

Technical Report

EHU-KZAA-TR-2-2009



Universidad del País Vasco Euskal Herriko Unibertsitatea

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

A quantitative analysis of estimation of distribution algorithms based on Bayesian networks

Carlos Echevoyen, Alexander Mendiburu,
Roberto Santana, Jose A. Lozano

September 2009

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

A quantitative analysis of estimation of distribution algorithms based on Bayesian networks

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

Carlos Echevoyen*, Alexander Mendiburu*, Roberto Santana†
Jose A. Lozano*

*Department of Computer Science and Artificial Intelligence
University of the Basque Country, Donostia-San Sebastián, Spain

†Universidad Politécnica de Madrid

Abstract

The successful application of estimation of distribution algorithms (EDAs) to solve different kinds of problems has reinforced their candidature as promising black-box optimization tools. However, their internal behavior is still not completely understood and therefore it is necessary to work in this direction in order to advance their development. This paper presents a new methodology of analysis which provides new information about the behavior of EDAs by quantitatively analyzing the probabilistic models learned during the search. We particularly focus on calculating the probabilities of the optimal solutions, the most probable solution given by the model and the best individual of the population at each step of the algorithm. We carry out the analysis by optimizing functions of different nature such as Trap5, two variants of Ising spin glass and Max-SAT. By using different structures in the probabilistic models, we also analyze the influence of the structural model accuracy in the quantitative behavior of EDAs. In addition, the objective function values of our analyzed key solutions are contrasted with their probability values in order to study the connection between function and probabilistic models. The results not only show information about the EDA behavior, but also about the quality of the optimization process and setup of the parameters, the relationship between the probabilistic model and the fitness function, and even about the problem itself. Furthermore, the results allow us to discover common patterns of behavior in EDAs and propose new ideas in the development of this type of algorithms.

1 Introduction

Estimation of distribution algorithms (EDAs) [33, 25, 38] are a population-based optimization paradigm in the field of evolutionary computation [15] that has acquired special relevance in the last decade. Nowadays, they are a strong

alternative to solve optimization problems from different domains such as engineering [53], biomedical informatics [1, 46] or robotics [54]. However, despite their successful application there are a wide variety of open questions [47] regarding the behavior of this type of algorithms.

The main characteristic of EDAs is the use of probabilistic models to lead the search towards promising areas of the space of solutions. By making use of a subset of promising solutions belonging to the population, the employed probabilistic models allow to estimate a new probability distribution over the search space at each step of the algorithm. Thus, each of the possible solutions has an associated probability of being sampled which varies during the optimization process. The probability values assigned to the solutions are the main source to determine which solutions will be returned by the algorithm. Consequently, given a problem, the fundamental objective is to get higher probability values for the highest quality solutions throughout an iterative process. Naturally, to reach the optimal solution is an inherent challenge and a reference in the development of both theoretical [55] and practical [42] EDAs.

In order to better understand how these algorithms solve the problems, the characteristics of the probabilistic models used are a rich source of information which has been studied in several works [20, 3, 5, 14, 22, 27, 29]. Particularly, one class of models that has been extensively applied in EDAs are Bayesian networks [37], which allow to encode probability distributions through a structure, that expresses explicit independence relations among variables, and a set of parameters. There exist different implementations of EDAs based on this type of models [16, 40, 30]. When Bayesian networks are used, a straightforward form of analysis is through the explicit interactions among the variables that they provide. Thus, it has been shown how different parameters of the algorithm influence the structural model accuracy [27], how the dependences of the probabilistic models change during the search [22] and moreover how the networks learned can provide information about the problem structure [14, 22, 12].

In order to continue the study of EDAs, we take a different but complementary path which was initiated in a preliminary version [13] of this work. Specifically, we propose a new methodology based on a quantitative analysis of the probabilistic models. As we have argued, the particular probability values assigned to the solutions during the search are the raw material from which EDAs obtains the results. Therefore, studying such probabilities provide new useful information to better understand the behavior of this type of algorithm. Following this criterion, our quantitative analysis of EDAs is based on monitoring the probability of certain distinguished solutions during the search: 1) the optimal solution of the function, 2) the solution with the highest probability in the distribution and 3) the best individual at each population. In order to complete the quantitative analysis, we also record the fitness function values for the solutions 2) and 3) during the search. The proposed analysis is carried out when the estimation of Bayesian network algorithm (EBNA) [16] solves problems of different nature which can have a unique or several optimal solutions. We use different structural models which can be learned from the population or created by reproducing interactions among the variables of the solved problem. And finally, we also use different population sizes in order to analyze the influence of this parameter in the algorithm. Basically, throughout this analysis we are able to study open issues in EDAs such as: 1) how the probability assigned to the optimal solutions evolve during the search, 2) how the structural model accuracy influences their

behavior and 3) how the relationship between the fitness function and probability values is.

The results obtained in the study not only provide information on these key issues but also show a different perspective of EDAs that is able to reveal clear patterns of behavior inside the algorithm. Furthermore, both probability and function values analyzed are able to capture the quality of the probabilistic models in terms of their use within EDAs. Throughout the analysis proposed it is also possible to observe how the convergence of the algorithm occurs and detect multimodality in the problems solved. Finally, based directly on the results, we are able to propose different ideas in order to assist in the use of EDAs in real problems and contribute to their development. The main proposals are related to 1) bringing forward premature convergence, 2) measuring the quality of different components and their impact in the search when the optimal solution is unknown, 3) on-line monitoring of the optimization process allowing certain automatic decision making and 4) how to take advantage of the available information of the problem.

The rest of the paper is organized as follows. Section 2 explains the motivations for this work and raises specific open questions that we want to investigate. Section 3 presents estimation of Bayesian network algorithm (EBNA), and introduces Bayesian networks and abductive inference. Section 4 explains the experimental design. Sections 5, 6, 7 and 8 discuss Trap5, Gaussian Ising, $\pm J$ Ising and Max-SAT problems respectively, quantitatively analyzing the behavior of EDAs for each problem when different structural models and population sizes are used. Section 9 discusses relevant previous works. Section 10 draws the conclusions obtained during the study. Finally, Section 11 points out possible future studies.

2 Motivation

The motivation of this work is twofold. Firstly, our main target is to shed light on certain basic questions of great interest that still remain open in EDAs. Secondly, we argue that, by collecting the new information provided by the proposed analysis, it is possible to discover patterns of behavior and draw general conclusions which will allow us to better understand EDAs and help with their development.

We focus our work on raising the following open questions:

- How does the probability assigned by the probability distributions to the optimal solution evolve during the search?

This first question plays a very important role in this work and it is related with a number of assumptions currently made in the application of EDAs. For example, whether in order to solve a problem, it is a necessary condition that the probability associated by the algorithm to the optimal solution steadily increases at each generation or whether the optimum has assigned the highest probability during the search.

Another important and related question is the following:

- Which probabilistic dependences should the model take into account in order to reach the optimum?

Regarding the structural models, we want to study to what extent it is necessary to represent the original interactions between the variables of the function to achieve an efficient search. By using different structural models in EDAs, we can study the effect that introducing more information of the problem into the structural model has in the behavior of EDAs in general and in the previously mentioned assumptions about the probability of the optimum in particular.

Finally, we complete our quantitative analysis by raising this last question:

- How does the function value for the most probable solution given by the probabilistic models evolve during the search?

A contribution of this work is the exact calculation and analysis of the solution with the highest probability in the distributions generated by an EDA at each generation. Thus, by obtaining its fitness function value and comparing it with the best individual of the population during the search, we can study how the probabilistic model is capturing the properties of the function. It would be desirable that the function value of the solution with the highest probability increases during the search.

3 Background

3.1 Notation

Let X be a random variable, a value of X is denoted x . $\mathbf{X} = (X_1, \dots, X_n)$ will denote a vector of random variables. We will use $\mathbf{x} = (x_1, \dots, x_n)$ to denote an assignment to the variables. Each variable X_i has r_i possible values, $x_i^1, \dots, x_i^{r_i}$. We will work with discrete variables. The joint probability distribution of \mathbf{X} is represented as $p(\mathbf{X} = \mathbf{x})$ or $p(\mathbf{x})$. We use $p(X_i = x_i | X_j = x_j)$ or, in a simplified form, $p(x_i | x_j)$, to denote the conditional probability distribution of X_i given $X_j = x_j$.

3.2 Bayesian networks

Formally, a Bayesian network is a pair (S, θ) representing a graphical factorization of a probability distribution. The structure S is a directed acyclic graph which reflects the set of conditional (in)dependencies among the variables. On the other hand, θ is a set of parameters for the local probability distributions associated with each variable.

The factorization of the probability distribution is codified as:

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

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

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

$$p(x_i^k | \mathbf{pa}_i^j, \theta_i) \equiv \theta_{ijk} \quad (2)$$

where $\mathbf{pa}_i^1, \dots, \mathbf{pa}_i^{q_i}$ denote the q_i possible values of the parent set \mathbf{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 $\boldsymbol{\theta}_i = ((\theta_{ijk})_{k=1}^{r_i})_{j=1}^{q_i}$. The introduction of Bayesian networks in EDAs requires appropriate methods of learning and sampling. As has been shown, to complete the network learning it is necessary to obtain the structure S and the set of parameters $\boldsymbol{\theta}$.

When the structure is not given, it can be learned from a data set. We use a structural learning algorithm based on a “score+search” technique [25]. Particularly, the search is carried out using B algorithm [6] and the score is the Bayesian Information Criterion (BIC) [48]. Regarding the second step, the parameters $\boldsymbol{\theta}$ of the Bayesian network are calculated by maximum likelihood using Laplace correction [25]. Finally, to sample the Bayesian network, a forward sampling method is used. A variable is sampled only when all its parents have been sampled. This method is known as probabilistic logic sampling (PLS) [7].

3.3 EDAs based on Bayesian networks

Algorithm 1: **EBNA**

```

1   $BN_0 \leftarrow (S_0, \boldsymbol{\theta}^0)$  where  $S_0$  is an arc-less structure, and  $\boldsymbol{\theta}^0$  is uniform
2   $D_0 \leftarrow$  Sample  $N$  individuals from  $BN_0$ 
3   $t \leftarrow 1$ 
4  do {
5     $D_{t-1} \leftarrow$  Evaluate individuals
6     $D_{t-1}^{Se} \leftarrow$  Select  $N/2$  individuals from  $D_{t-1}$ 
7     $S_t^* \leftarrow$  Obtain a network structure
8     $\boldsymbol{\theta}^t \leftarrow$  Calculate  $\theta_{ijk}^t$  using  $D_{t-1}^{Se}$  as the data set
9     $BN_t \leftarrow (S_t^*, \boldsymbol{\theta}^t)$ 
10    $D_t \leftarrow$  Sample  $N - 1$  individuals from  $BN_t$  and create the new
      population
11 } until Stop criterion is met

```

Following the main scheme of EDAs, EBNA [16] works with populations of N individuals that constitute sets of N candidate solutions. The initial population is generated according to a uniform distribution, and hence, all the solutions have the same probability to be sampled. Each iteration starts by selecting a subset of promising individuals from the population. Although there are different selection methods, in this case we use truncation selection with threshold 50%. Thus, the $N/2$ individuals with the best fitness value are selected. The next step is to learn a Bayesian network from the subset of selected individuals. Once the Bayesian network is built, the new population can be generated. At this point there are different possibilities. We use an elitist criterion. From the Bayesian network, $N - 1$ new solutions are sampled and then mixed with the N individuals of the current population. The N best individuals, among the $2N - 1$ available, constitute the new population. The procedure of selection, learning and sampling is repeated until a stop condition

is fulfilled. A pseudocode of EBNA is shown in Algorithm 1.

3.4 Abductive inference in Bayesian networks

In general, abductive reasoning tries to find the hypothesis that would best explain a set of facts or observations. In the probabilistic network context, the abductive inference [18] consists of finding the maximum a posteriori probability state of the network variables, given an evidence (observed variables).

Formally, the total abductive inference involves all the problem variables and is defined as follows. Given a probability distribution over the vector of random variables \mathbf{X} and an evidence \mathbf{e} , that is an instance of the observed variable set $\mathbf{E} \subseteq \mathbf{X}$, we want to obtain the assignment $\mathbf{x}_{\mathbf{U}}^*$ to the unobserved variables $\mathbf{X}_{\mathbf{U}} = \mathbf{X} \setminus \mathbf{E}$ such that,

$$\mathbf{x}_{\mathbf{U}}^* = \arg \max_{\mathbf{x}_{\mathbf{U}}} p(\mathbf{x}_{\mathbf{U}} | \mathbf{e}) \quad (3)$$

Usually $\mathbf{x}_{\mathbf{U}}^*$ is known as the *most probable explanation*.

