Globally Multimodal Problem Optimization Via an Estimation of Distribution Algorithm Based on Unsupervised Learning of Bayesian Networks

J. M. Peña [a], J. A. Lozano [b], and P. Larrañaga [b]

[a] *Computational Biology, Dept. of Physics and Measurement Technology*
Linköping University, Sweden
jmp@ifm.liu.se

[b] *Intelligent Systems Group, Dept. of Computer Science and Artificial Intelligence*
University of the Basque Country, Spain
{*ccploalj, ccplamup*}*@si.ehu.es*

Abstract

Many optimization problems are what can be called globally multimodal, i.e. they present several global optima. Unfortunately, this is a major source of difficulties for most estimation of distribution algorithms that makes their effectiveness and efficiency degrade, due to genetic drift. With the aim of overcoming these drawbacks for discrete globally multimodal problem optimization, this paper introduces and evaluates a new estimation of distribution algorithm based on unsupervised learning of Bayesian networks. We report satisfactory experiments with symmetrical binary optimization problems.

Key words: Estimation of distribution algorithms, Bayesian networks, unsupervised learning

1 Introduction

Estimation of distribution algorithms (EDAs) [41,53,57,60] are some relatively novel evolutionary algorithms (EAs) that are receiving increasing attention in the literature. Like any other class of EAs, EDAs solve a given optimization problem by evolving a population of individuals, i.e. a set of solutions to the optimization problem, towards promising zones of the search space. Such an evolution is mainly based on iterating between two steps: Selection of fit individuals from the current population, and combination of the selected individuals in order to create an offspring population and replace (partially) the current one. Unlike most EAs, EDAs do not make use of variation operators (e.g. crossover and/or mutation) in the combination step. Instead, EDAs generate the offspring population at each iteration by learning and subsequent simulation of a joint probability distribution for the individuals selected.

How the joint probability distribution is estimated from the individuals selected at each iteration as well as what assumptions are made for this process to be tractable is what distinguishes one EDA from another. However, existing EDAs typically disregard that many optimization problems are what can be called *globally multimodal*, i.e. they present multiple global optima, and that often it is necessary or desirable to discover as many global optima as possible instead of only one of them. In this paper, we propose and evaluate a new EDA tailored to this scenario: The unsupervised estimation of Bayesian network algorithm (UEBNA). The only peculiarity of the UEBNA is the use of a Bayesian network for data clustering [63–66,68] in order to factorize the joint probability distribution for the individuals selected at each iteration. This allows modelling simultaneously the basins of the different global optima represented by the selected individuals. Therefore, we conjecture that the UEBNA should be able to discover more global optima per run than existing EDAs while speeding up convergence in terms of number of evaluations of the objective function, because individuals belonging to different basins are not used together which usually results in poorly fit individuals and, thus, delays convergence.

The remainder of this paper is structured as follows. Section 2 reviews EDAs. Section 3 describes unsupervised learning of Bayesian networks and shows how this is incorporated into the EDA framework, resulting in the UEBNA. Section 4 evaluates the UEBNA on symmetrical binary optimization problems. Finally, Section 5 closes with some discussion and conclusions.

2 Estimation of Distribution Algorithms

Among stochastic heuristic search strategies for problem optimization, *evolutionary algorithms (EAs)* are well known for their good performance and wide applicability. Some classical EAs are genetic algorithms [25,34], evolutionary programming [18,19] and evolution strategies [70,74]. More recently, a novel class of EAs, known as *estimation of distribution algorithms (EDAs)* [41,53,57,60], has been proposed and evaluated successfully in a wide variety of scenarios. This section first reviews EDAs and, then, discusses the difficulties that they encounter when optimizing globally multimodal problems. Prior to this, we introduce some terms used throughout the text.

The main feature shared by all the instances of the EA paradigm is being inspired by natural evolution of species. That is why much of the nomenclature of EAs is borrowed from the field of natural evolution. For instance, we talk about *populations* to refer to sets of solutions to an optimization problem, each solution is called an *individual*, and each basic component of an individual is called a *gene*. The main components of most EAs are: An initial population of individuals, a *selection method* over individuals, a set of *variation operators* over individuals, and a *replacement method* over individuals. Basically, all the EAs work in the same iterative way: At each iteration or *generation* some individuals of the current population are selected according to the selection method and modified by the variation operators in order to create new individuals and, consequently, a new population according to the replacement method. The objective of this iterative process is to evolve the population towards promising zones of the search space of the problem at hand.

2.1 Generic Estimation of Distribution Algorithm

The most distinctive characteristic of EDAs with respect to the rest of EAs is that EDAs replace the application of variation operators in order to generate the next population from the current one at each iteration by learning and subsequent simulation of a joint probability distribution for those individuals selected from the current population by means of the selection method. This results in two important advantages of EDAs over classical EAs: The sometimes necessary design of variation operators tailored to the particular optimization problem at hand is avoided, and the number of parameters to be assessed by the user is reduced. A further advantage of EDAs over classical EAs is that the relationships between the random variables that represent the genes of every individual selected can be explicitly expressed through the joint probability distribution learnt from them, instead of being implicitly kept by the individuals of successive populations as building blocks. In fact, it was already recognized in [25,34] that detecting interacting genes would be beneficial to genetic algorithms. This source of knowledge was called linkage information. This idea has been exploited by many researchers for the last few years in order to enhance the performance of genetic algorithms [25,26,46]. Finally, EDAs outperform classical EAs in deceptive optimization problems [16,41,52]. The generic EDA iterates between three main steps, after the individuals of the initial population $\boldsymbol{po}_1$ have been generated, usually uniformly, and evaluated. The iterative process ends when the stopping criterion is met, e.g. performance of a maximum number of generations, uniformity in the current population, or no improvement with regard to the best individual of the previous generation. This causes the best solutions found so far being returned. The three main steps of the u-th iteration of the generic EDA are as follows for all u. First, N of the Q individuals of the current population $\boldsymbol{po}_u$ are selected by means of the selection method. Then, these selected individuals are used to construct a learning database $\boldsymbol{d}_u$ from which a joint probability distribution

Let $\boldsymbol{po}_1$ be a population composed of Q uniformly generated individuals
Evaluate the individuals in $\boldsymbol{po}_1$
$u = 1$
while the stopping condition is not met **do**
  Let $\boldsymbol{d}_u$ group N individuals selected from $\boldsymbol{po}_u$ via the selection method
  Let $p_u(\boldsymbol{x})$ be the joint probability distribution for $\boldsymbol{X}$ learnt from $\boldsymbol{d}_u$
  Let $\boldsymbol{of}_u$ be the offspring population composed of M individuals sampled from $p_u(\boldsymbol{x})$
  Evaluate the individuals in $\boldsymbol{of}_u$
  Let $\boldsymbol{po}_{u+1}$ be the population created from $\boldsymbol{po}_u$ and $\boldsymbol{of}_u$ via the replacement method
  $u++$
Return the best individuals found so far

Fig. 1. Pseudocode of the generic EDA.

for $\boldsymbol{X}$, $p_u(\boldsymbol{x})$, is induced. $\boldsymbol{X} = (X_1, \ldots, X_n)$ denotes an n-dimensional discrete random variable, where X_i is associated with the i-th gene of every individual in $\boldsymbol{d}_u$. Finally, M individuals are sampled from $p_u(\boldsymbol{x})$ and evaluated in order to create the offspring population $\boldsymbol{of}_u$ which, then, is used to generate the new population $\boldsymbol{po}_{u+1}$ by replacing some individuals of $\boldsymbol{po}_u$ via the replacement method. See Fig. 1 for a schematic of the generic EDA.

2.2 Families of Estimation of Distribution Algorithms

We have discussed above that replacing the application of variation operators by learning and simulation of $p_u(\boldsymbol{x})$ has immediate benefits. However, it carries some cost too, because learning $p_u(\boldsymbol{x})$ from $\boldsymbol{d}_u$ is not a trivial task. As the computation of all the parameters needed to completely specify $p_u(\boldsymbol{x})$ in the extensive representation is often prohibitive, several families of EDAs have arisen where this joint probability distribution is assumed to factorize according to a certain class of probabilistic models. The remainder of this section briefly reviews some of these families, according to an increasing order of complexity. The interested reader may consult [41] for a more thorough review.

The simplest family of EDAs is based on the assumption that $p_u(\boldsymbol{x})$ factorizes as a product of n univariate and mutually independent probability distributions, one for each X_i. Obviously, this is very far from what happens in difficult optimization problems, where relationships between the unidimensional random variables in $\boldsymbol{X}$ usually exist. However, this assumption simplifies learning the probabilistic model for the factorization of $p_u(\boldsymbol{x})$ from $\boldsymbol{d}_u$, as this process reduces to parameter fitting. Some examples of this approach are [2,40,51,71].

A slightly more complex family of EDAs consists of those algorithms that take into account only bivariate dependencies between the unidimensional random variables in $\boldsymbol{X}$ for the factorization of $p_u(\boldsymbol{x})$. Therefore, it is enough to use second order statistics to learn the probabilistic model for such a factorization. Some members of this family of EDAs are [3,4,12,62].

With the aim of improving performance, there have been proposed several instances of the generic EDA that involve statistics of order greater than two in the factorization of $p_u(\boldsymbol{x})$. See, for instance, [29,52,75]. However, the most relevant research within this approach is based on Bayesian networks [7,11,37,44,56], so that learning $p_u(\boldsymbol{x})$ from $\boldsymbol{d}_u$ reduces to learning a Bayesian network for $\boldsymbol{X}$ from $\boldsymbol{d}_u$. As a result, the factorization of $p_u(\boldsymbol{x})$ corresponds to the graphical factorization represented by the induced Bayesian network for $\boldsymbol{X}$. For a thorough discussion of these EDAs, the reader is referred to, for instance, [16,41,42,57,59].

2.3 Globally Multimodal Problem Optimization

Many optimization problems are globally multimodal and, often, it is necessary or desirable to identify as many global optima as possible. In this scenario, classical EAs are ineffective, as they converge to at best a single global optima. The explanation is straightforward. When optimizing a globally multimodal problem, the basins of different global optima may be represented in the population. As there is not significant selective preference for one of the basins in the population over another, the stochastic variations due to the selection method make the population drift towards one of them and, thus, discover only one global optimum at most. Moreover, this global optimum is randomly chosen from the existing global optima. This phenomenon is known as *genetic drift* [13,25,28]: In the absence of selective pressure, the stochastic nature of the selection method reduces population diversity.

