenbein and G. Paaß. From recombination of genes to the estimation of distributions I. Binary parameters. In H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature - PPSN IV*, volume 1141 of *Lectures Notes in Computer Science*, pages 178–187, Berlin, 1996. Springer Verlag.

[30] B. Naudts and L. Kallel. A comparison of predictive measures of problem difficulty in evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 4(1):1–15, 2000.

[31] M. Pelikan. *Hierarchical Bayesian Optimization Algorithm. Toward a New Generation of Evolutionary Algorithms*. Studies in Fuzziness and Soft Computing. Springer, 2005.

[32] M. Pelikan, D. E. Goldberg, and E. Cantú-Paz. BOA: The Bayesian optimization algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-1999)*, volume I, pages 525–532, Orlando, FL, 1999. Morgan Kaufmann Publishers, San Francisco, CA.

[33] V. Robles, J. M. Peña, M. S. Pérez, and V. Herves. GA-EDA: A new hybrid cooperative search evolutionary algorithm. In J. A. Lozano, P. Larrañaga, I. Inza, and E. Bengoetxea, editors, *Towards a New Evolutionary Computation. Advances in Estimation of Distribution Algorithms*, pages 187–220. Springer Verlag, 2006.

[34] R. Santana, P. Larrañaga, and J. A. Lozano. Protein folding in simplified models with estimation of distribution algorithms. *IEEE Transactions on Evolutionary Computation*, 12(4):418–438, 2008.

[35] G. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 7(2):461–464, 1978.

[36] J. L. Shapiro. Drift and scaling in estimation of distribution algorithms. *Evolutionary Computation*, 13(1):99–123, 2005.

[37] Q. Zhang. On stability of fixed points of limit models of univariate marginal distribution algorithm and factorized distribution algorithm. *IEEE Transactions on Evolutionary Computation*, 8(1):80–93, 2004.

[38] Q. Zhang, J. Sun, and E. Tsang. Combinations of estimation of distribution algorithms and other techniques. *International Journal of Automation and Computing*, pages 273–280, 2007.

[39] Q. Zhang, J. Sun, and E. P. K. Tsang. Evolutionary algorithm with guided mutation for the maximum clique problem. *IEEE Transactions on Evolutionary Computation*, 9(2):192–200, 2005.