However, when this technique is applied to the probability distributions associated to Bayesian networks in EDAs, there is no evidence. In this case, the objective is to look for the assignment \mathbf{x}^* with the highest probability for the whole set of variables \mathbf{X} . Knowing that $P(\mathbf{X}_{\mathbf{U}} | \mathbf{E}) = P(\mathbf{X} | \mathbf{E})$ and having an empty evidence set $\mathbf{E} = \emptyset$, Equation 3 can be directly converted into our target,

$$\mathbf{x}^* = \arg \max_{\mathbf{x}} p(\mathbf{x}) \quad (4)$$

In our context of EDAs, \mathbf{x}^* will be called the *most probable solution* (MPS). As it is proven in [52], this kind of inference is an NP-hard problem. Therefore, its exact resolution is only feasible in problems of limited length.

In this work, the point with the highest probability is exactly calculated using probability propagation in junction trees [7] or variable elimination techniques [11], as they are implemented in Bayes Net Toolbox [34].

4 Experimental design

The experiments are mainly designed in order to shed light on questions and assumptions mentioned in Section 2. Specifically, at each generation of EBNA, we record the probability and fitness values of distinguished solutions of the search space: the optimum, the most probable solution and the best individual in the population. We analyze these values throughout the search process completed by EBNA for different functions, structural models and population sizes. We also take into account both successful (the optimum is reached) and unsuccessful runs (the optimum is not reached).

In addition to our distinguished solutions, we also introduce useful information into the analysis. Particularly, at each step of the algorithm we calculate the accumulated entropy of the population by means of adding the entropy of each variable belonging to the function,

$$H(\mathbf{X}) = - \sum_{i=1}^n \sum_{j=1}^{r_i} p(x_i^j) \cdot \log_2 p(x_i^j) \quad (5)$$

This metric allows to reflect how the population managed by the algorithm is converging and also provides interesting information when EBNA solves multimodal problems. We show the close relationship between our analyzed probability values and the accumulated entropy of the population. Some works have already studied these types of measurements in EDAs [36, 35] in detail.

In the following section we will explain the different problems, structural models and parameters used for the experiments. In [44], the necessary tools to reproduce the experiments or to carry out similar analysis are implemented.

4.1 Problems

The whole set of problems is based on additively decomposable functions (ADFs) defined as,

$$f(\mathbf{x}) = \sum_{i=1}^m f_i(\mathbf{s}_i) \quad (6)$$

where $\mathbf{S}_i \subseteq \mathbf{X}$. Trying to cover a wide spectrum of applications and observe the behavior of EDAs in different scenarios, we chose the following four test problems: Trap5, Gaussian Ising, $\pm J$ Ising and Max-SAT. The exact details of each one are introduced in the following sections. These problems are selected for several reasons. Firstly, in order to investigate the influence of multimodality in the behavior of EDAs, we deal with problems that have different numbers of optimal solutions. The first two problems have a unique optimum and the last two problems have several optima. Secondly, all of them are optimization problems which have been widely used to analyze EDAs [22, 39, 4]. And finally, all the problems have a different nature. Trap5 [10] is a deceptive function designed in the context of genetic algorithms [19] aimed at finding their limitations. It is a separable function and in practice can be easily optimized if the structure is known. Gaussian Ising and $\pm J$ Ising come from statistical physics domains and are instances of the Ising model proposed to analyze ferromagnetism [24]. The variables are disposed on a grid and the interactions do not allow dividing the problem into independent subproblems of bounded order [32] efficiently. It is a challenge in optimization [22, 39] and in its general form is NP-complete [2]. Max-SAT is a variation for optimization of a classic benchmark problem in computational complexity, the propositional satisfiability or SAT. In fact, SAT was the first problem proven to be NP-complete [9] in its general form. An instance of this problem can contain a very high number of interactions among variables and in general, it can not be efficiently divided in subproblems of bounded size in order to reach the optimum. Except for the function Trap5, we are dealing with five instances for each type of problem.

Regarding the information store at each generation, when the functions with a unique optimum are optimized, we only record our three distinguished solutions. However, in the functions with several optima, EBNA reaches a subset of them and the analysis of the probability of the optimum must be extended. Thus, we calculate the probabilities during the search for all optima

reached by EBNA in the last generation. It leads us to see how the probability is distributed when there are different optimal solutions. In order to gain clarity in the results, we only show the maximum and minimum probabilities assigned by the probability distribution to the reached optimal solutions at each generation.

4.2 Structural models

In the literature, several works have been presented showing the influence of the structure of the probabilistic model in the behavior of the algorithm [12, 21]. In this paper, we also take into account this important fact creating structures which try to represent the dependences among the variables in each problem. In this sense, a utopian objective could be to find structures with the lowest computational complexity (maximum number of independence relations among variables) which would allow to reach the optimum with the maximum certainty.

Particularly, we use two approaches in order to obtain structural models. On the one hand, we use an approximate structural learning algorithm (B algorithm) which obtains a new structure from the selected individuals at each generation. This is a common way of solving optimization problems by means of EDAs for which there is no structural information available. On the other hand, we use two fixed structures related to the nature of the function and from which only parametric learning is carried out. Since all the functions are ADFs, an intuitive way to create a related structural model is by means of linking variables belonging to the same subfunction with arcs. In order to analyze the influence of the structural model accuracy, we use two structures with different amounts of information. The first structure tries to reflect all the interactions among the variables of the function. As a general method we connect two variables (representing nodes in the graph) by an edge in the structure if the corresponding variables are contained in the same sub-function. Thus, by providing direction to the edges without creating cycles, we obtain a Bayesian network structure which will be called *complete structure*. The second structure reproduces interactions among the variables of the function but only considering bivariate dependences. Depending on the problem, we select which dependences are introduced in the model. This structure has less information but it is always related with the nature of the problem. It will be called *bivariate structure*.

4.3 Parameter configuration

The sample size is very important in order to learn Bayesian networks [17] and, hence, in the behavior of EDAs based on this type of models. Thus, we use two different population sizes in order to analyze their influence in the algorithm. Firstly, we have used the bisection method [38] to determine an adequate population size to reach the optimum (with high probability). This size is denoted by m . The stopping criterion for bisection is to obtain the optimum in 10 out of 10 independent runs. The final population size is the average over 20 successful bisection runs. The population size m is always obtained from EBNA executions with B algorithm. The second population size is half of the bisection, $m/2$. With this size we try to create a more realistic scenario in which achieving the optimum is less likely. Thus, we can analyze the probability of the optimum when it is not reached.

We have observed through the analysis that the population size is less influential in the behavior of the algorithm when the structural model is fixed during the search. Thus, in these cases we only show the results for a unique population size because analyzing the difference between sizes does not provide relevant information. Depending on the problem, we show the analysis for the population size with the most useful information.

The experiments have been performed with problems of dimension $n = 100$. The stopping criterion for EBNA is a fixed number of iterations and it is independent of obtaining the optimum. Each execution will run n generations, that is, as many as the number of problem variables. This number of generations is enough to observe the convergence of the algorithm in most of the experiments.

4.4 Details of the results

We usually report the probability values in logarithmic scale in order to smoothen the probability slopes and better observe their behavior from the beginning of the run. In some cases, we also report the original probability values because this helps to better show the EDA behavior. This is especially useful in problems with several optimal solutions.

The number of runs which have reached the optimum at each generation is indicated with bars on the top of the charts where the probability values in logarithmic scale are shown. Although we have made runs with a fixed number of generations, the charts presented usually are cut when all runs have reached the optimum or the curves shown are stabilized.

Finally, for each experiment type i.e. EBNA solving a problem with a given structural learning approach and a given population size, 50 independent runs have been carried out. Each set of 50 executions is divided into successful and unsuccessful executions which will be analyzed separately.

The whole set of results can not be presented in this paper for the sake of simplicity. Therefore, we only show the most relevant and informative results for each problem. The complete analysis is available on the website of the Intelligent Systems Group¹.

5 EDA behavior solving Trap5

5.1 Trap5 description

Our first function, Trap-5 [10], is an additively separable (non overlapping) function with a unique optimum. It divides the set \mathbf{X} of n variables, into disjoint subsets \mathbf{X}_I of 5 variables. It can be defined using a unitation function $u(\mathbf{y}) = \sum_{i=1}^p y_i$ where $\mathbf{y} \in \{0, 1\}^p$ as,

$$\text{Trap-5}(\mathbf{x}) = \sum_{I=1}^{\frac{n}{5}} \text{trap}_5(\mathbf{x}_I) \quad (7)$$

where trap_5 is defined as,

$$\text{trap}_5(\mathbf{x}_I) = \begin{cases} 5 & \text{if } u(\mathbf{x}_I) = 5 \\ 4 - u(\mathbf{x}_I) & \text{otherwise} \end{cases} \quad (8)$$

¹http://www.sc.ehu.es/ccwbayes/members/carlos/eda_probs/

and $\mathbf{x}_I = (x_{5I-4}, x_{5I-3}, x_{5I-2}, x_{5I-1}, x_{5I})$ is an assignment to each trap partition \mathbf{X}_I . This function has one global optimum in the assignment of all ones for \mathbf{X} and a large number of local optima, $2^{n/5} - 1$.

Trap5 function has been used in previous works [22] to study the structure of the probabilistic models in EDAs based on Bayesian networks as well as the influence of different parameters [27]. It is important to note that this function is difficult to optimize if the probabilistic model is not able to identify interactions between variables [13].

5.2 Structures related to the problem

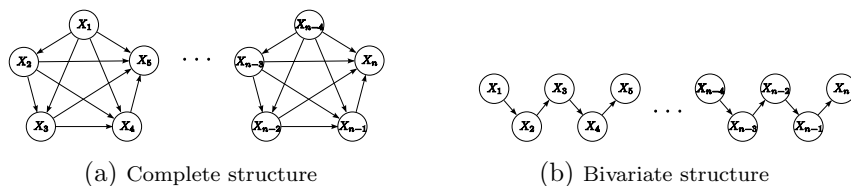


Figure 1: Fixed structural models related with the dependences among the variables in Trap5.

In this section we propose two fixed structures related with the Trap5 function. The complete structure is created by linking all the variables in each sub-function *trap5*. Thus, by providing direction to the arcs without creating cycles, we obtain the Bayesian network structure shown in Fig. 1(a). With this structure, there are no independences between variables of the same subgroup and variables in different partitions are independent. For this function, the complete structure provides an exact factorization [32]. On the other hand, the bivariate structure is formed by a chain for each subgroup of 5 variables. As can be seen in Fig. 1(b), the graph contains the minimum number of arcs necessary to connect all the variables belonging to each partition.

5.3 Using structural learning

In this section, we present and discuss the results obtained when EBNA tries to optimize the Trap5 function using B algorithm for the structural learning at each generation.

The results for successful runs are presented in Fig. 2. When the population size m (given by bisection) is used, EBNA reaches the optimum in 49 runs out of 50. Fig. 2(a) shows the probability values for the successful executions in logarithmic scale and Fig. 2(b) illustrates the original growth of the probability values. Theoretically, a convergence of the algorithm to the global optimum implies an increase in its probability value as the generations advance and this is reflected in the results. The probability values for the optimum and the most probable solution (MPS) grow simultaneously and very closely when the executions are successful. When the population size is decreased to $m/2$ the results drastically change and only 4 runs have reached the optimum. However, the behavior of the probability values in Fig. 2(d) are analogous to Fig. 2(a). Thus, when the optimum is achieved, its probability tends to 1 (Fig.2 (e)) and

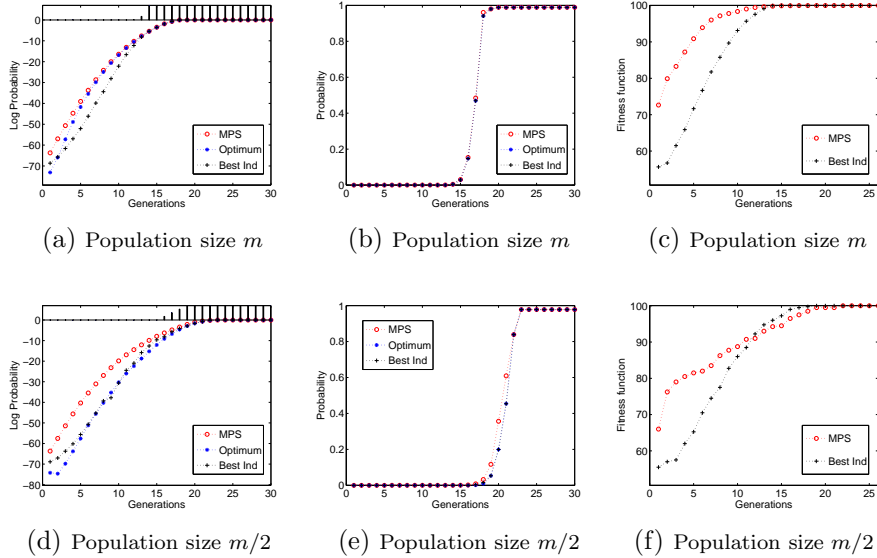


Figure 2: Successful runs when EBNA is applied to Trap5 using Algorithm B. We have 49 out of 50 successful runs with population size m and 4 out of 50 with population size $m/2$

converges with the MPS. However, we can observe that the probability curves for the MPS and the optimum are clearly more distant with size $m/2$. Another important observation is that the executions start to reach the optimum (bars on the top of Fig. 2(a) and 2(d)) when its probability is approximately -10 in logarithmic scale.