Globally multimodal optimization problems are challenging for classical EAs not only in terms of effectiveness but also in terms of efficiency: The existence of several global peaks makes convergence speed slow down until the population drifts to one of the global peaks [58]. Basically, the difficulties appear because combining good solutions coming from different parts of the search space or basins often results in poor solutions. The only mechanism that classical EAs have to make the population drift towards a single basin is genetic drift, however this phenomenon normally occurs very slowly. Therefore, the interest in obtaining several global optima of globally multimodal optimization problems by preventing genetic drift as much as possible is not only quantitative but also qualitative. Much attention has been devoted to the study of genetic drift as a cause of suboptimal convergence in classical EAs regarding, mainly, convergence time [28], niching [35] and population sizing [47]. However, few works exist where some classical EAs have been modified in order to alleviate genetic drift as much as possible for globally multimodal problem optimization [33].

EDAs should encounter exactly the same difficulties for globally multimodal problem optimization as classical EAs do, unless the class of probabilistic models that factorize $p_u(\boldsymbol{x})$ is flexible enough to model simultaneously the different basins that may be represented in $\boldsymbol{d}_u$. One way to guarantee this is by using probabilistic models that are able to encode conditional dependencies between the unidimensional random variables in $\boldsymbol{X}$, i.e. conditional dependencies between the random variables corresponding to genes. Alternatively, those EDAs that do not model conditional dependencies between the unidimensional random variables in $\boldsymbol{X}$ can perform well in globally multimodal problem optimization by incorporating *niching* [25,27], i.e. the population is distributed in niches or subpopulations, in order to avoid combining solutions coming from different basins. Both approaches are suggested in [58], although the authors evaluate only the latter for symmetrical (globally multimodal) problem optimization. Basically, this paper implements niching based on partitional data clustering via the K-means algorithm [1,30] within one of the simplest EDAs, namely the univariate marginal distribution algorithm [41,51]. In [23], the population-based incremental learning algorithm is extended to continuous problem optimization by learning and sampling a finite mixture model. This EDA can be seen as another example of niching, since each component in the mixture can represent a different niche. Unfortunately, the authors do not provide much evidence on the benefits of their algorithm for globally multimodal problem optimization.

3 An Estimation of Distribution Algorithm Based on Unsupervised Learning of Bayesian Networks

The previous section has outlined two approaches to alleviate the poor performance of most EDAs for globally multimodal problem optimization: Either using probabilistic models that are able to encode conditional dependencies, or incorporating niching (e.g. based on partitional data clustering). Although these two approaches are apparently unrelated, they can be easily combined if data clustering is faced from a model-based perspective and the class of models considered can encode conditional dependencies. Such a combined approach may benefit from the strengths of both original approaches and increase robustness and reliability on the problem optimization process. A sensible implementation of this consists in using unsupervised learning of Bayesian networks [63–66,68]. This section first introduces unsupervised learning of Bayesian networks and, then, shows how this can be incorporated into the EDA framework for effective and efficient globally multimodal problem optimization.

3.1 Unsupervised Learning of Bayesian Networks

Data clustering is one of the main problems that arises in a great variety of fields [1,15,30,48,63]. Given some data $\boldsymbol{d}$ in the form of a set of instances with an underlying group-structure, data clustering may be roughly defined as the search for the best description of this group-structure, when the group mem-

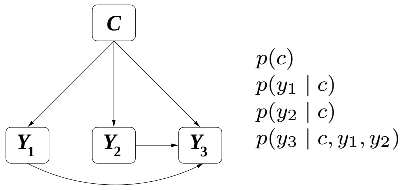

Fig. 2. Structure and conditional probability distributions of a BN for data clustering for $\boldsymbol{X} = (C, \boldsymbol{Y}) = (C, Y_1, Y_2, Y_3)$.

bership of every instance is unobserved. Each of the groups in $\boldsymbol{d}$ is called a *cluster*. The lack of knowledge of the cluster membership of every instance in $\boldsymbol{d}$ makes data clustering be also referred to as *unsupervised learning*.

Most solutions to data clustering problems can be classified as being either *partitional*, *hierarchical* or *model-based*. Partitional and hierarchical approaches describe the group-structure underlying $\boldsymbol{d}$ as a partition of $\boldsymbol{d}$ or as a sequence of tree-like nested partitions of $\boldsymbol{d}$, respectively. On the other hand, model-based approaches describe the group-structure underlying $\boldsymbol{d}$ through a probabilistic model induced from $\boldsymbol{d}$. In this paper, we take a model-based approach to data clustering. In particular, we assume that $\boldsymbol{d}$ contains N instances or cases, i.e. $\boldsymbol{d} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$. The l-th case of $\boldsymbol{d}$ is represented by an $(n+1)$-dimensional discrete vector $\boldsymbol{x}_l = (x_{l1}, \ldots, x_{ln+1})$ partitioned as $\boldsymbol{x}_l = (c_l, \boldsymbol{y}_l)$ for all l: c_l is the unobserved cluster membership, and $\boldsymbol{y}_l = (y_{l1}, \ldots, y_{ln})$ is the vector of observations or *predictive attributes*. We assume as well that the number of clusters underlying $\boldsymbol{d}$, denoted by K, is known. From a model-based perspective, every case in $\boldsymbol{d}$ can be seen as a partial instance of an $(n+1)$-dimensional discrete random variable $\boldsymbol{X} = (X_1, \ldots, X_{n+1})$ partitioned as $\boldsymbol{X} = (C, \boldsymbol{Y})$: C is a unidimensional discrete random variable representing the unobserved cluster membership, i.e. the *cluster random variable*, and $\boldsymbol{Y} = (Y_1, \ldots, Y_n)$ is an n-dimensional discrete random variable representing the set of predictive attributes, i.e. the *predictive random variable*. Therefore, model-based data clustering can be solved by learning a joint probability distribution for $\boldsymbol{X}$ from $\boldsymbol{d}$. One of the paradigms specially well suited for such a purpose are Bayesian networks [7,11,37,44,56].

Let $\boldsymbol{X} = (C, \boldsymbol{Y})$ be a random variable as stated above. A *Bayesian network (BN)* for data clustering for $\boldsymbol{X}$ is a pair $(\boldsymbol{s}, \boldsymbol{\theta})$, where $\boldsymbol{s}$ is the *model structure* and $\boldsymbol{\theta}$ are the *model parameters* [63–66,68]. The model structure $\boldsymbol{s}$ is an acyclic directed graph whose nodes correspond to the unidimensional random variables in $\boldsymbol{X}$. Throughout the text, the terms node and random variable are used interchangeably. The model parameters $\boldsymbol{\theta}$ specify a conditional probability distribution for each node X_i in $\boldsymbol{s}$ given its parents $\boldsymbol{Pa}_i$ in $\boldsymbol{s}$, $p(x_i \mid \boldsymbol{pa}_i)$. These conditional probability distributions are all typically multinomial.

A BN for data clustering $(\boldsymbol{s}, \boldsymbol{\theta})$ for $\boldsymbol{X}$ represents a joint probability distribu-

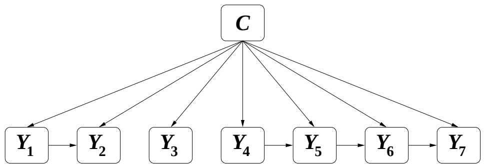

Fig. 3. Structure of a TANB model for data clustering.

tion for $\boldsymbol{X}$, $p(\boldsymbol{x})$, through the following graphical factorization:

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

Therefore, $\boldsymbol{s}$ encodes a set of conditional (in)dependencies between the random variables in $\boldsymbol{X}$. Moreover, $\boldsymbol{s}$ is usually constrained so that every Y_i is a child of C, i.e. $C \in \boldsymbol{Pa}_i$ for all $i > 1$. This restriction is imposed by the assumption that C has an impact on the joint probability distribution for $\boldsymbol{Y}$. See Fig. 2 for an example of a BN for data clustering.

In this paper, we interpret unsupervised learning of BNs as an optimization problem. This is a challenging optimization problem in general. As a matter of fact, it has been proven in [8] that the identification of the BN structure with the highest Bayesian Dirichlet equivalent score [31] among all the BN structures in which every node has no more than t parents is an NP-hard optimization problem for $t > 1$. It is usually assumed that this hardness holds for other common scores as well, though there is not yet a formal proof [9]. These results also apply to unsupervised learning of BNs.

As search space, we consider the space of structures of BNs for data clustering. This space can be restricted to the space of DAGs for $\boldsymbol{Y}$, due to the fact that every Y_i is a child of C. Alternative search spaces include the space of equivalence classes of structures of BNs for data clustering [9,55], and the space of ancestral orderings of structures of BNs for data clustering [22,43]. Note that, as usually, model parameter fitting is considered a secondary optimization problem: Given a BN structure for data clustering, maximum likelihood (ML) or maximum a posteriori model parameter estimates can be effectively obtained via approximation techniques such as gradient descent methods [6], Gibbs sampling [24] or the EM algorithm [14,49]. In some cases, it may be desirable to restrict the attention to those BNs for data clustering that trade off expressivity for simplicity. This reduces the search space while the models considered are still expressive enough. One such example is the class of *tree augmented naive Bayes (TANB) models for data clustering* [50,63,64,66]. TANB models for data clustering include those BNs for data clustering such that every Y_i has at most one other unidimensional predictive random variable in $\boldsymbol{Y}$ as a parent. See Fig. 3 for an example. TANB models have been also used in data classification [21,39].

As search strategy, we consider the *Bayesian structural EM (BSEM) algorithm*

1. Let $\boldsymbol{s}_1$ be the initial model structure
2. **for** $u = 1, 2, \ldots$ **do**
3. Run the EM algorithm in order to approximate the ML parameters $\widehat{\boldsymbol{\theta}}_{\boldsymbol{s}_u}$ for $\boldsymbol{s}_u$
4. Perform a greedy hill-climbing search over model structures, evaluating each one by $Sc(\boldsymbol{s} : \boldsymbol{s}_u, \boldsymbol{d}) = E[\log L(\boldsymbol{d} \mid \boldsymbol{s}) \mid \boldsymbol{d}^{\boldsymbol{Y}}, \widehat{\boldsymbol{\theta}}_{\boldsymbol{s}_u}, \boldsymbol{s}_u] = \sum_{\boldsymbol{d}^C} \log L(\boldsymbol{d}^C, \boldsymbol{d}^{\boldsymbol{Y}} \mid \boldsymbol{s}) L(\boldsymbol{d}^C \mid \boldsymbol{d}^{\boldsymbol{Y}}, \widehat{\boldsymbol{\theta}}_{\boldsymbol{s}_u}, \boldsymbol{s}_u)$
5. Let $\boldsymbol{s}_{u+1}$ be the model structure with the highest score among those visited in step 4
6. **if** $Sc(\boldsymbol{s}_{u+1} : \boldsymbol{s}_u, \boldsymbol{d}) = Sc(\boldsymbol{s}_u : \boldsymbol{s}_u, \boldsymbol{d})$ **then**
7. Return $(\boldsymbol{s}_u, \widehat{\boldsymbol{\theta}}_{\boldsymbol{s}_u})$

Fig. 4. Pseudocode of the BSEM algorithm for unsupervised learning of BNs.