The results for the executions where the optimum was not reached are shown in Fig. 3. Fig. 3(a) only represents the probability values for 1 execution but there are no oscillations in the curve. This indicates that the behavior of the probability values has a constant pattern. In Fig. 3(a) and 3(d) we can see the joint growth of the probability values for the MPS and the optimum at the beginning of the run. However, after a certain generation, both values start to diverge and the optimum is not obtained. If we observe Fig. 3(b) and 3(e), we can see that the rapid decline in the probability of the optimum takes place when the exponential growth of the probability values for the MPS and the best individual occur. Furthermore, there is a clear difference between the runs with population size m and $m/2$. For this last population size, the probability of the optimum clearly decreases to lower values. Even at the beginning of the run, its probability values are far from the highest probability in the distribution. In Fig. 4 we present the accumulated entropies of the population during the search. How this measure is related with the exponential growth of the probability values can be seen. Moreover, we can observe how the population converges to a unique solution when EBNA uses Algorithm B since the entropy tends to 0.

Concerning the fitness function value, in Fig. 2(c) we show the results when the population size m , given by bisection, is used in EBNA. The MPS value increases at each generation and it is better than the best individual in almost

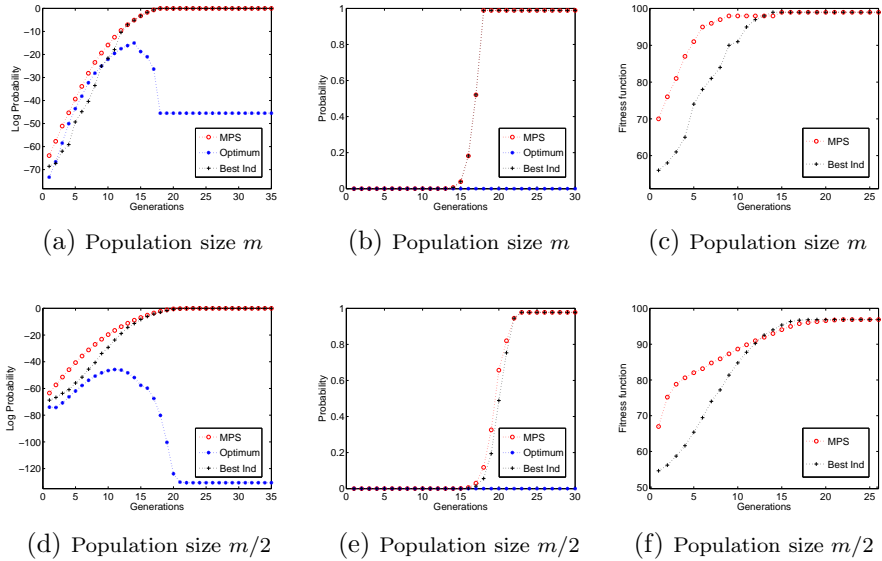


Figure 3: Unsuccessful runs when EBNA is applied to *Trap5* using Algorithm B. We have 1 out of 50 unsuccessful runs with population size m and 46 out of 50 with population size $m/2$.

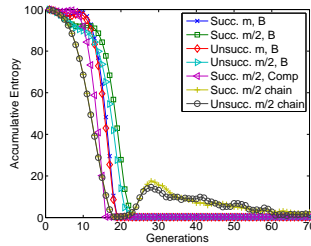
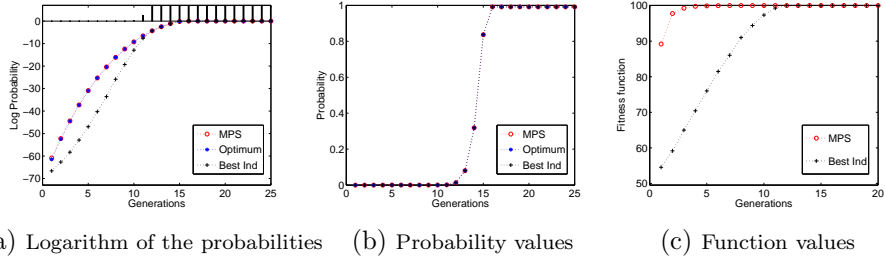


Figure 4: Accumulated entropies of the population when EBNA solves *Trap5*

all generations. However, by looking at Fig. 2(f) we can observe that the MPS has a lower growth with population size $m/2$. Moreover, the best individual of the population is better than the MPS after generation 12 approximately. For unsuccessful runs, the curves of fitness function values presented in Fig. 3(c) and 3(f) are similar to those shown for successful executions. This behavior is constant in all the problems analyzed in this work. It is also interesting to note that the fitness function value for MPS never decreases in the results presented.

5.4 Using fixed structures

In this section, we show the behavior of the algorithm when a different amount of information is introduced in the structural model. We only show the analysis with population size $m/2$ because it is sufficient to provide relevant results in this group of experiments.

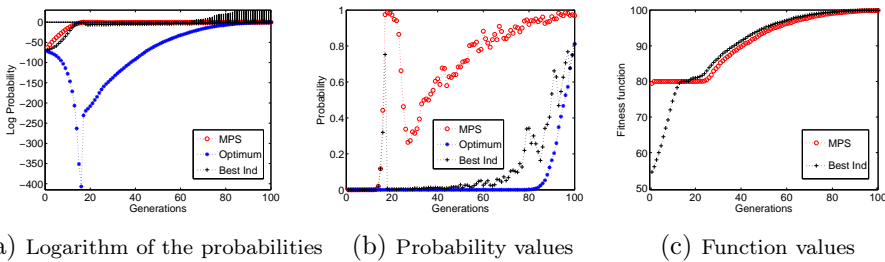


(a) Logarithm of the probabilities (b) Probability values (c) Function values

Figure 5: Successful runs when EBNA is applied to Trap5 using the complete structure with population size $m/2$. The optimum is reached for the 50 executions.

The probability and fitness values when the complete structure is introduced in EBNA are shown in Fig. 5. In this case we obtain an ideal behavior for an optimization process with EDAs because the optimum has the highest probability during the whole run and it is found in all executions. Furthermore, in Fig. 5(c) we can observe that the function values for MPS are close to the optimum from the first steps of the algorithm.

The behavior of the algorithm changes drastically when the bivariate structure (Fig. 1(b)) is introduced. Although EBNA has a good performance because it reaches the optimum 38 out of 50 runs, the evolution of the probability values in Fig. 6(a) is particular. The probability of the optimum decreases at the beginning of the run and when the algorithm seems to converge to a local optimum, it suddenly recovers. This fact also occurs for the unsuccessful executions in Fig. 7 where the probability of the optimum increases in the last generations. In this case, the optimum has a high probability at the end of the run, which supports the belief that waiting for more generations, the algorithm would be able to reach the optimum.



(a) Logarithm of the probabilities (b) Probability values (c) Function values

Figure 6: Successful executions when EBNA is applied to Trap5 using the bivariate structure with population size $m/2$. We have 38 out of 50 successful executions.

The reason for such an uncommon behavior is as follows. In the first part of the run, when the probability of the optimum decreases, the algorithm is deceived by the function and most of the individuals in the population become the local optimum. This local optimum is the assignment of zeros for all the set of variables \mathbf{X} because it is the suboptimal value that $trap_5$ function gives

to each trap partition \mathbf{X}_I . We can observe in Fig. 4 how the curves of entropy for the bivariate model (Succ. $m/2$ chain and Unsucc. $m/2$ chain) tend to 0 when the probability of the optimum is minimum in Fig. 6(a) and 7. After this stage, the algorithm recovers and the probability of the optimum starts to increase. The curves of the bivariate model in Fig. 4 shows how different individuals are included in the population just when the algorithm seems to converge to the local optimum. It indicates that the algorithm samples better individuals and it is reflected in the the fitness function values in Fig. 6(c). This fact can be explained through the Laplace correction and the fixed structure of chain subgraphs. This quantitative analysis justifies why it is possible to reach the optimum for this function with a simple bivariate structure.

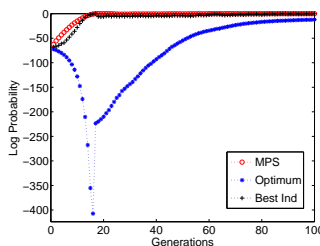


Figure 7: Logarithm of the probabilities for 12 unsuccessful executions when EBNA is applied to Trap5 using the bivariate structure with population size $m/2$.

6 EDA behavior solving Gaussian Ising

6.1 2D Ising spin glass description

Ising spin glass is an optimization problem which has been solved and analyzed in different works related with EDAs [22, 39, 41]. A classic 2D Ising Spin Glass can be formulated in a simple way. The set of variables \mathbf{X} is seen as a set of n spins disposed on a regular 2D grid L with $n = l \times l$ sites and periodic boundaries (see Fig. 8). Each node of L corresponds to a spin X_i and each edge (i, j) corresponds to a coupling between X_i and X_j . Thus, each spin variable interacts with its four nearest neighbors in the toroidal structure L . Moreover, each edge of L has an associated coupling strength J_{ij} between the related spins. For the classical Ising model each spin takes the value 1 or -1 . The target is, given couplings J_{ij} , to find the spin configuration that minimizes the energy of the system computed as,

$$E(\mathbf{x}) = - \sum_{(i,j) \in L} x_i J_{ij} x_j - \sum_{i \in L} h_i x_i \quad (9)$$

where the sum runs over all coupled spins. In our experiments we take $h_i = 0 \forall i \in L$. The states with minimum energy are called *ground states*.

Depending on the range chosen for the couplings J_{ij} we have different versions of the problem. Thus, the problem is called *Gaussian Ising* when the couplings

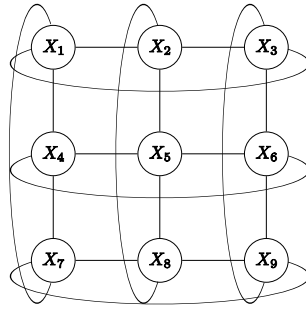


Figure 8: A 3×3 grid structure L showing the interactions between spins for a 2D Ising Spin Glass with periodic boundaries. Each edge has an associated strength J_{ij} .

J_{ij} are real numbers generated following a Gaussian distribution. For this type of couplings, the problem has only one optimum. A specified set J_{ij} of coupling defines a spin glass instance. We generated 5 Gaussian Ising instances using the Spin Glass Ground State server² for the experiments. The minimum energy of the system is also provided in this server.

6.2 Structures related to the problem

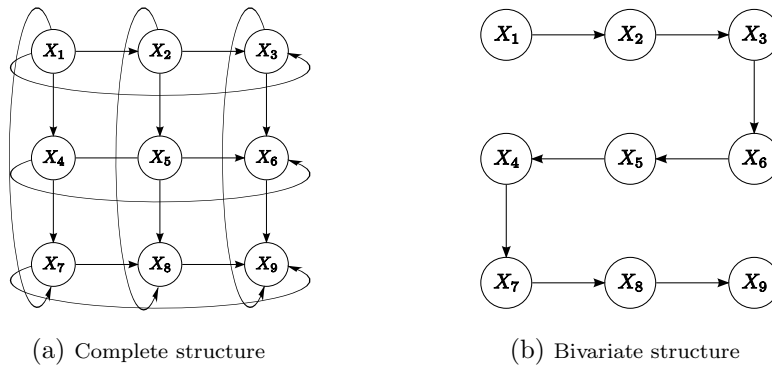


Figure 9: Fixed structural models for 2D Ising spin glass.