[20]. The BSEM algorithm relies on following the basic intuition of the iterations of the EM algorithm: To take advantage of the best estimate of the joint probability distribution found so far in order to compute quantities of interest that can not be directly obtained from the data at hand, and then to use effective and efficient model learning algorithms for complete data. Fig. 4 shows a pseudocode of the BSEM algorithm for unsupervised learning of BNs. The BSEM algorithm iterates between two main steps that are as follows for the u-th iteration for all u. The first step (step 3 in Fig. 4) approximates the ML parameters $\widehat{\boldsymbol{\theta}}_{\boldsymbol{s}_u}$ for the current model structure $\boldsymbol{s}_u$ given the observed data $\boldsymbol{d}^{\boldsymbol{Y}}$, usually via the EM algorithm. On the other hand, the second step (step 4 in Fig. 4) searches for the highest scoring model structure in order to replace the current one. This latter step is usually solved through a greedy hill-climbing search considering all the possible additions, removals and reversals of a single arc at each point in the search. The score that guides the search is usually the expected $\log L(\boldsymbol{d} \mid \boldsymbol{s})$, where $\boldsymbol{s}$ is the model structure being evaluated and the expectation is taken with respect to $\boldsymbol{d}^{\boldsymbol{Y}}$, $\boldsymbol{s}_u$ and $\widehat{\boldsymbol{\theta}}_{\boldsymbol{s}_u}$:

$$\begin{aligned} Sc(\boldsymbol{s} : \boldsymbol{s}_u, \boldsymbol{d}) &= E[\log L(\boldsymbol{d} \mid \boldsymbol{s}) \mid \boldsymbol{d}^{\boldsymbol{Y}}, \widehat{\boldsymbol{\theta}}_{\boldsymbol{s}_u}, \boldsymbol{s}_u] \\ &= \textstyle\sum_{\boldsymbol{d}^C} \log L(\boldsymbol{d}^C, \boldsymbol{d}^{\boldsymbol{Y}} \mid \boldsymbol{s}) L(\boldsymbol{d}^C \mid \boldsymbol{d}^{\boldsymbol{Y}}, \widehat{\boldsymbol{\theta}}_{\boldsymbol{s}_u}, \boldsymbol{s}_u) \end{aligned} \tag{2}$$

where $\boldsymbol{d}^C$ denotes a labelling or completion of $\boldsymbol{d}$. Note that this score requires going through every possible completion $\boldsymbol{d}^C$ of $\boldsymbol{d}$, which may be prohibitive. Instead, we approximate Eq. (2) by considering only the completion $\boldsymbol{d}^C$ that scores the highest $L(\boldsymbol{d}^C \mid \boldsymbol{d}^{\boldsymbol{Y}}, \widehat{\boldsymbol{\theta}}_{\boldsymbol{s}_u}, \boldsymbol{s}_u)$. Therefore, the score for the structural search step of each iteration of the BSEM algorithm can be computed in factorable and closed form as reported in [10,31].

3.2 Unsupervised Estimation of Bayesian Network Algorithm

This section describes the *unsupervised estimation of Bayesian network algorithm (UEBNA)*, whose only peculiarity with respect to existing EDAs is being based on unsupervised learning of BNs. As discussed previously, incorporating unsupervised learning of BNs into the EDA framework seems a natural solu-

Let $\boldsymbol{po}_1$ be a population composed of Q uniformly generated individuals
Evaluate the individuals in $\boldsymbol{po}_1$
$u = 1$
while the stopping condition is not met **do**
  Let $\boldsymbol{d}_u$ group N individuals selected from $\boldsymbol{po}_u$ via the selection method
  Let $(\boldsymbol{s}_u, \widehat{\boldsymbol{\theta}}_{\boldsymbol{s}_u})$ be the BN for data clustering learnt from $\boldsymbol{d}_u$ via the BSEM algorithm
  Let $(\boldsymbol{s}_u, \overline{\boldsymbol{\theta}}_{\boldsymbol{s}_u})$ be $(\boldsymbol{s}_u, \widehat{\boldsymbol{\theta}}_{\boldsymbol{s}_u})$ with the exception that $p(c)$ has been modified to be uniform
  Let $\boldsymbol{of}_u$ be the offspring population composed of M individuals sampled from $(\boldsymbol{s}_u, \overline{\boldsymbol{\theta}}_{\boldsymbol{s}_u})$
  Evaluate the individuals in $\boldsymbol{of}_u$
  Let $\boldsymbol{po}_{u+1}$ be the population created from $\boldsymbol{po}_u$ and $\boldsymbol{of}_u$ via the replacement method
  $u++$
Return the best individuals found so far

Fig. 5. Pseudocode of the UEBNA.

tion to alleviate the poor performance of most EDAs for globally multimodal problem optimization: It allows modelling simultaneously the different basins that may be represented by the individuals selected at each iteration, whereas preventing genetic drift.

As outlined in Fig. 5, the UEBNA consists in the iteration of the same three main steps as the generic EDA (see Fig. 1): Selection of promising individuals from the current population, probabilistic modelling of the selected individuals, and model sampling to create the new population. The singularities of the u-th iteration of the UEBNA are as follows for all u:

- The l-th case of $\boldsymbol{d}_u$ is represented by an $(n+1)$-dimensional discrete vector $\boldsymbol{x}_l = (x_{l1}, \ldots, x_{ln+1})$ partitioned as $\boldsymbol{x}_l = (c_l, \boldsymbol{y}_l)$ for all l: c_l is the unobserved cluster membership, and $\boldsymbol{y}_l = (y_{l1}, \ldots, y_{ln})$ is the l-th selected individual. Therefore, every case in $\boldsymbol{d}$ can be seen as a partial instance of an $(n+1)$-dimensional discrete random variable $\boldsymbol{X} = (X_1, \ldots, X_{n+1})$ partitioned as $\boldsymbol{X} = (C, \boldsymbol{Y})$. This fits the discussion in Section 3.1.
- The BSEM algorithm should be provided with the number of clusters K underlying $\boldsymbol{d}_u$. In general, the higher the number of clusters, the higher the flexibility and expressivity but also the complexity of the model. Therefore, this parameter may have an impact on the performance of the UEBNA.
- When $(\boldsymbol{s}_u, \widehat{\boldsymbol{\theta}}_{\boldsymbol{s}_u})$ is simulated to obtain $\boldsymbol{of}_u$, the number of individuals produced from each cluster is determined by the probability distribution for the cluster random variable $p(c)$. This implies that the number of individuals sampled from each cluster is proportional to its size. This sampling scheme favors large clusters, no matter their average fitness, and makes the population drift towards them. This seems unreasonable because it promotes genetic drift, and the UEBNA is aimed at preventing this phenomenon as much as possible. The explanation is straightforward. In the absence of selective pressure, i.e. when there is not significant selective preference for a cluster over another, the stochastic nature of the selection method makes some clusters have more representatives than others in the pool of selected individuals. Therefore, sampling more individuals from those clusters that have more members promotes genetic drift. On the other hand, sam-

pling a number of individuals from each cluster that is proportional to its average fitness involves excessive selective pressure, i.e. those clusters with good average fitness are favored by the selection method but also by the sampling scheme, which is undesirable as well. With the purpose of avoiding promoting genetic drift and excessive selective pressure in the UEBNA, it seems justified to sample the same number of individuals from each of the clusters encoded by $(\boldsymbol{s}_u, \widehat{\boldsymbol{\theta}}_{\boldsymbol{s}_u})$. This is accomplished by first modifying $p(c)$ in $(\boldsymbol{s}_u, \widehat{\boldsymbol{\theta}}_{\boldsymbol{s}_u})$ to be a uniform distribution and, then, sampling the resulting model, here denoted by $(\boldsymbol{s}_u, \overline{\boldsymbol{\theta}}_{\boldsymbol{s}_u})$. A similar discussion can be found in [58] regarding the number of individuals that should be sampled from each cluster in an attempt to implement niching.

- $\boldsymbol{of}_u$ is constructed by restricting the M instances sampled from $(\boldsymbol{s}_u, \overline{\boldsymbol{\theta}}_{\boldsymbol{s}_u})$ to their values for $\boldsymbol{Y}$.

4 Experimental Evaluation

This section evaluates the UEBNA for symmetrical (globally multimodal) problem optimization. Specifically, the evaluation involves optimization problems that show what is known as *symmetry on the alphabet* or *spin-flip symmetry* [79]: An optimization problem contains spin-flip symmetry when gene-complementary solutions score the same fitness. Therefore, spin-flip symmetrical optimization problems are globally multimodal. Some optimization problems that show spin-flip symmetry and, thus, global multimodality are twomax problems, graph partitioning problems, random number partitioning problems and graph coloring problems. As reported in [54,58,78,79], this class of globally multimodal optimization problems are challenging for most EAs, including EDAs. The evaluation of the UEBNA for some symmetrical optimization problems should provide us with sufficient insight to assess whether or not the UEBNA performs effectively and efficiently for globally multimodal problem optimization.

When one wants to identify several global optima of a spin-flip symmetrical optimization problem, it is tempting to consider searching for just one of them and, then, obtain another global optima by just changing the genes of the global optimum discovered to their complementary values. This approach is discarded in the evaluation of the UEBNA for the following reasons. Firstly, this shortcut is based on the knowledge that the search space contains spin-flip symmetry. However, a key feature of EAs in general and EDAs in particular is that they do not make assumptions about the search space of the optimization problem at hand. Secondly, working in this way only addresses the quantitative side (effectiveness) of the problem optimization process, i.e. the number of global peaks discovered, while ignoring the qualitative side (efficiency), i.e. the convergence speed (see Section 2.3). Moreover, this shortcut

performs poorly even in terms of effectiveness if more than two global optima exist in the spin-flip symmetrical optimization problem at hand. Finally, this approach can not be applied to globally multimodal problem optimization in general. In other words, the knowledge of the search space being symmetrical is not used at all in the evaluation of the UEBNA.
This section first describes the evaluation setup and, then, presents the symmetrical optimization problems in the evaluation. Finally, the dynamics and performance of the UEBNA in these optimization problems are reported and discussed.

4.1 Evaluation Setup

The BSEM algorithm run at each iteration of the UEBNA restricts the search to TANB models for data clustering. Furhtermore, the BSEM algorithm should be provided with the number of clusters K underlying the set of selected individuals. As noted earlier, this can be seen as a parameter that sets the flexibility and expressive power of the models in the search space of the BSEM algorithm. In the evaluation, we consider different values of K in order to assess the impact of this parameter on the performance of the UEBNA. We start with $K = 2$ and increase it until no further improvement is observed.
The convergence criterion for the EM algorithm run at each iteration of the BSEM algorithm is satisfied when either the relative difference between successive values for $\log L(\boldsymbol{d} \mid \boldsymbol{\theta_s}, \boldsymbol{s})$ is less than 1 or 150 iterations are reached. Preliminary experiments with more demanding convergence criteria did not lead to significantly better results.
In the implementation of the UEBNA, we use HUGIN API version 3.1 [36] wherever probabilistic inference or sampling of TANB models for data clustering is required (e.g. steps 3 and 4 in Fig. 4 and step 8 in Fig. 5). This means that probabilistic inference is done as indicated in [38,45].
For comparison purposes, we benchmark the UEBNA against two well established EDAs, namely the *univariate marginal distribution algorithm (UMDA)* [41,51] and the *estimation of Bayesian network algorithm (EBNA)* [16,41,42]. The UMDA is based on the assumption that $p_u(\boldsymbol{x})$ factorizes as follows:

$$p_u(\boldsymbol{x}) = \prod_{i=1}^{n} p_u(x_i) \tag{3}$$

for all u. Moreover, $p_u(x_i)$ is restricted to be a univariate multinomial distribution whose parameters are estimated from $\boldsymbol{d}_u$ according to the ML criterion for all i. On the other hand, the EBNA reduces learning $p_u(\boldsymbol{x})$ from $\boldsymbol{d}_u$ to induction of a BN for $\boldsymbol{X}$ from $\boldsymbol{d}_u$ for all u. For this purpose, the EBNA runs a greedy hill-climbing search over BN structures for $\boldsymbol{X}$ considering all the possible additions, removals and reversals of a single arc at each point in the search. The score that guides the search is the Bayesian information criterion (BIC) [72]. The sample from the BN for $\boldsymbol{X}$ learnt at each iteration of the EBNA is obtained by probabilistic logic sampling [32].

The reasons for using the UMDA and the EBNA as benchmarks in the evaluation of the UEBNA are the following ones. First, both the UMDA and the EBNA have received much attention in the literature. Moreover, the EBNA is close in spirit to the UEBNA, as both are based on learning and simulation of BNs. Finally, the UMDA and the EBNA provide the opportunity to compare the performance of three different approaches for globally multimodal problem optimization: The UMDA is an EDA that neither encodes conditional dependencies nor implements niching, the EBNA is an EDA that can encode conditional dependencies but it does not use niching, and the UEBNA is an EDA that combines encoding of conditional dependencies with niching via unsupervised learning of BNs (see Section 2.3 and Section 3).

The three EDAs in the evaluation use *truncation selection* as the selection method, i.e. the most fit individuals in the current population are selected. Furthermore, the replacement method creates the new population by replacing the least fit individuals in the current one by all the offspring population. The algorithms stop when the relative difference between the sum of the objective function values of all the individuals of the population of two successive generations is 0 or 100 generations are reached. For the three EDAs in the evaluation, the population size, the number of selected individuals at each iteration, and the number of generated individuals at each iteration are 4000, 3000 and 3000, respectively. Preliminary experiments confirmed that these parameter values are well suited for the three EDAs in evaluation and that they do not favor any of the them over the rest. Having said this, using the same optimization schedule for all the EDAs in the evaluation eases comparison.

For each pair composed of one EDA and one globally multimodal optimization problem in the evaluation, the performance criteria measured are (i) the number of global optima discovered, (ii) the average deviation with respect to the expected number of individuals representing each global optima discovered, and (iii) the number of evaluations of the objective function and the runtime until convergence. The first two criteria reflect the effectiveness of the problem optimization process, while the last two criteria assesses its efficiency. The second performance criterion is calculated as follows:

$$\frac{1}{Op} \cdot \sum_{i=1}^{Op} \frac{|Q/Op - Q_i^*|}{Q/Op} \cdot 100 \tag{4}$$

where $|\cdot|$ is the absolute value function, Op is the number of global optima captured, Q is the population size, and Q_i^* is the number of individuals in the population of the last generation representing the i-th global optima discovered. As there is not selective preference for a global peak over another, it is desirable that those global optima identified are equally well represented in the population of the last generation, i.e. $Q_i^* \approx Q/Op$ for all i. Significant underrepresentation or overrepresentation of one or several of the global peaks discovered should be detected. Therefore, the closer the value of Eq. (4) to 0, the better. Likewise, the closer the value to 100, the worse.

4.2 Symmetrical Optimization Problems

The paragraphs below present the symmetrical optimization problems in the evaluation in terms of their search spaces and objective functions. These symmetrical optimization problems have been borrowed or adapted from [58,61].

4.2.1 Twomax Problem

The twomax problem is a simple symmetrical optimization problem whose search space is $\{0,1\}^n$, i.e. the set of binary strings of length n, and whose objective function is as follows:

$$F_{twomax}(\boldsymbol{z}) = F_{twomax}(z_1, \ldots, z_n) = \left| \frac{n}{2} - \sum_{i=1}^{n} z_i \right| . \tag{5}$$

The objective is maximization and there are two global optima: $\boldsymbol{z}_1^* = (0, \ldots, 0)$ and $\boldsymbol{z}_2^* = (1, \ldots, 1)$ with fitness equal to $\frac{n}{2}$. In all the EDAs in the evaluation, every solution $\boldsymbol{z}$ is represented by an n-dimensional binary individual where the i-th gene coincides with z_i for all i. The evaluation involves two instances of the twomax problem with $n = 50, 100$ and denoted by $P_{twomax50}$ and $P_{twomax100}$, respectively.

4.2.2 Graph Bisection Problem

The graph bisection problem aims to split the set of nodes of a given graph into two equally sized subsets so that the number of edges between the two subsets is minimized. Consequently, the search space of the graph bisection problem is the set of all the partitions of the nodes of the given graph into two equally sized subsets. The fitness of a given solution is calculated as the number of nodes in the graph at hand minus the number of edges connecting the two subsets of nodes in the solution. Thus, the objective is maximization. In all the EDAs in the evaluation, every solution $\boldsymbol{z}$ is represented by an n-dimensional binary individual where the i-th gene corresponds to the i-th node of the graph for bisection for all i. Then, each gene of a given individual classifies one of the nodes of the graph into one of the two subsets. Under this codification, the search space of the graph bisection problem can be represented as the set $\{(z_1, \ldots, z_n) \mid (z_1, \ldots, z_n) \in \{0,1\}^n$ and $\sum_{i=1}^{n} z_i = \frac{n}{2}\}$, where n is the number of nodes in the graph at hand. Note that only individuals

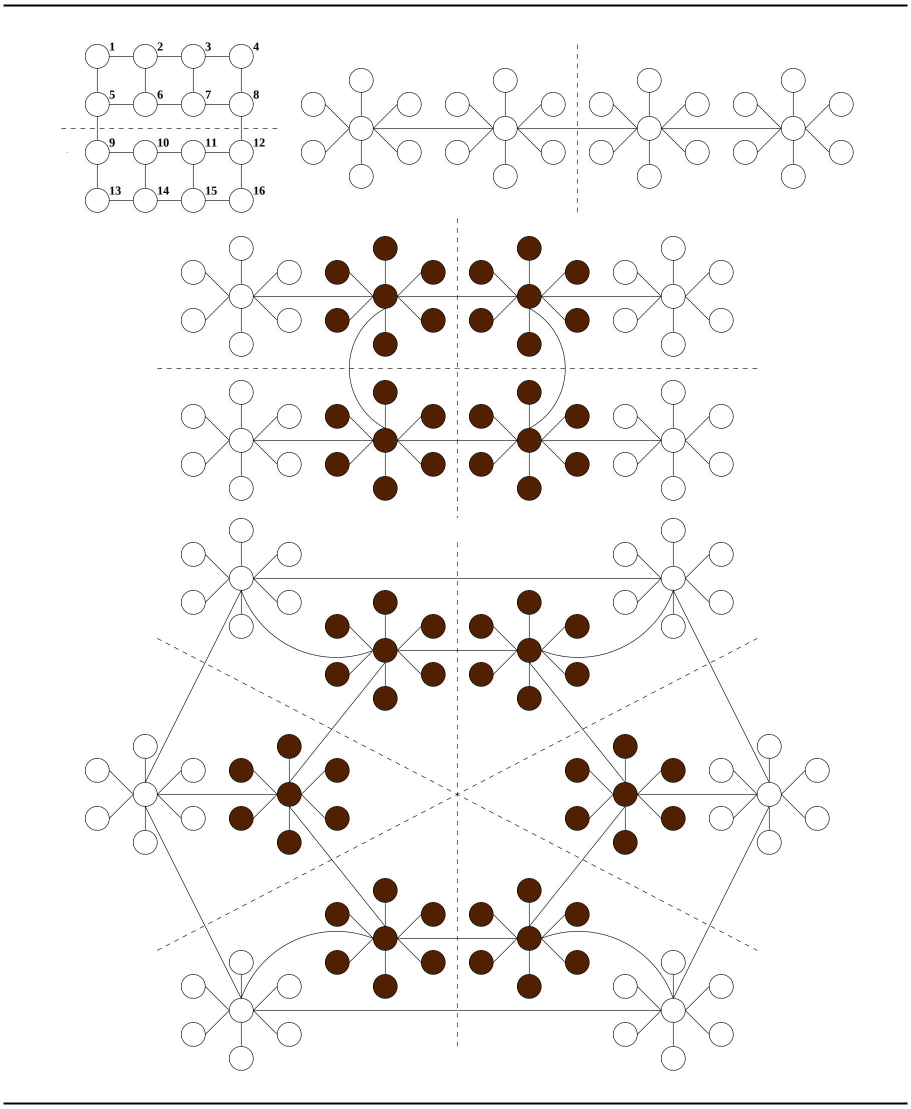

Fig. 6. Graphs for P_{grid16} (top, left), P_{cat28} (top, right), $P_{catring28}$ (middle, only dark nodes), $P_{catring56}$ (middle, all the nodes), $P_{catring42}$ (bottom, only dark nodes) and $P_{catring84}$ (bottom, all the nodes). Dashed lines indicate optimal cuts.