In order to create a complete structure for this problem, we reproduce the undirected graph L in the model, which represents all the interactions among variables in the function, and direct the edges without creating cycles to obtain a Bayesian network. Starting from the first spin (variable X_1) we give a westward and southward direction to the edges as can be seen in Fig. 9 (a).

For this problem the direction of the arcs could modify the behavior of EBNA because we would have different Bayesian networks and therefore different independence relations among the variables [7]. In this problem, the complete structure does not provide an exact factorization [32].

²http://www.informatik.uni-koeln.de/ls_juenger/index.html

The second structure is a simple model which connects all variables using a chain. This structure introduces very few interactions related with the problem, as can be seen in Fig. 9 (b).

6.3 Using structural learning

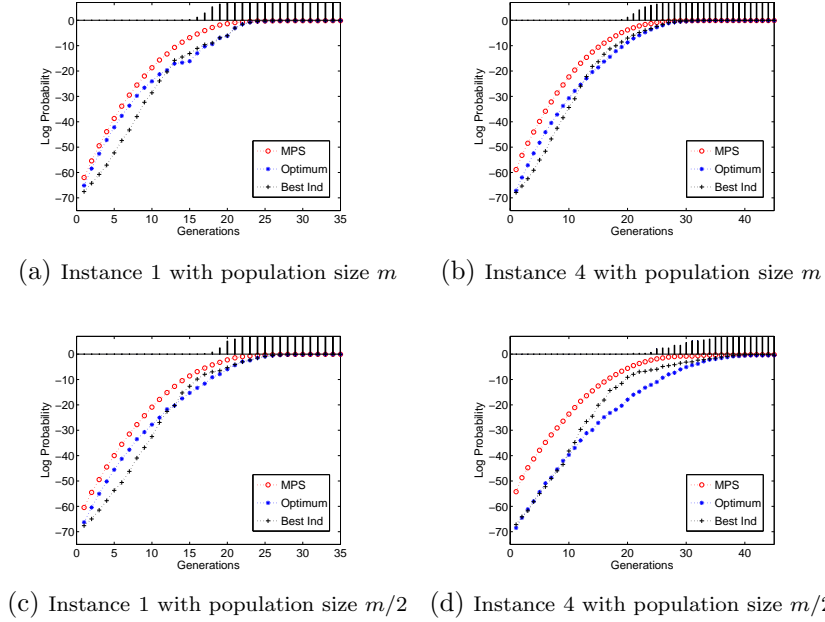


Figure 10: Logarithm of the probabilities in successful runs when EBNA is applied to Gaussian Ising using B algorithm.

The results obtained when EBNA solves Gaussian Ising problem by learning a new structural model at each generation are shown and discussed in this section. In Table 1 we summarize the number of runs that have reached the optimum for each instance and probabilistic model. Firstly, if we look at the number of successful runs with Algorithm B, we can see that the population size is less decisive than in Trap5. In this problem, we achieve a relatively high number of successful runs when population size $m/2$ is used (compare 4 out of 50 in Trap5 with $m/2$). We will see how this fact is reflected in the analysis. Particularly, in this section we present the results for instances 1 and 4 because they are sufficiently representative of the EBNA behavior.

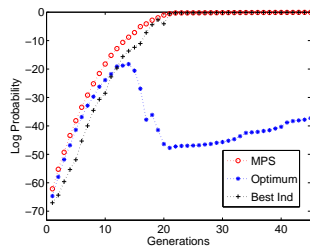
In order to show the probability curves in successful runs we present Fig. 10. In general, we can observe that the behavior is analogous to that in Trap5: the probability of the optimum increases during the search being the most probable solution at the end of the run, and for an adequate population size m , the curves for the MPS and the optimum are closer than for $m/2$. However, in Gaussian Ising, this difference between the probability curves for both population sizes is lower than for Trap5 (Fig. 2(a) and 2(d)). Therefore, the population size is less critical for Gaussian Ising and it is reflected in the performance of the algorithm (Table 1). The curves of probability with the population size given by bisection

Table 1: Number of EBNA runs that have reached the optimum from 50 executions of different Gaussian Ising instances. We report the results for different structural models and population sizes

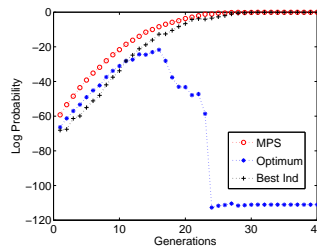
Instances G. Ising	B alg.		Complete		Bivariate	
	m	$m/2$	m	$m/2$	m	$m/2$
I1	46	34	50	47	1	0
I2	47	31	44	37	0	0
I3	49	36	37	29	0	0
I4	46	20	50	46	7	5
I5	48	28	10	6	0	3

makes another difference between these two problems. In Trap5 (Fig. 2), the probability of the optimum is approximately the highest in the distribution from the beginning of the run. However, in Gaussian Ising (Fig. 10(a) and 10(b)), despite using an adequate population size, the probability of the optimum keeps a visible distance from the MPS throughout the generations.

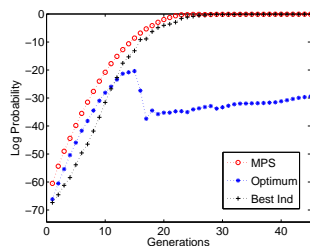
For these problems, the runs do not reach the optimum (bars on the top of the charts in Fig. 10 and 2) before their probability value exceeds approximately the threshold of -20 in logarithmic scale.



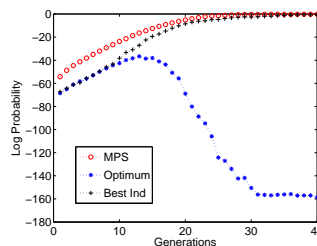
(a) Instance 1 with population size m



(b) Instance 4 with population size m



(c) Instance 1 with population size $m/2$



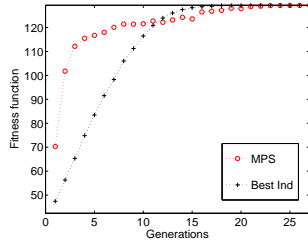
(d) Instance 4 with population size $m/2$

Figure 11: Logarithm of the probabilities in unsuccessful runs when EBNA is applied to Gaussian Ising using B algorithm.

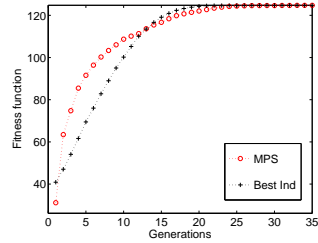
The analysis of the probability values for unsuccessful runs is presented in Fig. 11. For instance 1, the probability of the optimum with population size m (Fig. 11(a)) and $m/2$ (Fig. 11(c)) decreases to similar values at the end of the

runs. On the other hand, the results for instance 4 with different population sizes (Fig. 11(b) and 11(d)) show a difference in the final probability value reached by the optimum between m and $m/2$. For the second size, these probability values are clearly lower. This fact reflects that for instance 4 the population size is more decisive in the EBNA performance (Table 1). In general, in unsuccessful runs we observe the same behavior as in Trap5: the growth in the probability of the optimum at the beginning of the runs and its subsequent fall in few generations. Moreover, in this type of executions, the probability of the optimum decreases before reaching approximately the value of -20 in logarithmic scale.

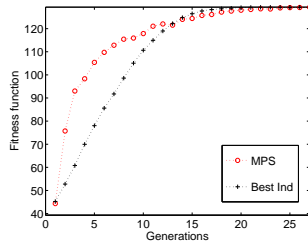
In Gaussian Ising the probability values have a similar exponential increase after a certain generation as in Trap5. This moment of the run is also related with the marked decrease in the probability of the optimum in unsuccessful runs. In the same way, the accumulated entropy reflects the convergence of the algorithm to a unique solution. However, these charts are not presented here because they do not provide new relevant information.



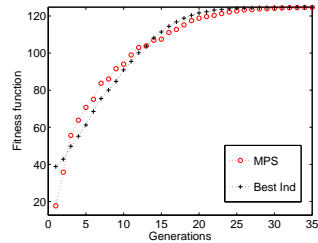
(a) Instance 1 with population size m



(b) Instance 4 with population size m



(c) Instance 1 with population size $m/2$



(d) Instance 4 with population size $m/2$

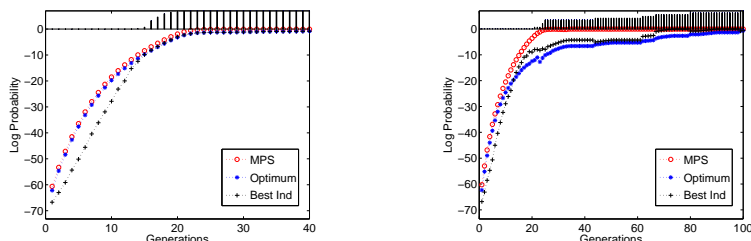
Figure 12: Fitness function values in successful runs when EBNA is applied to Gaussian Ising using B algorithm.

Regarding the fitness function values, the results for this type of analysis are similar between successful and unsuccessful runs. This fact was shown for Trap5 and is repeated in the rest of the problems. Therefore, from now on, we only show the results for successful runs when the analysis of function values is discussed. For Gaussian Ising, these results are presented in Fig. 12. An important fact is the different behavior of EBNA between population size m given by bisection (Fig. 12(a) and 12(b)) and with the half $m/2$ (Fig. 12(c) and 12(d)). When we use an adequate population size to reach the optimum, the difference in function value between the MPS and the best individual is bigger at the beginning of the runs.

However, as we previously discussed, EBNA has a different behavior depending on the problem that it is solving and this is also reflected in the analysis of the function values. We can compare Gaussian Ising (Fig. 12) with Trap5 (Fig. 2(c) and 2(f)) in order to see the difference. This analysis of the function values shows that: (1) the population size is less influential in Gaussian Ising than in Trap5 and (2) with the population size given by bisection the difference between MPS and the best individual is less marked in Gaussian Ising.

6.4 Using fixed structures

At this point we analyze the probabilistic behavior of EBNA when it uses the proposed fixed structures to solve the Gaussian Ising problem. As we discussed in Section 4, the population size is less influential when EBNA works with fixed structures. This fact is reflected in the number of successful runs shown in Table 1. In this section we only present the results for the population size m given by bisection.



(a) Instance 1 with population size m (b) Instance 5 with population size m

Figure 13: Logarithm of the probabilities in successful runs when EBNA is applied to Gaussian Ising using the complete structure.

First for all, by looking at Table 1, we can see that the complete structure does not always reach the optimum as in Trap5. In fact, for instance 5, EBNA with this structure provides a poor performance which can be explained with the quantitative analysis. In order to illustrate the behavior of EBNA in successful runs with the complete structure, we use instances 1 and 5. In Fig. 13(a) we report the analysis of the probability values for the instance 1 where EBNA always reaches the optimum. In this case, the complete structure has a high performance but it is not perfect as in Tra5 (Fig. 5(a)) because the optimum is not the MPS during the run.

However, we must say that the difference between the probability curves of the optimum and the MPS is really small. On the other hand, in Fig. 13(b) we report a poor behavior of EBNA based on the complete structure solving instance 5. At the beginning, the probability of the optimum is very close to the probability of the MPS but in a few generations both probabilities start to be clearly separated. This is probably due to the criterion for directing the arcs. Different directions can create different independence relations among the variables. Therefore, depending on the instance, one selected direction could be better than another.

In Fig. 14 we report the results when EBNA has not reached the optimum

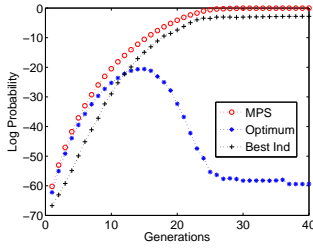


Figure 14: Logarithm of the probabilities in unsuccessful runs when EBNA solves instance 5 of Gaussian Ising using the complete structure with population size m

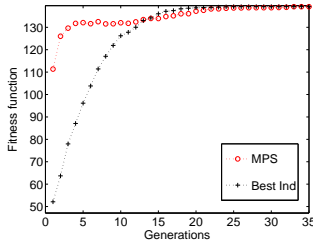


Figure 15: Fitness function values in successful runs when EBNA is applied to instance 5 of Gaussian Ising using the complete structure with population size m .

with the complete structure, for instance 5. The behavior is similar to other unsuccessful executions. It is very interesting to note that the probability of the optimum is never higher than -20 and starts to decrease after generation 10 following a similar pattern than in the rest of the analysis.

The function values for the MPS and the best individual at each population can be seen in Fig. 15. We only show the results for instance 5 where EBNA has a low performance. However, in spite of this performance, the function value for the MPS is close to the optimum from the beginning of the run. It indicates that the search could be improved by using, for example, a more accurate sampling when information about the problem is available. The behavior of these function values is always similar for the different instances when a complete structure is used.

In this problem, our bivariate structure of chain has not had an outstanding behavior but it is very important to observe that with these simple structures the optimum has sometimes been reached (Table 1). Fig. 16 shows the analysis for instance 4, where EBNA has achieved the best performance using the bivariate structure. We can see that the probability of the optimum is separated from the highest probability during the run. This fact is related to the low number of successful runs. As regards to the unsuccessful runs shown in Fig. 16(b), we observe that the probability of the optimum does not reach high probability values before decreasing. Its probability starts to decrease when the probability

of the best individual has the value of, approximately -30 .

In this case, the function value for the MPS, presented in Fig. 17, is again better than the best individual of the population at the beginning of the run. Moreover, the behavior of these values is comparable with the corresponding values obtained with Algorithm B and population size $m/2$ in Fig. 12(d).

To conclude this section we want to highlight an interesting pattern of behavior. We have observed that the curves of probability values for the best individual and the MPS increase in the same way with different structural models and population sizes.

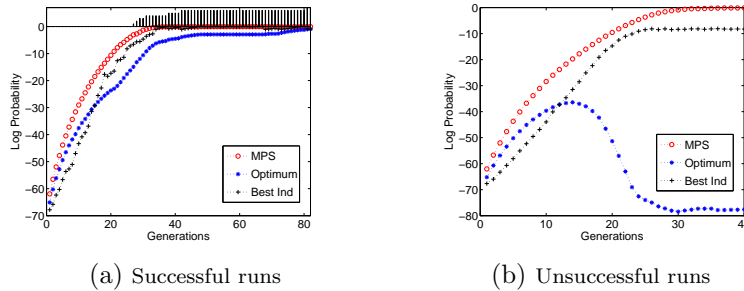


Figure 16: Logarithm of the probabilities when EBNA is applied to instance 4 of Gaussian Ising problem using the bivariate structure.

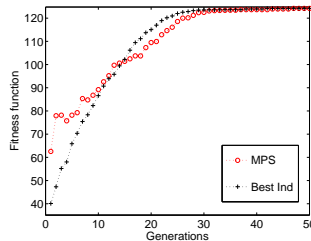


Figure 17: Fitness function values in successful runs when EBNA is applied to instance 4 of Gaussian Ising problem using the bivariate structure.

7 EDA behavior solving $\pm J$ Ising

7.1 $\pm J$ Ising description

As explained in Section 6, the main difference between both version of $2D$ Ising spin glass that we use, is the range of values chosen for the couplings J_{ij} . In this second type Ising problem, the couplings J_{ij} are set randomly to either $+1$ or -1 with equal probability. This version, that will be called $\pm J$ Ising, could have different configurations of the spins that reach the ground state (lowest energy) and therefore many optimal solutions may arise. As for the previous case, 5 $\pm J$ Ising instances were generated using the Spin Glass Ground State server. This

server also provided the value of the minimum energy of the system. Concerning the fixed structural models, we use the same structures as in Gaussian Ising because the problem has the same nature.

7.2 Using structural learning

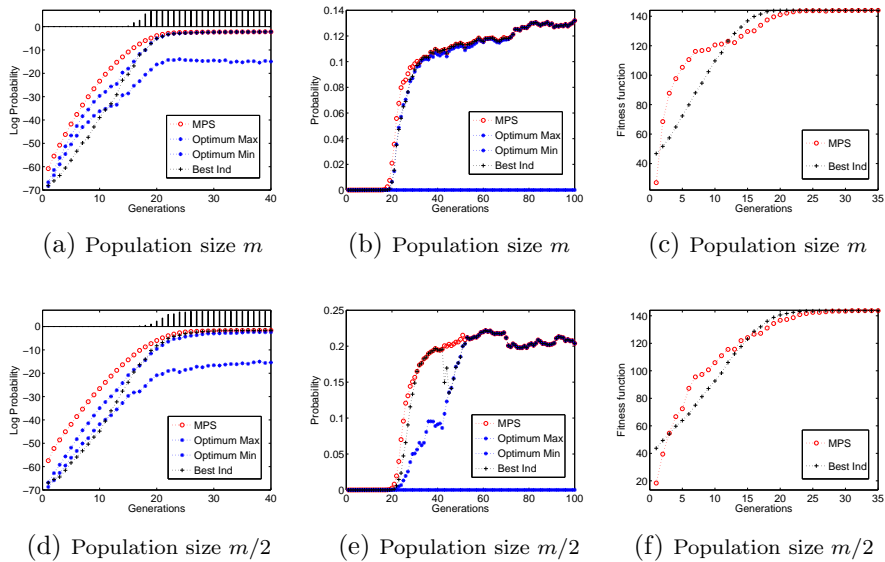


Figure 18: Successful runs when EBNA is applied to instance 5 of $\pm J$ Ising using B algorithm.

Firstly, in Table 2 we report the performance of EBNA given by the number of successful runs for each $\pm J$ Ising instance and structural model. By comparing this table with Table 1, where the successful runs are shown for Gaussian Ising, we can observe a similar pattern when a structural learning is introduced in the algorithm. However, clear differences appear in the probability values for our distinguished solutions showed in Fig. 18 when EBNA uses Algorithm B. In this case, we selected instance 5 because its results are representative. For this problem, we also report the probability values in addition to the logarithmic scale. It allows us to show the influence of multimodality in the algorithm during the search. In addition, we also show in Table 3 the number of optimal solutions that EBNA has reached on average at the end of the runs.

By looking at Fig. 18(a) and 18(d) we can verify that the probabilities for the reached optima increase during the generations. Moreover, the highest probability assigned to the set of optima tends to be the highest in the distribution. We can observe a close relationship between the maximum and minimum probabilities assigned to the optimal solutions where both have a similar increase during the search. In this sense, its behavior is analogous to problems with a single optimum. However, in this case we can clearly see that the probability of the MPS does not tend to 1 (Fig. 18(b) and 18(e)). This indicates that the probability distribution is shared out among different optimal solutions at the end of the run. This fact is verified by the accumulated entropy of the population

Table 2: Number of EBNA runs that have reached the optimum out of 50 executions of different $\pm J$ Ising instances. We report the results for the different structural models and population sizes

Instances $\pm J$ Ising	B alg.		Complete		Bivariate	
	m	$m/2$	m	$m/2$	m	$m/2$
I1	47	30	33	27	10	8
I2	47	25	49	42	4	1
I3	48	30	26	17	0	0
I4	50	18	46	42	4	4
I5	49	27	50	47	17	20

Table 3: On average, number of optimal solutions that EBNA has reached at the end of 50 executions when it is applied to $\pm J$ Ising. We report the results for the different structural models and population sizes

Instances $\pm J$ Ising	B alg.		Complete		Bivariate	
	m	$m/2$	m	$m/2$	m	$m/2$
I1	18	11	6	6	9	8
I2	121	63	134	60	22	17
I3	78	42	49	52	0	0
I4	54	33	32	22	12	10
I5	75	39	107	54	36	26

shown in Fig. 20. It is greater than 0 in the last generations and therefore the population contains different solutions. In this case, the exponential growth of the probability values after a certain generation can also be seen and it is related with the accumulated entropy.

By looking at Table 3, it can be seen that when the population size is reduced, EBNA reaches a lower number of optima. This fact can be seen mainly in two aspects in the analyzed probability values. Firstly, with population size m (Fig. 18(a)), the maximum probability assigned to the optima is closer to the highest in the distribution than with $m/2$ (Fig. 18(d)). Secondly, the final probability values for the MPS are lower with population size m (Fig. 18(b)) than with $m/2$ (Fig. 18(e)) because EBNA reaches a higher number of optimal solutions. In $\pm J$ Ising, the behavior of the curve for the maximum probability assigned to the optima is similar to the curve of the optimum in problems with only one. In this case, the runs do not reach any optimal solution before the maximum probability values does not exceed, approximately -20 in logarithmic scale.

We also show an example of the probabilistic behavior of EBNA when no optimum is reached. In this case, we only can record the probability of the optima found by the algorithm in the successful runs because the whole set of optima is unknown. Fig. 19 shows the analysis when EBNA is applied to instance 5 with population size $m/2$. We can observe an analogous pattern to that in problems with a unique optimum. The probabilities of the optima

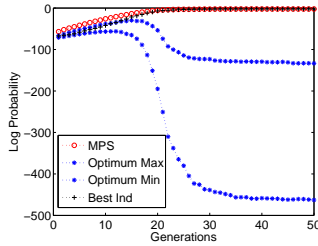


Figure 19: Logarithm of the probabilities in *unsuccessful* runs when EBNA is applied to instance 5 of $\pm J$ Ising using Algorithm B and population size $m/2$

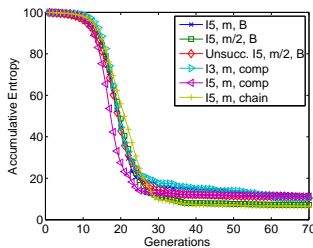


Figure 20: Accumulated entropy of the population when EBNA solves different instances of $\pm J$ Ising using our three types of structural models

increase at the beginning of the run and after a certain generation they decrease quickly. In this case, the optimal solutions analyzed have a very low probability at the end of the search. If we look at the entropy curve for this type of runs (Unsucc. I5, $m/2$, B) in Fig. 20, we can see that this curve does not tend to 0 and therefore, there are different solutions in the population at the end of the run. This indicates that for unsuccessful runs the probability is also distributed among different solutions at the end of the run. For the problems with several optima we will not discuss more results for unsuccessful runs because their analysis does not provide new relevant information.

Regarding the function values presented in Fig. 18(c) and 18(f), the MPS has a similar behavior as in the rest of the analyzed problems. At the beginning of the run it is better than the best individual of the population and this difference is lower when the population size is reduced. However, after a certain generation the curve of the MPS crosses the curve of the best individual of the population, and the fitness function for the MPS is lower during some generations.

7.3 Using fixed structures

Here, we present the EDA analysis when the structures related with the Ising problem (Fig. 9) are employed. Firstly, by looking at Table 2 we can see that the complete structure does not always have a good performance. This fact also happened in Gaussian Ising problem. Again, it is probably due to the direction assigned to the arcs. For this problem, it is interesting to point out

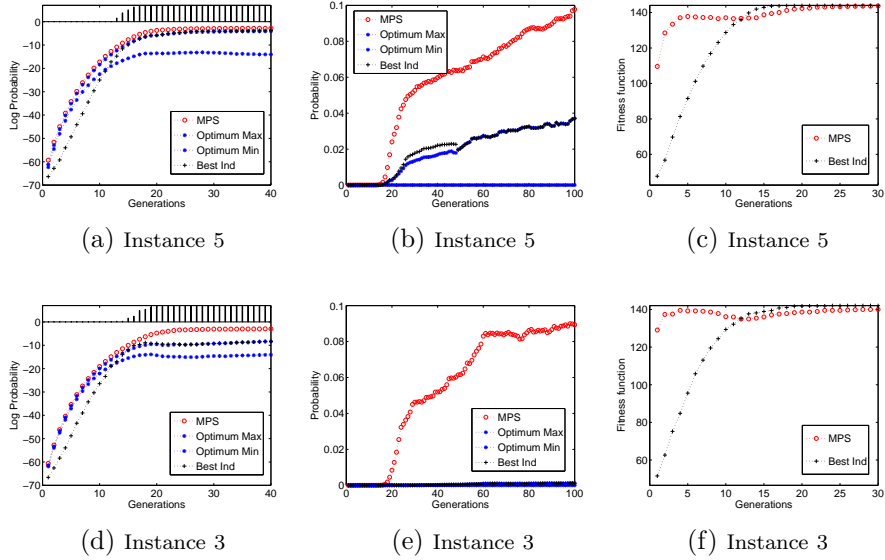


Figure 21: Successful runs when EBNA is applied to $\pm J$ Ising using the complete structure with population size m .

the number of successful executions obtained with the bivariate structure. This simple structure is able to reach the optimum in a significant number of runs for some instances. Clearly, it obtains a higher number of successful executions than for Gaussian Ising.