with equal number of zeroes and ones represent feasible solutions. However, the generation of the offspring at each iteration of the EDAs in the evaluation is not a closed operation with respect to this feasibility condition. Thus, some individuals that may appear during the problem optimization process may need to be repaired. A simple randomized repair operator is used in the EDAs in the evaluation: An unfeasible solution is converted into a feasible one by, iteratively, picking at random a gene in the majority and changing it to its complementary value until a feasible solution is obtained.
The evaluation involves 10 instances of the graph bisection problem. The first three instances consist of three grid-like graphs, with $n = 16, 36, 64$, cut in halves and connected by two edges. There are two global optima with fitness equal to $n - 2$. In the forthcoming, these optimization problems are denoted

by P_{grid16}, P_{grid36} and P_{grid64}, respectively. The evaluation also involves three so-called caterpillar graphs, with sizes $n = 28, 42, 56$, composed of four, six and eight, respectively, seven node star-shaped graphs connected in a line. There are two global optima with fitness equal to $n - 1$. In the forthcoming, these optimization problems are referred to as P_{cat28}, P_{cat42} and P_{cat56}, respectively. The last four instances of the graph bisection problem involve extensions of the caterpillar graphs so that there are more than two global optima. $P_{catring28}$ and $P_{catring56}$ involve graphs with $n = 28, 56$, respectively, and have four global optima with fitness equal to $n-2$. On the other hand, $P_{catring42}$ and $P_{catring84}$ involve graphs with $n = 42, 84$, respectively, and have six global optima whose fitness is equal to $n - 4$. Fig. 6 shows most of the graphs for bisection. In addition to the difficulties derived from their symmetrical nature, these instances of the graph bisection problem present another source of difficulties: They are highly multimodal and present many local optima and only a few global optima [58,61,73].

4.3 Results

Fig. 7 shows the dynamics of the UMDA, the EBNA and the UEBNA until convergence in one run for $P_{twomax50}$. The histograms summarize the number of solutions (vertical axis) in the population of different generations whose sum of genes is equal to the value of the horizontal axis. As previously stated, the two global optima of the optimization problem are gene-complementary and correspond to the left-most and right-most sides of the histograms. The histograms show that, as problem optimization progresses, the population drifts to one side in the case of the UMDA and to both sides in the case of the EBNA and the UEBNA. Moreover, the individuals of the population of the last generation of the EBNA and the UEBNA are almost equally distributed among both global peaks. It can also be seen in the histograms that genetic drift occurs so slowly that the UMDA takes longer than the other two EDAs to converge. This clearly confirms what has been argued in Section 2.3 about the necessity of considering EDAs based on either encoding of conditional dependencies or niching, or both, for effective and efficient globally multimodal problem optimization. Finally, it should be also mentioned that, although the EBNA and the UEBNA perform equally well in terms of effectiveness, they differ in their efficiency: The UEBNA reaches convergence faster than the EBNA. This suggests that TANB models for data clustering are more appropriate than BNs for modelling the joint probability distribution for the individuals selected at each iteration. Note, however, that the EBNA relies on unrestricted BNs, which can potentially model more complex conditional dependencies than TANB models for data clustering. Therefore, this supports that combining model-based data clustering with the ability to model conditional dependencies is more robust and reliable against genetic drift when

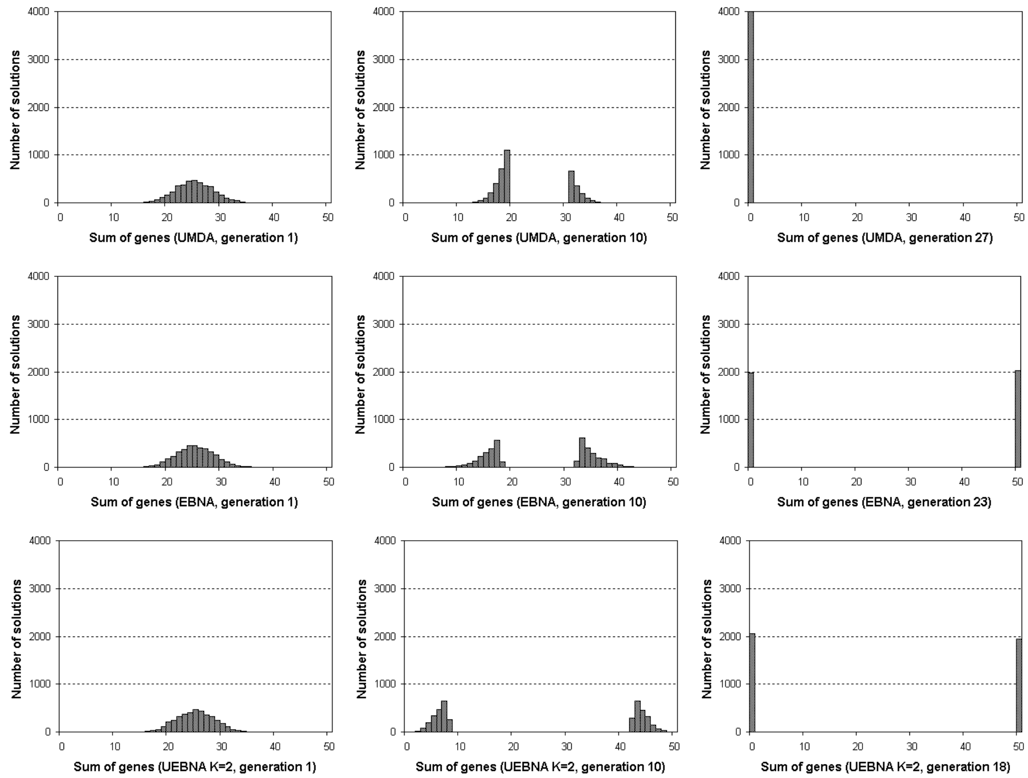

Fig. 7. Dynamics of the UMDA (top row), the EBNA (middle row), and the UEBNA (bottom row) until convergence in one run for $P_{twomax50}$. The horizontal axis of each histogram represents the sum of the genes of a solution, whereas the vertical axis denotes the number of corresponding solutions in the population of different generations.

incorporated into the EDA framework than the ability to model conditional dependencies alone. As discussed in Section 3, this is the main motivation behind the development of the UEBNA.

Fig. 8 provides the reader with additional evidence on the efficiency of the UEBNA for globally multimodal problem optimization. Specifically, the figure plots the average fitness of the individuals in the population as a function of the number of generations and until convergence for the UMDA, the EBNA and the UEBNA in one run for $P_{twomax100}$, $P_{catring28}$ and $P_{catring42}$. Note that the first optimization problem presents two global optima, the second four and the third six. These curves clearly show that the UEBNA speeds up converge without degrading the quality of the solutions obtained.

Figs. 7 and 8 illustrate the behavior of the UMDA, the EBNA and the UEBNA for a single run for $P_{twomax50}$, $P_{twomax100}$, $P_{catring28}$ and $P_{catring42}$. The remainder of the 10 independent runs performed for these optimization problems leads to the same conclusions as those discussed above. Moreover, the observed patterns can be extended to the 10 independent runs performed for the rest of the symmetrical optimization problems in the evaluation. For the sake

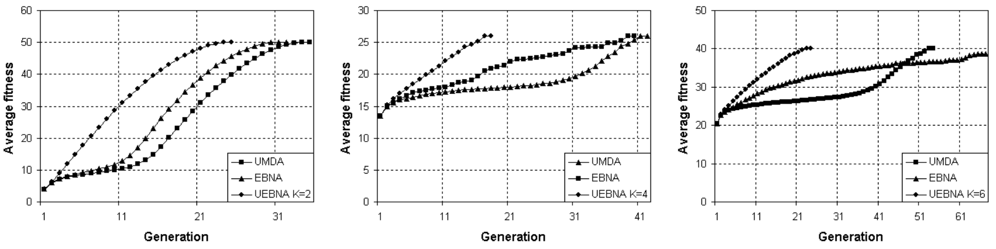

Fig. 8. Average fitness of the individuals in the population as a function of the number of generations and until convergence for the UMDA, the EBNA and the UEBNA in one run for $P_{twomax100}$ (left), $P_{catring28}$ (middle) and $P_{catring42}$ (right).

of brevity, figures are not reported. Instead, Tables 1 and 2 summarize the results that the UMDA, the EBNA and the UEBNA (with different values for K) achieve for each of the 12 symmetrical optimization problems in the evaluation. For each combination of one EDA and one symmetrical optimization problem in the evaluation, the tables report performance in terms of average and standard deviation (i) over the 10 independent runs performed (*All runs*), and (ii) over successful runs (*Successful runs*), i.e. over those runs out of the 10 independent runs performed where at least one global peak of the symmetrical optimization problem at hand is identified. For *All runs*, the performance criteria measured are the number of global optima discovered (*Optima*), the number of evaluations of the objective function until convergence (*Eval*) and the runtime in seconds until convergence ($Time$).[1] For *Successful runs*, the average deviation with respect to the expected number of individuals representing each global optima discovered (*Deviation*) is calculated as indicated in Eq. (4) and reported, in addition to *Optima*, *Eval* and $Time$. *Optima* and *Deviation* relate to the effectiveness of the EDAs, while *Eval* and $Time$ related to the efficiency. Finally, it should be mentioned that, when all the 10 independent runs performed for any of the symmetrical optimization problems in the evaluation are successful, the values for the performance criteria for *All runs* and *Successful runs* coincide. In this case, only the values for the performance criteria for *Successful runs* are reported, for the sake of readability.

Table 1 summarizes the effectiveness of the EDAs in the evaluation. The first conclusion that can be achieved from the results in the table is that the UEBNA enjoys higher rate of successful runs than the UMDA and the EBNA. In addition, the average number of global optima identified per run, i.e. *Optima* in *All runs*, indicates that the UEBNA outperforms by far both the UMDA and the EBNA in the 12 symmetrical optimization problems in the evaluation. The poor behavior of the UMDA and the EBNA illustrates that the globally multimodal optimization problems in the evaluation are challenging. The UEBNA behaves very effectively even in those optimization problems with four and six global peaks. Regarding effectiveness per successful run, i.e.