In order to show the behavior of EBNA with our fixed structures, we show instances 3 and 5 with the population size m given by bisection. In Fig. 21 we report the analysis when the complete structure is introduced in the algorithm. In Fig. 21(a) and 21(b) we can observe that at the beginning of the run the probabilities for the optima and MPS are really close for a high number of generations. This fact is more noticeable when EBNA is executed with the complete structure than with Algorithm B. This indicates that correctly introducing information related with the problem could be beneficial. By looking at Table 2, we can observe that EBNA with the complete structure performs better for the instance 5 and this is reflected in the probability values. In Fig. 21(a) we can see that the maximum probability assigned to the set of optima is very close to the MPS. On the other hand, EBNA does not achieve a good performance (Table 2) when it is applied to instance 3 and the assigned probabilities to the set of optima are more distant from the highest probability (Fig. 21(b)). Fig. 21(b) and 21(e) show that the probability of the MPS is lower than 0.1 at the end of the run. This indicates that the probabilities are shared among different solutions as in the case of using the structural learning in EBNA. The accumulated entropy of the population in Fig. 20 confirms that the populations contain different solutions at the end of the run.

Regarding the function values (Fig. 21(c) and 21(e)), once again the MPS is very close to the optimum value of the function from the beginning of the run when the complete structure is used. This behavior is kept in all analyzed problems when EBNA introduce this type of structure regardless of the performance

of the algorithm in terms of number of successful runs. However, the function values for the best individual of the population have not had a clear improvement. Therefore, the sampling method could be losing valuable information collected by the probabilistic model.

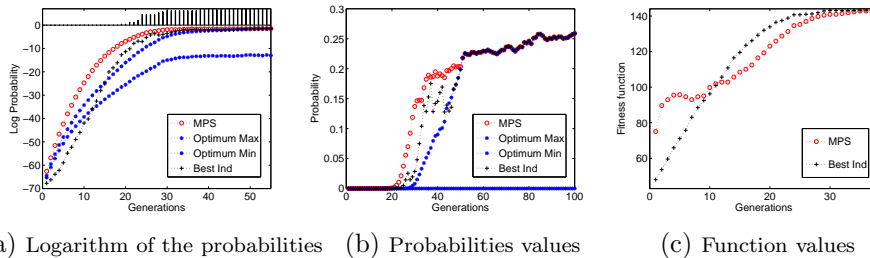


Figure 22: Successful runs when EBNA is applied to instance 5 of $\pm J$ Ising using the bivariate structure with population size m .

In Fig. 22 we show the results obtained when EBNA uses the bivariate structural model in order to solve instance 5. In this case, we can observe a different behavior of the probability values. Firstly, although the probabilities for the optima increase during the search, there is a notable distance between the curves of the maximum and minimum probability. Secondly, this structural model with little information assigns lower probabilities to the optima than a complete structure. Therefore, these probabilities are more distant from the highest in the distribution. However, it is also possible to reach optimal solutions. Lastly, the final probabilities for the MPS are higher than in runs with Algorithm B or the complete structure. Accordingly, Table 2 shows that EBNA with the bivariate structure reaches the lowest number of optima for instance 5. Nevertheless, EBNA with this structure is generally able to reach less optimal solutions and therefore the entropy of the population (Fig. 20) is lower than in the rest of cases.

In Fig. 22(c) we report the function values when EBNA solves this problem with the chain structure. It is very interesting to note that in the first generation the MPS is clearly better than the best individual in the population. Moreover, at the beginning of the run, its difference is even more notable that in the case of EBNA using Algorithm B.

8 EDA behavior solving Max-SAT

8.1 Max-SAT description

The last problem in our analysis is the maximum satisfiability or Max-SAT problem, which has often been used in different works about EDAs [39, 4]. In a simple way, without going into details, given a set of Boolean variables \mathbf{X} and a Boolean expression ϕ , SAT problem asks if there is an assignment \mathbf{x} of the variables such that the expression ϕ is satisfied. In a Boolean expression we can combine the variables using Boolean connectives such as \wedge (logical and), \vee (logical or) and \neg (negation). An expression of the form x_i or $\neg x_i$ is called a literal.

Every Boolean expression can be rewritten into an equivalent expression in a convenient specialized style. In particular, we use the *conjunctive normal form* (CNF) $\phi = \bigwedge_{i=1}^q C_i$. Each of the q C_j s is the disjunction of two or more literals which are called clauses of the expression ϕ . We work with clauses of length $k = 3$. When $k \geq 3$, the SAT problem becomes NP-Complete [9]. An example of a CNF expression with 5 Boolean variables X_1, X_2, X_3, X_4, X_5 and 3 clauses would be, $\phi = (x_1 \vee \neg x_3 \vee x_5) \wedge (\neg x_1 \vee x_3 \vee x_4) \wedge (x_1 \vee \neg x_4 \vee \neg x_2)$.

The Max-SAT problem has the same structure as SAT, but the result, for an assignment \mathbf{x} , is the number of satisfied clauses instead of a truth value. In order to solve Max-SAT, the assignment for \mathbf{X} that maximizes the number of satisfied clauses must be found. Thus, the optimization function can be written as,

$$f_{Max-SAT}(\mathbf{x}) = \sum_{i=1}^q \phi(C_i) \quad (10)$$

where each clause C_i of three literals is evaluated as a Boolean expression that returns 1 if the expression is *true* or 0 if it is *false*. Since C_i is a disjunction, it is satisfied if at least one of its literals is *true*. The variables of \mathbf{X} can overlap arbitrarily in the clauses.

Particularly, we work with 3-CNF SAT problems obtained from the SATLIB [23] repository which provides a large number of SAT instances. All the instances used are satisfiable and our test-set comprises 5 instances with 100 variables and 430 clauses. It is important to note that there could be several assignments for \mathbf{X} that satisfy all clauses and therefore this problem could have different optimal solutions.

8.2 Structures related to the problem

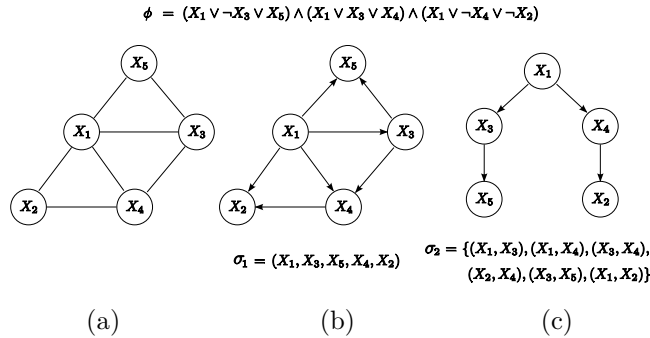


Figure 23: Structures for MAX-SAT given a SAT expression ϕ . (a) Related undirected graph, (b) related Bayesian network where σ is the ancestral order and (c) related tree structure where σ is an order to add edges in the tree.

In this problem, a more elaborate procedure to create the structures was needed. It is mainly because different Max-SAT instances have different interactions among variables and therefore there are different structures for the problem. In this case, we have designed particular methods to create either the complete structure or the bivariate structure.

In order to create the complete structure, we join the variables belonging to same clause C_i with edges. This step is illustrated in the example of Fig. 23(a) where a SAT formula is proposed. Now, in order to create the structure for a Bayesian network, we must direct the edges without creating cycles. However, depending on the ancestral order chosen to direct the edges, the complexity of the Bayesian network can suffer important variations. In fact, we could have an optimization problem in which, given an undirected graph, the objective is to find a directed graph with a minimum number of parents per variable and hence a Bayesian network with a minimum number of parameters. Nevertheless, this is out of the scope of our work. To solve this problem we use a heuristic ancestral order to get a directed acyclic graph. Thus, the variables are ordered from the highest to the lowest number of neighbors in the undirected graph, or in other words, from the highest to lowest the number of overlaps in the clauses of the SAT instance. This type of structure is illustrated in Fig. 23(b) where σ_1 is the defined ancestral order. However, there is a problem with these complete structures generated to solve Max-SAT problems and it is related to the calculation of the MPS given by the Bayesian network. The size of the cliques used to complete this task is too big and therefore the calculation has an extremely high computational cost. To solve this new problem, we were forced to reduce the complexity of the structure by deleting some edges. The criterion is to remove, for each variable, the parents that correspond with interactions that appear less frequently in the clauses. In this way, two is the maximum number of parents that allows a successful calculation of the MPS in all instances. Although in this case we are not able to use all the interactions of the problem in our analysis, this structure will also be called *complete*.

To create the bivariate structure, we have used a tree instead of a chain because we consider it a more appropriate option. To create the tree we have followed a procedure similar to the Chow-Liu algorithm [8]. In Fig. 23(c) we illustrate a possible final result for a particular SAT formula. Firstly, we create an order σ_2 , from highest to lowest, related to the number of times that each couple of variables appear together in the SAT clauses. This is the scoring criterion for the arcs. Starting with an empty structure and following such an order, at each step we add an undirected edge without creating cycles. If there are ties, the selection is random. At the end of the procedure, the root of the tree is the most over-lapped variable in the SAT formula taken from the most frequent couple.

8.3 Using structural learning

In this section we report the obtained results when EBNA solves Max-SAT using Algorithm B to learn a new structure at each step. In this point of the analysis, we have made many comments about our results, and therefore, for this last problem we only present the new relevant information. As in the previous problems, in Table 4 we report the performance of the algorithm in terms of the number of successful executions for each instance and structural model. In this table we can see that when EBNA uses Algorithm B, the population size has little influence on the number of successful runs. Consequently, the analyzed probability values have a similar behavior with both population sizes and hence, only the results for the size given by bisection are reported. Particularly, we show the results for instance 3.

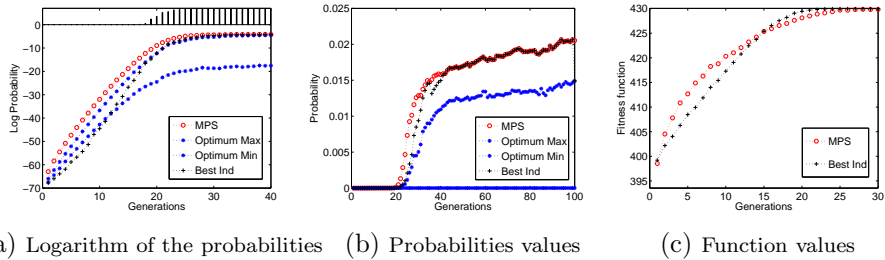


Figure 24: Successful runs when EBNA is applied to instance 3 of Max-SAT using Algorithm B with population size m .

Table 4: Number of EBNA runs that have reached the optimum out of 50 executions of different Max-SAT instances. We report the results for the different structural models and population sizes

Instances Max-SAT	B alg.		Complete		Bivariate	
	m	$m/2$	m	$m/2$	m	$m/2$
I1	47	39	45	41	49	41
I2	46	45	41	33	3	8
I3	44	42	19	16	31	32
I4	47	41	46	41	32	22
I5	48	45	19	26	47	40

In Table 5 we report the number of optima that EBNA has reached on average at the end of the successful runs. In this problem, EBNA is generally able to reach a high number of optima and this fact is also reflected in our analyzed probability values (Fig. 24). Firstly, we can observe that there is a noticeable distance between the maximum and minimum probability assigned to the set of optima during the search. Secondly, the probability for the MPS reaches lower values than in $\pm J$ Ising problem for which EBNA achieves fewer optimal solutions. The accumulated entropy of the population in Fig. 25 confirms that there are different solutions in the population at the end of the run. Despite the probability curves reaching lower values at the end of the run in Max-SAT, we can observe an exponential growth after generation 20. As with the rest of the problems, this fact occurs when the probability of the MPS or the best individual in the population reaches probability values in logarithmic scale around approximately -20 .

For this problem, the fitness function for the MPS shown in Fig 24(c), does not differ greatly from the best individual of the population. The MPS is a little better at the beginning of the run but in some generations the best individual overcomes the MPS and at the end of the run both curves converge.