[1] All the experiments are run on a Pentium 900 MHz.

Table 1
Effectiveness of the UMDA, the EBNA and the UEBNA for the 12 symmetrical optimization problems in the evaluation. All the values are given in terms of average and standard deviation over 10 independent runs.

		All runs	*Successful runs*	
Problem	*EDA*	*Optima* ± *sd*	*Optima* ± *sd*	*Deviation* ± *sd*
$P_{twomax50}$ (2 global optima)	UMDA	— ± —	1.0 ± 0.0	0 ± 0
	EBNA	— ± —	1.5 ± 0.5	24 ± 37
	UEBNA K=2	— ± —	2.0 ± 0.0	1 ± 1
	UEBNA K=4	— ± —	2.0 ± 0.0	1 ± 1
$P_{twomax100}$ (2 global optima)	UMDA	— ± —	1.0 ± 0.0	0 ± 0
	EBNA	— ± —	1.0 ± 0.0	0 ± 0
	UEBNA K=2	— ± —	2.0 ± 0.0	1 ± 1
	UEBNA K=4	— ± —	2.0 ± 0.0	1 ± 1
P_{grid16} (2 global optima)	UMDA	— ± —	1.0 ± 0.0	0 ± 0
	EBNA	— ± —	2.0 ± 0.0	34 ± 32
	UEBNA K=2	— ± —	2.0 ± 0.0	2 ± 2
	UEBNA K=4	— ± —	2.0 ± 0.0	2 ± 2
P_{grid36} (2 global optima)	UMDA	0.7 ± 0.5	1.0 ± 0.0	13 ± 35
	EBNA	— ± —	1.8 ± 0.4	98 ± 4
	UEBNA K=2	— ± —	2.0 ± 0.0	2 ± 2
	UEBNA K=4	— ± —	2.0 ± 0.0	4 ± 6
P_{grid64} (2 global optima)	UMDA	0.2 ± 0.4	1.0 ± 0.0	0 ± 0
	EBNA	0.9 ± 0.6	1.1 ± 0.4	47 ± 51
	UEBNA K=2	1.4 ± 1.0	2.0 ± 0.0	9 ± 18
	UEBNA K=4	— ± —	2.0 ± 0.0	7 ± 14
P_{cat28} (2 global optima)	UMDA	0.9 ± 0.3	1.0 ± 0.0	0 ± 0
	EBNA	— ± —	2.0 ± 0.0	51 ± 30
	UEBNA K=2	— ± —	2.0 ± 0.0	3 ± 3
	UEBNA K=4	— ± —	2.0 ± 0.0	2 ± 1
P_{cat42} (2 global optima)	UMDA	0.6 ± 0.5	1.0 ± 0.0	0 ± 0
	EBNA	1.2 ± 1.0	2.0 ± 0.0	85 ± 22
	UEBNA K=2	— ± —	2.0 ± 0.0	2 ± 1
	UEBNA K=4	— ± —	2.0 ± 0.0	1 ± 1
P_{cat56} (2 global optima)	UMDA	0.5 ± 0.5	1.0 ± 0.0	0 ± 0
	EBNA	0.2 ± 0.6	2.0 ± 0.0	99 ± 0
	UEBNA K=2	— ± —	2.0 ± 0.0	9 ± 22
	UEBNA K=4	— ± —	2.0 ± 0.0	2 ± 2
$P_{catring28}$ (4 global optima)	UMDA	— ± —	1.0 ± 0.0	0 ± 0
	EBNA	2.8 ± 1.3	3.1 ± 0.9	81 ± 13
	UEBNA K=2	— ± —	4.0 ± 0.0	51 ± 9
	UEBNA K=4	— ± —	4.0 ± 0.0	28 ± 17
	UEBNA K=6	— ± —	4.0 ± 0.0	6 ± 12
$P_{catring56}$ (4 global optima)	UMDA	0.7 ± 0.5	1.0 ± 0.0	0 ± 0
	EBNA	1.0 ± 0.9	1.7 ± 0.5	94 ± 11
	UEBNA K=2	— ± —	2.9 ± 0.9	43 ± 39
	UEBNA K=4	— ± —	3.4 ± 0.7	6 ± 14
	UEBNA K=6	— ± —	3.7 ± 0.5	2 ± 1
	UEBNA K=8	— ± —	3.8 ± 0.4	2 ± 1
$P_{catring42}$ (6 global optima)	UMDA	— ± —	1.0 ± 0.0	0 ± 0
	EBNA	2.9 ± 1.7	3.2 ± 1.4	79 ± 17
	UEBNA K=2	— ± —	5.8 ± 0.6	51 ± 22
	UEBNA K=4	— ± —	5.6 ± 0.7	48 ± 12
	UEBNA K=6	— ± —	5.9 ± 0.3	25 ± 18
	UEBNA K=8	— ± —	5.8 ± 0.6	19 ± 22
$P_{catring84}$ (6 global optima)	UMDA	0.8 ± 0.4	1.0 ± 0.0	13 ± 35
	EBNA	0.2 ± 0.4	1.0 ± 0.0	50 ± 71
	UEBNA K=2	— ± —	2.2 ± 0.4	13 ± 24
	UEBNA K=4	— ± —	3.7 ± 0.7	10 ± 16
	UEBNA K=6	— ± —	4.3 ± 0.8	3 ± 1
	UEBNA K=8	— ± —	4.7 ± 1.0	5 ± 9
	UEBNA K=10	— ± —	4.8 ± 0.8	34 ± 20

Table 2
Efficiency of the UMDA, the EBNA and the UEBNA for the 12 symmetrical optimization problems in the evaluation. All the values are given in terms of average and standard deviation over 10 independent runs.

		All runs		*Successful runs*	
Problem	*EDA*	*Eval* ± *sd*	*Time* ± *sd*	*Eval* ± *sd*	*Time* ± *sd*
$P_{twomax50}$	UMDA	— ± —	— ± —	87100 ± 11229	157 ± 103
(2 global	EBNA	— ± —	— ± —	69700 ± 1703	416 ± 12
optima)	UEBNA K=2	— ± —	— ± —	55000 ± 0	421 ± 33
	UEBNA K=4	— ± —	— ± —	56200 ± 2098	1011 ± 106
$P_{twomax100}$	UMDA	— ± —	— ± —	117400 ± 17596	215 ± 50
(2 global	EBNA	— ± —	— ± —	98800 ± 5138	3320 ± 279
optima)	UEBNA K=2	— ± —	— ± —	76600 ± 1265	1976 ± 89
	UEBNA K=4	— ± —	— ± —	79900 ± 2470	5644 ± 463
P_{grid16}	UMDA	— ± —	— ± —	109300 ± 20418	201 ± 46
(2 global	EBNA	— ± —	— ± —	113200 ± 43317	215 ± 116
optima)	UEBNA K=2	— ± —	— ± —	53500 ± 2916	156 ± 21
	UEBNA K=4	— ± —	— ± —	51400 ± 2366	210 ± 23
P_{grid36}	UMDA	217900 ± 57150	488 ± 162	200286 ± 53996	436 ± 144
(2 global	EBNA	— ± —	— ± —	244900 ± 71653	1056 ± 323
optima)	UEBNA K=2	— ± —	— ± —	85600 ± 8462	620 ± 89
	UEBNA K=4	— ± —	— ± —	94000 ± 6782	909 ± 64
P_{grid64}	UMDA	299500 ± 12268	757 ± 57	281500 ± 23335	671 ± 67
(2 global	EBNA	249400 ± 61103	2801 ± 620	235750 ± 61120	2670 ± 630
optima)	UEBNA K=2	128200 ± 12506	2424 ± 313	123143 ± 7690	2286 ± 226
	UEBNA K=4	— ± —	— ± —	124900 ± 3479	3809 ± 483
P_{cat28}	UMDA	128200 ± 24008	215 ± 57	124333 ± 21915	207 ± 55
(2 global	EBNA	— ± —	— ± —	138100 ± 67765	386 ± 215
optima)	UEBNA K=2	— ± —	— ± —	57100 ± 2846	344 ± 35
	UEBNA K=4	— ± —	— ± —	60700 ± 949	435 ± 22
P_{cat42}	UMDA	175600 ± 26937	325 ± 59	166500 ± 21668	309 ± 51
(2 global	EBNA	238300 ± 74289	1196 ± 373	244000 ± 71875	1212 ± 344
optima)	UEBNA K=2	— ± —	— ± —	73900 ± 1449	829 ± 65
	UEBNA K=4	— ± —	— ± —	76900 ± 1449	1064 ± 85
P_{cat56}	UMDA	209200 ± 22812	427 ± 59	197200 ± 7823	396 ± 20
(2 global	EBNA	277000 ± 52612	2305 ± 423	160000 ± 0	1357 ± 0
optima)	UEBNA K=2	— ± —	— ± —	96700 ± 7675	1803 ± 242
	UEBNA K=4	— ± —	— ± —	94600 ± 2366	1956 ± 123
$P_{catring28}$	UMDA	— ± —	— ± —	127600 ± 16601	212 ± 35
(4 global	EBNA	203200 ± 90777	587 ± 290	192000 ± 88652	550 ± 281
optima)	UEBNA K=2	— ± —	— ± —	54700 ± 949	347 ± 35
	UEBNA K=4	— ± —	— ± —	58000 ± 2000	495 ± 60
	UEBNA K=6	— ± —	— ± —	59800 ± 2530	514 ± 63
$P_{catring56}$	UMDA	218200 ± 38761	423 ± 79	200714 ± 28028	389 ± 63
(4 global	EBNA	238000 ± 50060	1992 ± 428	240000 ± 38683	2028 ± 338
optima)	UEBNA K=2	— ± —	— ± —	97300 ± 4111	1761 ± 256
	UEBNA K=4	— ± —	— ± —	94600 ± 2757	1926 ± 177
	UEBNA K=6	— ± —	— ± —	96700 ± 3592	2349 ± 158
	UEBNA K=8	— ± —	— ± —	94600 ± 1897	2914 ± 216
$P_{catring42}$	UMDA	— ± —	— ± —	169000 ± 20000	313 ± 46
(6 global	EBNA	218800 ± 63815	1098 ± 323	221333 ± 67151	1111 ± 340
optima)	UEBNA K=2	— ± —	— ± —	73000 ± 1414	853 ± 157
	UEBNA K=4	— ± —	— ± —	73000 ± 0	1038 ± 85
	UEBNA K=6	— ± —	— ± —	75700 ± 3302	1218 ± 87
	UEBNA K=8	— ± —	— ± —	75700 ± 2627	1398 ± 152
$P_{catring84}$	UMDA	253900 ± 33438	601 ± 99	248875 ± 32189	583 ± 89
(6 global	EBNA	277900 ± 38963	5609 ± 752	260500 ± 23335	5486 ± 481
optima)	UEBNA K=2	— ± —	— ± —	123400 ± 4858	3944 ± 290
	UEBNA K=4	— ± —	— ± —	120100 ± 4483	4684 ± 221
	UEBNA K=6	— ± —	— ± —	124300 ± 3860	5567 ± 612
	UEBNA K=8	— ± —	— ± —	127900 ± 2470	6400 ± 515
	UEBNA K=10	— ± —	— ± —	121000 ± 3162	7535 ± 504

Optima and *Deviation* in *Successful runs*, the results compiled in the table support what has been discussed in the paragraphs above. The UMDA is ineffective for globally multimodal problem optimization, as at best a single global peak is identified per run. On the other hand, the EBNA and the UEBNA are able to discover several global optima per run. However, the results confirm the clear superiority of the UEBNA over the EBNA: On average, more global peaks are discovered per run and they are more equally represented in the population of the last generation. All this indicates that the UEBNA enjoys a robust and reliable behavior against genetic drift. Finally, it should be also observed that increasing the value of K for the UEBNA has a positive effect on the effectiveness, specially when optimizing the symmetrical problems with four and six global optima.

Table 2 summarizes the efficiency of the EDAs in the evaluation. Regarding the number of evaluations of the objective, i.e. $Eval$, the table shows that the UEBNA significantly speeds up convergence: The saving in number of evaluations that the UEBNA induces over the UMDA and the EBNA for any of the 12 symmetrical optimization problems is considerable. This proves that the UEBNA is able to alleviate genetic drift, accelerating convergence as a result. Furthermore, increasing the value of K for the UEBNA does not significantly increase $Eval$ (it is even reduced in some cases) while, as observed above, effectiveness does improve. Unfortunately, one iteration of the UEBNA is much more time consuming than one iteration of the UMDA or the EBNA, as can be appreciated from the total runtime in $Time$, because it involves running the EM algorithm at least once. In any case, for 10 out of the 12 optimization problems in the evaluation, the UEBNA scores lower runtime than the EBNA for at least one of the values for K considered. This means that the UEBNA can identify more global optima than the EBNA with less evaluations of the objective function and in a shorter runtime.

It is worth mentioning that our current implementation of the UEBNA, being a proof of concept only, can be considerably improved in terms of runtime by accelerating the EM algorithm, which is the most time consuming part of the code. This implies that the runtime for the UEBNA reported in Table 2 should be read as an upper bound. Several accelerated versions of the EM algorithm have been proposed in the literature [5,17,49]. The results reported in these papers illustrate that these techniques can reduce substantially the runtime without degrading significantly the quality of the ML parameters. We could further accelerate the EM algorithm by implementing this simple observation: The ML parameters do not usually change substantially between consecutive generations of the UEBNA. Therefore, we could use the ML parameters obtained in one generation in order to initialize the EM algorithm in the next generation. This should reduce the number of iterations of the EM algorithm to converge. By implementing these improvements, the advantages of the UEBNA will be even more apparent. In this paper, we are primarily interested in evaluating the effectiveness of the UEBNA as a proof of concept, while we consider the runtime a secondary performance criterion because it

depends very much on the implementation of the UEBNA. It is out of the score of this paper to compare different implementations.
As summary, it can be said that the UEBNA behaves effectively as well as efficiently for symmetrical problem optimization. Specifically, the results discussed above confirm that the UEBNA is able to alleviate genetic drift with the help of unsupervised learning of TANB models. This means that the UEBNA reduces the likelihood of suboptimal convergence, obtains several global optima per run, and speeds up convergence.

5 Conclusions

The main contribution of this paper is the introduction and evaluation of a new estimation of distribution algorithm (EDA), called unsupervised estimation of Bayesian network algorithm (UEBNA), for effective and efficient globally multimodal problem optimization. The main steps of the UEBNA are the same as those of any other EDA: Selection of promising individuals, probabilistic modelling of the selected individuals, and model sampling in order to create the new population. The only peculiarity of the UEBNA with respect to existing EDAs is being based on unsupervised learning of Bayesian networks (BNs) in order to model the selected individuals at each iteration. This makes the UEBNA able to model simultaneously the different basins that may be represented by the individuals selected at each iteration, whereas preventing genetic drift as much as possible, because this phenomenon is the main responsible for the poor performance of most evolutionary algorithms (EAs), including EDAs, when optimizing globally multimodal problems.
We evaluate the UEBNA for symmetrical (globally multimodal) problem optimization, which is known to be challenging for most EAs, including EDAs. We benchmark the UEBNA against two well established EDAs, namely the univariate marginal distribution algorithm and the estimation of Bayesian network algorithm. The results obtained confirm the ability of the UEBNA to reduce the likelihood of suboptimal convergence, obtain more global optima per run, and speed up convergence with respect to the two benchmarks. Thus, we can conclude that the UEBNA performs effectively and efficiently for symmetrical (globally multimodal) problem optimization.
In addition to BNs for data clustering, other classes of probabilistic graphical models for data clustering that may be considered within the EDA framework, as illustrated in Fig. 5, for globally multimodal problem optimization are mixtures of Bayesian networks [76,77], and Bayesian multinets and recursive Bayesian multinets for data clustering [63,68]. Despite having received little attention in the literature, these classes of probabilistic graphical models for data clustering enjoy greater flexibility and expressive power than BNs for data clustering: They can encode context-specific conditional

(in)dependencies, whereas BNs for data clustering can encode only context-non-specific conditional (in)dependencies. These models can be particularly useful when the optimization problems at hand are known to be globally multimodal but not necessarily symmetrical. This is a line of research we are currently studying.

In this paper, we focus on discrete domains. The vast majority of the existing EDAs for discrete problem optimization have been already adapted to continuous problem optimization (see [41] for a revision). Likewise, we can extend the UEBNA to deal with continuous globally multimodal problem optimization by replacing unsupervised learning of BNs at each iteration by unsupervised learning of conditional Gaussian networks [63,64,67,69]. Like a BN for data clustering, a conditional Gaussian network for data clustering consists of a constrained acyclic directed graph and a set of conditional probability distributions. Unlike BNs for data clustering, the conditional probability density functions for the unidimensional predictive random variables are linear regression models conditioned on the value of the cluster random variable. As a result, the generalized joint probability distribution encoded is a conditional Gaussian distribution [7,11,44]. Currently, our main line of research concerns empirical evaluation of this extension of the UEBNA for continuous globally multimodal problem optimization. Preliminary experiments look promising.

Acknowledgements

We thank the three anonymous referees for their useful suggestions. We also thank Marc Schoenauer for handling the review process.

References

[1] M. R. Anderberg. *Cluster Analysis for Applications.* Academic Press, 1973.

[2] S. Baluja. Population-Based Incremental Learning: A Method for Integrating Genetic Search Based Function Optimization and Competitive Learning. Technical Report CMU-CS-94-163, Carnegie Mellon University, 1994.

[3] S. Baluja and S. Davies. Using Optimal Dependency-Trees for Combinatorial Optimization: Learning the Structure of the Search Space. In *Proceedings of the Fourteenth International Conference on Machine Learning*, pages 30–38. Morgan Kaufmann Publishers, 1997.

[4] S. Baluja and S. Davies. Fast Probabilistic Modeling for Combinatorial Optimization. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, pages 469–476, 1998.

[5] E. Bauer, D. Koller, and Y. Singer. Update Rules for Parameter Estimation in Bayesian Networks. In *Proceedings of the Thirteenth Conference on Uncertainty on Artificial Intelligence*, pages 3–13. Morgan Kaufmann Publishers, 1997.

[6] J. Binder, D. Koller, S. Russell, and K. Kanazawa. Adaptive Probabilistic Networks with Hidden Variables. *Machine Learning*, 29:213–244, 1997.

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

[8] D. M. Chickering. Learning Bayesian Networks is NP-Complete. In *Proceedings of the Fifth International Workshop on Artificial Intelligence and Statistics*, pages 121–130. Springer-Verlag, 1996.

[9] D. M. Chickering. Learning Equivalence Classes of Bayesian-Network Structures. *Journal of Machine Learning Research*, 2:445–498, 2002.

[10] G. F. Cooper and E. Herskovits. A Bayesian Method for the Induction of Probabilistic Networks from Data. *Machine Learning*, 9:309–347, 1992.

[11] R. G. Cowell, A. P. Dawid, S. L. Lauritzen, and D. J. Spiegelhalter. *Probabilistic Networks and Expert Systems*. Springer-Verlag, 1999.

[12] J. S. De Bonet, C. L. Isbell, and P. Viola. MIMIC: Finding Optima by Estimating Probability Densities. *Neural Information Processing Systems*, 9, 1997.

[13] K. A. De Jong. *An Analysis of the Behavior of a Class of Genetic Adaptive Systems*. PhD Thesis, University of Michigan, 1975.

[14] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum Likelihood from Incomplete Data Via the EM Algorithm. *Journal of the Royal Statistical Society B*, 39(1):1–38, 1977.

[15] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. John Wiley and Sons, 1973.

[16] R. Etxeberria and P. Larrañaga. Global Optimization Using Bayesian Networks. In *Proceedings of the Second Symposium on Artificial Intelligence*, pages 332–339, 1999.

[17] J. Fischer and K. Kersting. Scaled CGEM: A Fast Accelerated EM. In *Proceedings of the Fourteenth European Conference on Machine Learning*, pages 133–144. Springer, 2003.

[18] L. J. Fogel. Autonomous Automata. *Industrial Research*, 4:14–19, 1962.

[19] L. J. Fogel. *On the Organization of Intellect*. PhD Thesis, University of California, 1964.

[20] N. Friedman. The Bayesian Structural EM algorithm. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, pages 129–138. Morgan Kaufmann Publishers, 1998.

[21] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian Network Classifiers. *Machine Learning*, 29:131–163, 1997.

[22] N. Friedman and D. Koller. Being Bayesian About Network Structure. A Bayesian Approach to Structure Discovery in Bayesian Networks. *Machine Learning*, 50(1):95–125, 2003.

[23] M. Gallagher, M. Frean, and T. Downs. Real-Valued Evolutionary Optimization Using a Flexible Probability Density Estimator. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 840–846. Morgan Kaufmann Publishers, 1999.