8.4 Using fixed structures

As we discussed previously, for this particular problem, is not feasible to calculate the MPS if all interactions of the problem are reproduced in the probabilistic

Table 5: On average, number of optimal solutions that EBNA has reached at the end of 50 executions when it is applied to Max-SAT. We report the results for the different structural models and population sizes

Instances Max-SAT	B alg.		Complete		Bivariate	
	m	$m/2$	m	$m/2$	m	$m/2$
I1	273	116	220	85	263	147
I2	1346	704	428	356	293	263
I3	879	338	71	50	136	119
I4	149	86	58	54	42	50
I5	882	401	123	99	165	124

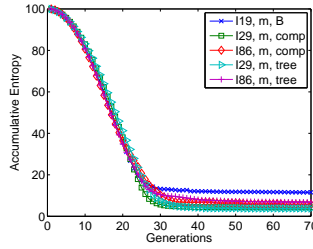


Figure 25: Accumulated entropy of the population when EBNA solves different instances of *Max-SAT* using our three types of structural models

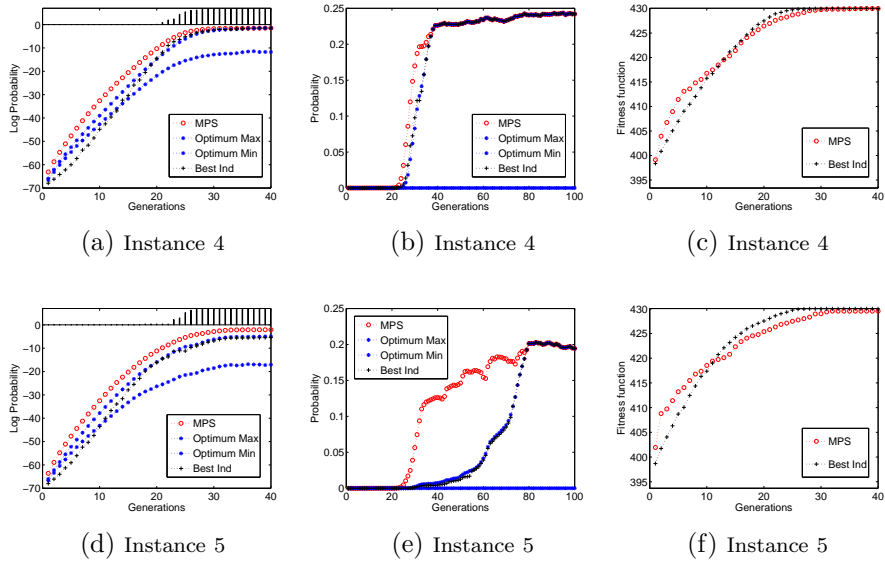


Figure 26: Successful runs when EBNA is applied to Max-SAT using the complete structure with population size m .

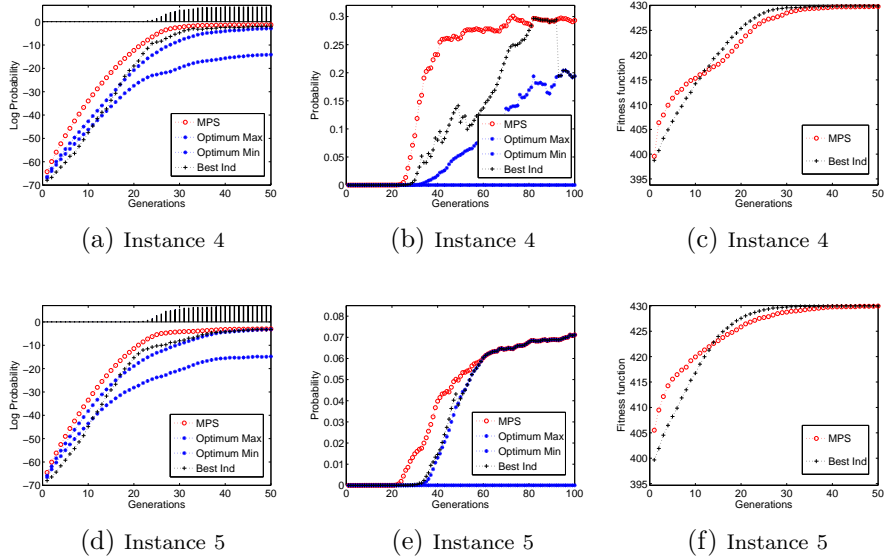


Figure 27: Successful runs when EBNA is applied to Max-SAT using the tree structure with population size m .

model. For this reason, in Table 4 we can see that between our complete structure and the bivariate model there is not much difference. In fact, the tree structure performs better than the complete structure in some instances in terms of number of successful runs. Again, it shows that EDAs with simple structures can achieve a good performance.

In this section we report the results with population size m for instance 4, where EBNA has a better performance with the complete structure, and for instance 5 where the bivariate structure performs better (Table 4).

In Fig. 26 and 27 we report the results for EBNA with the complete structures and with the bivariate structures respectively. For instance 4, when the complete structure is introduced in EBNA (Fig. 26(a) and 26(b)), the probability of the optimum is closer to the highest probability from the beginning and it tends to be the MPS in an earlier generation than when the bivariate structure is used (Fig. 27(a) and 27(b)). However, for instance 5, the optimum becomes the MPS in an earlier generation with the bivariate structure. It is important to note that the probability values reached for the MPS and the best individual at the end of the run are closely related with the number of optimal solutions achieved by EBNA. For instance 4, the complete structure achieves few more successful runs, therefore the final probability values are slightly lower. On the other hand, for instance 5, the bivariate structure has a higher number of successful runs and its final probability values are clearly lower. In general, when fixed structures are used in EBNA, the accumulated entropy (Fig. 25) is lower than with B algorithm. Furthermore, if we look at Table 5, the number of reached optima is also lower with these fixed structures.

Regarding the fitness function values, we show the corresponding results in Fig. 26(c) and 26(f) for the complete structure and in Fig. 27(c) and 27(f) for the bivariate structure. In the rest of the problems when the complete structure

is used, the fitness function for the MPS is close to the optimum from the beginning of the run. In Max-SAT, both structures manage a similar behavior in terms of fitness function values.

9 Related work

In [32], an analysis of the probability assigned by EDAs to the optimum solution is carried out for the Boltzmann EDA (BEDA) and factorized distribution algorithms (FDAs) that use valid and invalid factorizations. The analysis of the probabilities, which was carried out for a toy example, served to illustrate that, under the infinite population assumptions made by BEDA, the use of a valid factorization is a sufficient but not necessary condition for a steady increase, until convergence, of the probability given by BEDA to the optimum. Our work can be seen as an extension of the work presented in [32] in the sense that we investigate the probabilities of EDAs that apply in structural and parametric learning of a more complex class of models and across a range of different problems. We also provide a method to determine exactly the most probable solution given by the model.

Most of the research done in the investigation of the models learned by EDAs that use Bayesian networks [16, 40, 31] has focused on structural descriptors of the networks, in particular on the type (i.e. correct or spurious) and number of the network edges [20, 14, 22, 27, 12, 26]. The analysis of the Bayesian network edges learned by EDAs have allowed to study the effect of the selection and replacement [22, 27] as well as the learning method [14, 12, 26] in the accuracy of the models learned by EDAs and the efficiency of these algorithms. A more recent work [26] considers the likelihood given to the selected set during the model learning step as another source of information about the algorithm's behavior. In this case, not only the structure but also the probabilities are taken into consideration to compute the model descriptor. Nevertheless, none of the previously mentioned papers uses the probabilities given by the models to some distinguished solutions (e.g. most probable explanation, known optimum, etc.) as a means to reveal information about EDAs. In no case there is a reference to the most likely solution that could be sampled from the learned model.

For EDAs that use Markov models [50, 43, 49], different issues related with the relationship between the fitness function and the probabilistic models learned by EDAs (the Markov models) have been investigated. Relevant to the work presented in this paper, is the use of the models learned by the distribution estimation using Markov network algorithm (DEUM) [51, 50, 49] as predictors of the fitness function.

In [5], the product moment correlation coefficient between the Markov model learned by DEUM and the fitness function is used to measure the quality of the model as a fitness function predictor. For a given solution, the prediction is the value given by the Markov model to the solution. The quality of the model is measured using the correlation computed from samples of the search space. Furthermore, the prediction accuracy of Markov models with different structural complexity is investigated for different selection strategies and population sizes.

A substantial difference between the work presented in [5] and the results introduced in this paper is that the analysis of the prediction given by the models is constrained to the solutions taken from the selected population or

random samples. The most probable explanation given by the model is not computed. Another difference is that the evolution of the models throughout the generations is not analyzed. By computing the most probable individual given by the model at each generation we are able to obtain a dynamic view of the quality of the probability model.

10 Conclusions

In this work we have analyzed EDAs from a quantitative point of view in order to better understand their internal behavior. Through the recording of probability and function values for a set of distinguished solutions during the search, we have studied directly the probability distributions generated by this type of algorithms. More specifically, the proposed analysis has allowed us to investigate basic open issues raised in Section 2, whose study entails a deeper understanding and development of EDAs. Now, we are going to look into these questions again, this time providing the new knowledge that we have obtained during the study. Thus, our first and main question was:

- How does the probability assigned to the optimal solution by the probability distributions evolve during the search?

We can distinguish different scenarios depending on the number of optimal solutions of the function to be optimized and the success of the search. Nevertheless, the probability of an optimal solution must always exceed a certain threshold in order to be reached. On one hand, when EBNA is applied to the unimodal problems (Trap5 and Gaussian Ising) and the optimal solution is found, its probability continuously increases until it reaches the value of 1. One exception is function Trap5 and the bivariate structure where the probability of the optimum decreases at the beginning of the run and it increases in the last generations. On the other hand, when EBNA successfully solves the multimodal problems ($\pm J$ Ising and Max-SAT), it is able to reach a subset of the optimal solutions and their probability values also increase during the search. In these problems, the probability is distributed among different solutions at the end of the run (note that the number of generations is limited). Thus, the non-convergence to 1 of the probability values of the MPS or the best individual of the population (both probability curves always rise simultaneously) reflects the multimodality of the function.

In unsuccessful runs, the probability of the optimum always has a similar pattern: at the beginning of the run it increases with the probability of the MPS and the best individual of the population but after a certain generation, before reaching a determined probability threshold, it decreases rapidly. This moment of the run is fully connected with the exponential growth of the probability of the MPS and the best individual which causes the stagnation of the search in a few generations. We believe that it is possible to bring forward this convergence by using probability values of solutions readily available such as the best individual of the population. In our experiments, after this probability approximately reaches the value of -20 in logarithmic scale, the algorithm converges very quickly (both the probability curves and the entropy curves finally stabilize). This probability threshold also could be useful to distinguish between an exploration and exploitation phase. Thus, we could stop the search

at the right time (before the probability of the optimum starts to decrease) and take advantage of the information contained in the probabilistic model by using exploitation techniques.

Finally, the population size also influences the probability assigned to the optimum during the search. Therefore, this is reflected in the number of successful runs. When the population size given by bisection is used, the probability values for the optimum are closer to the highest in the distribution and this is beneficial in order to solve the problem. When EBNA solves $\pm J$ Ising with several optima, the maximum probability assigned to the set of optima is also closer to the highest in the distribution when an adequate population size is used. However, in Max-SAT, the probability curves for the maximum probability are very similar for both population sizes and hence, also the number of successful runs. We have observed that the performance of the algorithm in terms of number of successful runs is closely related to the difference between the probability of the optimum and the MPS during the search.

- Which probabilistic dependences should the model take into account in order to reach the optimum?

As in other works [32, 22], the results shows that it is not strictly necessary to have all the interactions in the model between the variables of the function in order to reach optimal solutions. Nevertheless, the probability curve of an optimum is clearly influenced by the amount of information introduced in the model.

When the complete structure is used in EBNA in order to solve Trap5, we obtain a perfect behavior: the optimum is the most probable solution during the search and it is always quickly reached. However, in general, the use of a complete structure in the terms explained in this work, does not always guarantee successful runs although the probability of the optimum is very close to the highest in the distribution (at least in the first generations). We believe that this fact is due to the selected direction for the arcs which can create different independence relations among variables and hence influence the behavior of the algorithm. It supports the conclusion obtained in [22] about the difficulty of efficiently creating structural models by hand even with complete knowledge of the problem. Nevertheless, it would be interesting to take advantage of the high probability assigned to the optimum when these types of structures are used in order to improve the search. For example, if we know the interactions among the variables of a problem, we could reproduce them in an undirected graph and, at each step of an EBNA, look for the best direction for the edges according to a score. This could be a way to exploit knowledge of the problem.

On the other hand, when structures with little information about the problem are used in EBNA, the probability of the optimum is clearly more distant from the highest in the distribution and the algorithm has a poor performance in general. However, through these types of structures we can also achieve good results such as in the case of Trap5 or Max-SAT. Although in Trap5 we found an unusual behavior, in the Max-SAT problem the maximum probability assigned to the set of optima is close to the highest in the distribution. Furthermore, in this problem, a model with few interactions performs at least as well as a model with more interactions.

- How does the function value for the most probable solution evolve during the search?

The function value for the MPS always increases during the search until it stabilizes in the last generations. At the beginning of the run, it is usually better than the best individual in the population. Furthermore, the MPS increases this difference in function value when the structural model is complete or an adequate population size is used in EBNA. This difference in function value could be used, 1) to improve the setup of EDA parameters in real problems where the optimum is not known, 2) to measure the quality of the information introduced about the problem in the model and 3) to measure the quality of sampling methods.