[24] S. Geman and D. Geman. Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.

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

[26] D. E. Goldberg, K. Deb, H. Kargupta, and G. R. Harik. Rapid, Accurate Optimization of Difficult Problems Using Fast Messy Genetic Algorithms. In *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 56–64. Morgan Kaufmann Publishers, 1993.

[27] D. E. Goldberg and J. Richardson. Genetic Algorithms with Sharing for Multimodal Function Optimization. In *Proceedings of the Second International Conference on Genetic Algorithms*, pages 41–49. Morgan Kaufmann Publishers, 1987.

[28] D. E. Goldberg and P. Segrest. Finite Markov Chain Analysis of Genetic Algorithms. In *Proceedings of the Second International Conference on Genetic Algorithms*, pages 1–8. Morgan Kaufmann Publishers, 1987.

[29] G. R. Harik. Linkage Learning Via Probabilistic Modeling in the ECGA. Technical Report IlliGAL No. 1999010, University of Illinois at Urbana-Champaign, 1999.

[30] J. A. Hartigan. *Clustering Algorithms.* John Wiley and Sons, 1975.

[31] D. Heckerman, D. Geiger, and D. M. Chickering. Learning Bayesian Networks: The Combination of Knowledge and Statistical Data. *Machine Learning*, 20:197–243, 1995.

[32] M. Henrion. Propagation of Uncertainty by Probabilistic Logic Sampling in Bayes' Networks. In *Uncertainty in Artificial Intelligence 2*, pages 149–164, 1988.

[33] C. Hocaoğlu and A. C. Sanderson. Multimodal Function Optimization Using Minimal Representation Size Clustering and Its Applications to Planning Multipaths. *Evolutionary Computation*, 5(1):81–104, 1997.

[34] J. H. Holland. *Adaptation in Natural and Artificial Systems.* The University of Michigan Press, 1975.

[35] J. Horn. Finite Markov Chain Analysis of Genetic Algorithms with Niching. In *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 110–117. Morgan Kaufmann Publishers, 1993.

[36] F. Jensen. *HUGIN API Reference Manual Version 3.1.* 1997.

[37] F. V. Jensen. *Bayesian Networks and Decision Graphs.* Springer-Verlag, 2001.

[38] F. V. Jensen, S. L. Lauritzen, and K. G. Olesen. Bayesian Updating in Causal Probabilistic Networks by Local Computations. *Computational Statistics Quarterly*, 5(4):269–282, 1990.

[39] E. J. Keogh and M. J. Pazzani. Learning Augmented Bayesian Classifiers: A Comparison of Distribution-Based and Classification-Based Approaches. In *Proceedings of the Seventh International Workshop on Artificial Intelligence and Statistics*, pages 225–230. Morgan Kaufmann Publishers, 1999.

[40] V. Kvasnicka, M. Pelikan, and J. Pospichal. Hill Climbing with Learning (An Abstraction of Genetic Algorithms). *Neural Network World*, 6:773–796, 1996.

[41] P. Larrañaga and J. A. Lozano (eds.). *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation.* Kluwer Academic Publishers, 2001.

[42] P. Larrañaga, R. Etxeberria, J. A. Lozano, and J. M. Peña. Combinatorial Optimization by Learning and Simulation of Bayesian Networks. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pages 343–352. Morgan Kaufmann Publishers, 2000.

[43] P. Larrañaga, C. M. H. Kuijpers, R. H. Murga, and Y. Yurramendi. Searching for the Best Ordering in the Structure Learning of Bayesian Networks. *IEEE Transactions on Systems, Man and Cybernetics*, 26(4):487–493, 1996.

[44] S. L. Lauritzen. *Graphical Models.* Clarendon Press, 1996.

[45] S. L. Lauritzen and D. J. Spiegelhalter. Local Computations with Probabilities on Graphical Structures and Their Application to Expert Systems. *Journal of the Royal Statistical Society B*, 50(2):157–224, 1988.

[46] F. G. Lobo, K. Deb, D. E. Goldberg, G. R. Harik, and L. Wang. Compressed Introns in a Linkage Learning Genetic Algorithm. In *Proceedings of the Third Annual Conference on Genetic Programming*, pages 551–558. Morgan Kaufmann Publishers, 1998.

[47] S. Mahfoud. Population Sizing for Sharing Methods. In *Foundations of Genetic Algorithms 3*. Morgan Kaufmann Publishers, 1994.

[48] G. J. McLachlan and K. E. Basford. *Mixture Models: Inference and Applications to Clustering.* Marcel Dekker, 1988.

[49] G. J. McLachlan and T. Krishnan. *The EM Algorithm and Extensions.* John Wiley and Sons, 1997.

[50] M. Meilă. *Learning with Mixtures of Trees.* PhD Thesis, Massachusetts Institute of Technology, 1999.

[51] H. Mühlenbein. The Equation for Response to Selection and Its Use for Prediction. *Evolutionary Computation*, 5(3):303–346, 1997.

[52] H. Mühlenbein, T. Mahnig, and A. Ochoa. Schemata, Distributions and Graphical Models in Evolutionary Optimization. *Journal of Heuristics*, 5:215–247, 1999.

[53] H. Mühlenbein and G. Paaß. From Recombination of Genes to the Estimation of Distributions I. Binary Parameters. In *Proceedings of Parallel Problem Solving from Nature IV*, pages 178–187, 1996.

[54] B. Naudts and J. Naudts. The Effect of Spin-Flip Symmetry on the Performance of the Simple GA. In *Proceedings of Parallel Problem Solving from Nature V*, pages 67–76. Springer-Verlag, 1998.

[55] J. D. Nielsen, T. Kočka, and J. M. Peña. On Local Optima in Learning Bayesian Networks. In *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence*, 2003.

[56] J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann Publishers, 1988.

[57] M. Pelikan. *Bayesian Optimization Algorithm: From Single Level to Hierarchy*. PhD Thesis, University of Illinois at Urbana-Champaign, 2002.

[58] M. Pelikan and D. E. Goldberg. Genetic Algorithms, Clustering, and the Breaking of Symmetry. In *Proceedings of Parallel Problem Solving from Nature VI*, pages 385–394. Springer-Verlag, 2000.

[59] M. Pelikan, D. E. Goldberg, and E. Cantú-Paz. BOA: The Bayesian Optimization Algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 525–532. Morgan Kaufmann Publishers, 1999.

[60] M. Pelikan, D. E. Goldberg, and F. G. Lobo. A Survey of Optimization by Building and Using Probabilistic Models. *Computational Optimization and Applications*, 21(1):5–20, 2000.

[61] M. Pelikan, D. E. Goldberg, and K. Sastry. Bayesian Optimization Algorithm, Decision Graphs, and Occam's Razor. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 519–526. Morgan Kaufmann Publishers, 2001.

[62] M. Pelikan and H. Mühlenbein. The Bivariate Marginal Distribution Algorithm. *Advances in Soft Computing-Engineering Design and Manufacturing*, pages 521–535, 1999.

[63] J. M. Peña. *On Unsupervised Learning of Bayesian Networks and Conditional Gaussian Networks*. PhD Thesis, University of the Basque Country, 2001.

[64] J. M. Peña, I. Izarzugaza, J. A. Lozano, E. Aldasoro, and P. Larrañaga. Geographical Clustering of Cancer Incidence by Means of Bayesian Networks and Conditional Gaussian Networks. In *Proceedings of the Eighth International Workshop on Artificial Intelligence and Statistics*, pages 266–271. Morgan Kaufmann Publishers, 2001.

[65] J. M. Peña, J. A. Lozano, and P. Larrañaga. Learning Bayesian Networks for Clustering by Means of Constructive Induction. *Pattern Recognition Letters*, 20(11-13):1219–1230, 1999.

[66] J. M. Peña, J. A. Lozano, and P. Larrañaga. An Improved Bayesian Structural EM Algorithm for Learning Bayesian Networks for Clustering. *Pattern Recognition Letters*, 21(8):779–786, 2000.

[67] J. M. Peña, J. A. Lozano, and P. Larrañaga. Performance Evaluation of Compromise Conditional Gaussian Networks for Data Clustering. *International Journal of Approximate Reasoning*, 28(1):23–50, 2001.

[68] J. M. Peña, J. A. Lozano, and P. Larrañaga. Learning Recursive Bayesian Multinets for Data Clustering by Means of Constructive Induction. *Machine Learning*, 47(1):63–89, 2002.

[69] J. M. Peña, J. A. Lozano, P. Larrañaga, and I. Inza. Dimensionality Reduction in Unsupervised Learning of Conditional Gaussian Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):590–603, 2001.

[70] I. Rechenberg. *Evolutionstrategie: Optimierung Technischer Systeme Nach Prinzipien der Biologischen Evolution*. Fromman-Holzboog Verlag, 1973.

[71] R. Santana and A. Ochoa. Dealing with Constraints with Estimation of Distribution Algorithms: The Univariate Case. In *Proceedings of the Second Symposium on Artificial Intelligence*, pages 378–384, 1999.

[72] G. Schwarz. Estimating the Dimension of a Model. *Annals of Statistics*, 6:461–464, 1978.

[73] J. Schwarz and J. Ocenasek. Experimental Study: Hypergraph Partitioning Based on the Simple and Advanced Algorithms BMDA and BOA. In *Proceedings of the Fifth International Conference on Soft Computing*, pages 124–130, 1999.

[74] H. P. Schwefel. *Numerical Optimization of Computer Models*. John Wiley and Sons, 1981.

[75] M. Soto, A. Ochoa, S. Acid, and L. M. de Campos. Introducing the Polytree Approximation of Distribution Algorithm. In *Proceedings of the Second Symposium on Artificial Intelligence*, pages 360–367, 1999.

[76] B. Thiesson, C. Meek, D. M. Chickering, and D. Heckerman. Learning Mixtures of Bayesian Networks. Technical Report MSR-TR-97-30, Microsoft Research, 1998.

[77] B. Thiesson, C. Meek, D. M. Chickering, and D. Heckerman. Learning Mixtures of DAG Models. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, pages 504–513. Morgan Kaufmann Publishers, 1998.

[78] C. Van Hoyweghen. Detecting Spin-Flip Symmetry in Optimization Problems. In *Theoretical Aspects of Evolutionary Computing*, pages 175–206. Springer-Verlag, 2001.

[79] C. Van Hoyweghen and B. Naudts. Symmetry in the Search Space. In *Proceedings of the Seventh IEEE International Conference on Evolutionary Computation*, pages 1072–1079. IEEE Press, 2000.