Particularly, when we reproduce all the interactions between the variables of the problem in the probabilistic model, the function value for the MPS is very close to the optimum from the beginning of the run. Therefore, if we use the structure of the problem, a sampling based on inference of the most probable solution [28] could be very beneficial in EDAs, at least in the first generations. This is another way to exploit the knowledge that we could have about the problem.

11 Future work

There are a number of trends where it is worth extending the results presented in this paper.

Firstly, a direct extension of this work is to reproduce the proposed analysis in problems with different characteristics as for example non-binary discrete problems. Another direct extension is to study the influence of different parameters such as selection or replacement in the descriptors of the probabilistic models introduced in this paper. Similarly, the influence in these descriptors of the model learning algorithm used (e.g. exact vs approximate learning of Bayesian networks [14]) is worth further investigation.

Secondly, it would be interesting to extend this type of quantitative analysis by calculating the k most probable solutions in the probability distributions generated by an EDA during the search. For instance, we can take k as a parameter and evaluate the change in the correlation probability-fitness value as a function of k . Similarly, we can evaluate the way in which the probability values are concentrated around the k most probable configurations. Furthermore, we can compare each model in terms of the average fitness of its k most probable configurations.

Thirdly, some of the ideas collected in the conclusions can be used in the development of adaptive EDAs [45]. For example: 1) by using the probability value of the best individual of the population to anticipate the convergence and take on-line decisions and 2) by using the relation between MPS and best individual in order to self-adjust the population size or to use an adequate sampling method during the search. In the second point, we could study the use of approximate techniques such as loopy belief propagation [37] to calculate the MPS, because in general, its computation is very hard.

Lastly, it would be possible to theoretically model the dependence between probability curves and parameters such as the number of variables, the population

size and the generation. We believe that our experimental results could help to further develop this type of theoretical models.

Acknowledgments

This work has been partially supported by the Saiotek and Research Groups 2007-2012 (IT-242-07) programs (Basque Government), TIN2008-06815-C02-01 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.

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), 2008.
- [2] F. Barahona. On the computational complexity of Ising spin glass model. *Journal of Physics A: Mathematical and General*, 15(10), 1982.
- [3] E. Bengoetxea. *Inexact Graph Matching Using Estimation of Distribution Algorithms*. PhD thesis, Ecole Nationale Supérieure des Télécommunications, 2003.
- [4] S. Brownlee, J. McCall, and D. Brown. Solving the MAXSAT problem using a multivariate EDA based on Markov networks. In *Proceedings of the 2007 conference on Genetic and Evolutionary Computation (GECCO-2007)*, pages 2423–2428, London, England, 2007. ACM.
- [5] S. Brownlee, J. McCall, Q. Zhang, and D. Brown. Approaches to selection and their effect on fitness modelling in an estimation of distribution algorithm. In *Proceedings of the 2008 Congress on Evolutionary Computation (CEC-2008)*, pages 2621–2628, Hong Kong, 2008. IEEE Press.
- [6] W. Buntine. Theory refinement on Bayesian networks. In *Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence*, pages 52–60, 1991.
- [7] E. Castillo, J. M. Gutierrez, and A. S. Hadi. *Expert Systems and Probabilistic Network Models*. Springer-Verlag, 1997.
- [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] S. A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, pages 151–158, Shaker Heights, Ohio, 1971.

- [10] K. Deb and D. E. Goldberg. Sufficient conditions for deceptive and easy binary functions. *Annals of Mathematics and Artificial Intelligence*, 10:385–408, 1994.
- [11] R. Dechter. Bucket elimination: A unifying framework for reasoning. *Artificial Intelligence*, 113(1-2):41–85, 1999.
- [12] C. Echegoyen, J. A. Lozano, R. Santana, and P. Larrañaga. Exact Bayesian network learning in estimation of distribution algorithms. In *Proceedings of the 2007 Congress on Evolutionary Computation (CEC-2007)*, pages 1051–1058. IEEE Press, 2007.
- [13] C. Echegoyen, A. Mendiburu, R. Santana, and J. Lozano. Analyzing the probability of the optimum in EDAs based on Bayesian networks. In *Proceedings of the 2009 Congress on Evolutionary Computation (CEC-2009)*, pages 1652–1659, Trondheim, Norway, 2009. IEEE Press.
- [14] 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.
- [15] E. A. Eiben and J. E. Smith. *Introduction to Evolutionary Computing (Natural Computing Series)*. Springer, 2003.
- [16] 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 (CIMA-99)*, pages 151–173, Havana, Cuba, 1999.
- [17] 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.
- [18] J. Gámez. *Advances in Bayesian Networks*, chapter Abductive inference in Bayesian networks: A review, pages 101–120. Springer, 2004.
- [19] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, Reading, MA, 1989.
- [20] M. Hauschild and M. Pelikan. Enhancing efficiency of hierarchical BOA via distance-based model restrictions. MEDAL Report No. 2008007, Missouri Estimation of Distribution Algorithms Laboratory (MEDAL), April 2008.
- [21] M. Hauschild, M. Pelikan, K. Sastry, and D. E. Goldberg. Using previous models to bias structural learning in the hierarchical BOA. MEDAL Report No. 2008003, Missouri Estimation of Distribution Algorithms Laboratory (MEDAL), 2008.
- [22] M. Hauschild, M. Pelikan, K. Sastry, and C. Lima. Analyzing Probabilistic Models in Hierarchical BOA. *IEEE Transactions on Evolutionary Computation*. To appear.

- [23] H. Hoos and T. Stutzle. SATLIB: An online resource for research on SAT. In H. van Maaren I. P. Gent and T. Walsh, editors, *SAT2000*, pages 283–292. IOS Press, 2000.
- [24] E. Ising. The theory of ferromagnetism. *Zeitschrift fuer Physik*, 31:253–258, 1925.
- [25] P. Larrañaga and J. A. Lozano, editors. *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*. Kluwer Academic Publishers, Boston/Dordrecht/London, 2002.
- [26] C. F. Lima, F. G. Lobo, and M. Pelikan. From mating pool distributions to model overfitting. In *Proceedings of the ACM Genetic and Evolutionary Computation Conference (GECCO-2008)*, pages 431–438. IEEE Press, 2008.
- [27] C. F. Lima, M. Pelikan, D. E. Goldberg, F. G. Lobo, K. Sastry, and M. Hauschild. Influence of selection and replacement strategies on linkage learning in BOA. In *Proceedings of the 2007 Congress on Evolutionary Computation (CEC-2007)*, pages 1083–1090. IEEE Press, 2007.
- [28] A. Mendiburu, R. Santana, and J. A. Lozano. Introducing belief propagation in estimation of distribution algorithms: A parallel framework. Technical Report EHU-KAT-IK-11/07, Department of Computer Science and Artificial Intelligence, University of the Basque Country, October 2007.
- [29] H. Mühlenbein and R. Höns. The factorized distributions and the minimum relative entropy principle. In M. Pelikan, K. Sastry, and E. Cantú-Paz, editors, *Scalable Optimization via Probabilistic Modeling: From Algorithms to Applications*, Studies in Computational Intelligence, pages 11–38. Springer-Verlag, 2006.
- [30] H. Mühlenbein and T. Mahnig. FDA – a scalable evolutionary algorithm for the optimization of additively decomposed functions. *Evolutionary Computation*, 7(4):353–376, 1999.
- [31] H. Mühlenbein and T. Mahnig. Evolutionary synthesis of Bayesian networks for optimization. In M. Patel, V. Honavar, and K. Balakrishnan, editors, *Advances in Evolutionary Synthesis of Intelligent Agents*, pages 429–455. MIT Press, Cambridge, Mass., 2001.
- [32] H. Mühlenbein, T. Mahnig, and A. Ochoa. Schemata, distributions and graphical models in evolutionary optimization. *Journal of Heuristics*, 5(2):213–247, 1999.
- [33] 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.
- [34] K. Murphy. The Bayes Net Toolbox for Matlab. *Computer science and Statistics: Proceedings of Interface*, 33, 2001.

- [35] J. Ocenasek. Entropy-based convergence measurement in discrete estimation of distribution algorithms. In J. A. Lozano, P. Larrañaga, I. Inza, and E. Bengoetxea, editors, *Towards a New Evolutionary Computation: Advances on Estimation of Distribution Algorithms*, pages 39–50. Springer-Verlag, 2006.
- [36] A. Ochoa and M. R. Soto. Linking entropy to estimation of distribution algorithms. In J. A. Lozano, P. Larrañaga, I. Inza, and E. Bengoetxea, editors, *Towards a New Evolutionary Computation: Advances on Estimation of Distribution Algorithms*, pages 1–38. Springer-Verlag, 2006.
- [37] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo, California, 1988.
- [38] M. Pelikan. *Hierarchical Bayesian Optimization Algorithm. Toward a New Generation of Evolutionary Algorithms*. Studies in Fuzziness and Soft Computing. Springer, 2005.
- [39] M. Pelikan and D. E. Goldberg. Hierarchical BOA solves Ising Spin Glasses and MAXSAT. In *In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2003)*, pages 1271–1282, Chicago, Illinois USA, 2003. Springer.
- [40] 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.
- [41] M. Pelikan and A. K. Hartmann. Searching for ground states of Ising spin glasses with hierarchical BOA and cluster exact approximation. In M. Pelikan, K. Sastry, and E. Cantú-Paz, editors, *Scalable Optimization via Probabilistic Modeling: From Algorithms to Applications*, Studies in Computational Intelligence, pages 333–349. Springer-Verlag, 2006.
- [42] M. Pelikan, K. Sastry, and D. E. Goldberg. Sporadic model building for efficiency enhancement of the hierarchical BOA. *Genetic Programming and Evolvable Machines*, 9(1):53–84, 2008.
- [43] R. Santana. A Markov network based factorized distribution algorithm for optimization. In *Proceedings of the 14th European Conference on Machine Learning (ECML-PKDD 2003)*, volume 2837 of *Lecture Notes in Artificial Intelligence*, pages 337–348, Dubrovnik, Croatia, 2003. Springer Verlag.
- [44] R. Santana, C. Echegoyen, A. Mendiburu, C. Bielza, J. A. Lozano, P. Larrañaga, R. Armañanzas, and S. Shakya. MATEDA: A suite of EDA programs in matlab. Technical Report EHU-KZAA-IK-2/09, Department of Computer Science and Artificial Intelligence, 2009.
- [45] R. Santana, P. Larrañaga, and J. A. Lozano. Adaptive estimation of distribution algorithms. In C. Cotta, M. Sevaux, and K. Sörensen, editors, *Adaptive and Multilevel Metaheuristics*, Studies in Computational Intelligence, pages 177–197. Springer-Verlag, 2007.

- [46] 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.
- [47] R. Santana, P. Larrañaga, and J. A. Lozano. Research topics on discrete estimation of distribution algorithms. *Memetic Computing*, 1(1):35–54, 2009.
- [48] G. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 7(2):461–464, 1978.
- [49] S. Shakya. *DEUM: A framework for an Estimation of Distribution Algorithm based on Markov Random Fields*. PhD thesis, The Robert Gordon University. School of Computing, Aberdeen, UK, 2006.
- [50] S. Shakya and J. McCall. Optimization by estimation of distribution with DEUM framework based on Markov random fields. *International Journal of Automation and Computing*, 4(3):262–272, 2007.
- [51] S. Shakya, J. McCall, and D. Brown. Using a Markov network model in a univariate EDA: An empirical cost-benefit analysis. In H.-G. Beyer and U.-M. O’Reilly, editors, *Proceedings of Genetic and Evolutionary Computation Conference (GECCO-2005)*, pages 727–734, Washington, D.C., USA, 2005. ACM Press.
- [52] S. E. Shimony. Finding MAPs for belief networks is NP-hard. *Artificial Intelligence*, 68(2):399–410, 1994.
- [53] P. A. Simionescu, D. Beale, and G. V. Dozier. Teeth-number synthesis of a multispeed planetary transmission using an estimation of distribution algorithm. *Journal of Mechanical Design*, 128(1):108–115, 2007.
- [54] B. Yuan, M. E. Orłowska, and S. W. Sadiq. Finding the optimal path in 3D spaces using EDAs - the wireless sensor networks scenario. In *Proceedings of the Adaptive and Natural Computing Algorithms, 8th International Conference, ICANNGA 2007, Warsaw, Poland, April 11-14, 2007*, pages 536–545. Springer Verlag, 2007.
- [55] Q. Zhang and H. Mühlenbein. On the convergence of a class of estimation of distribution algorithms. *IEEE Transactions on Evolutionary Computation*, 8(2):127–136, 2004.