A review on probabilistic graphical models in evolutionary computation

Pedro Larrañaga · Hossein Karshenas
Concha Bielza · Roberto Santana

Abstract Thanks to their inherent properties, probabilistic graphical models are one of the prime candidates for machine learning and decision making tasks especially in uncertain domains. Their capabilities, like representation, inference and learning, if used effectively, can greatly help to build intelligent systems that are able to act accordingly in different problem domains. Evolutionary algorithms is one such discipline that has employed probabilistic graphical models to improve the search for optimal solutions in complex problems. This paper shows how probabilistic graphical models have been used in evolutionary algorithms to improve their performance in solving complex problems. Specifically, we give a survey of probabilistic model building-based evolutionary algorithms, called estimation of distribution algorithms, and compare different methods for probabilistic modeling in these algorithms.

Keywords Probabilistic graphical model · Bayesian network · Evolutionary computation · Estimation of distribution algorithm

1 Introduction

Probability theory has provided a sound basis for many of scientific and engineering tasks. Artificial intelligence, and more specifically machine learning, is one of the fields that has exploited probability to develop new theorems and algorithms. A popular class of probabilistic graphical models (PGMs), Bayesian networks, first introduced by Pearl (1985), combine graph and probability theories to obtain a more comprehensible representation of the joint probability distribution. This tool can point out useful modularities in the underlying problem and help accomplish the reasoning and decision making tasks especially in uncertain domains. The application of these useful tools has been further improved by different methods proposed for PGM inference (Lauritzen and Spiegelhalter 1988) and automatic induction from a set of samples (Cooper and Herskovits 1992).

Meanwhile, the difficult and complex problems existing in real-world applications have increased the demand for effective meta-heuristic algorithms that are able to achieve good (and not necessarily optimal) solutions by performing an intelligent search of the space of possible solutions. Evolutionary computation is one of the most successful of these algorithms that has achieved very good results across a wide range of problem domains. Applying their nature-inspired mechanisms, e.g., survival of the fittest or genetic crossover and mutation, on a population of candidate solutions, evolutionary approaches like genetic algorithms (Holland 1975) have been able to perform an effective and diverse search of the vast solution space of problems.

Estimation of distribution algorithms (EDAs) (Mühlenbein and Paaß 1996; Bosman and Thierens 1999; Larrañaga and Lozano 2001; Pelikan 2005; Lozano et al. 2006) are a new class of evolutionary algorithms developed by fusing the two disciplines. They have proven to be promising optimization algorithms for many difficult problems with high computational complexity. These algorithms explore the search space by building a probabilistic model from a set of selected candidate solutions. This probabilistic model will be used to sample new solutions. As the result, these algorithms will provide a model expressing the regularities of the problem structure, as well as the final solutions.

The aim of this paper is to review and discuss the most noteworthy part of EDAs, namely the probabilistic models and how they are employed for optimization. It should be evident that an EDA also involves other steps and parts (e.g. initial population, selection methods, diversity preservation techniques), that goes beyond the consideration of this paper. The rest of the paper is organized as follows. Section 2 briefly introduces some basic terminology and concepts related to probabilistic models in the context of Bayesian networks and reviews some of the learning techniques for these probabilistic models. The main revision of different probabilistic models in EDAs is presented in Sect. 3. Finally, Sect. 4 concludes the paper.

2 Probabilistic graphical models

Different types of PGMs have been introduced in the literature: Bayesian networks (Pearl 1985), Markov networks, dependency networks (Heckerman et al. 2001), chain graphs (Frydenberg 1990). This section discusses some of the important concepts related to probabilistic modeling, mainly in the context of Bayesian networks, as one of the most prominent probabilistic models. It also gives a better understanding of how probabilistic modeling techniques are used in EDAs. For more information on PGMs, see Koller and Friedman (2009) and Larrañaga and Moral (2011).

2.1 Probability-related notations

Let $\boldsymbol{X} = (X_1, \ldots, X_n)$ be a vector of random variables and $\boldsymbol{x} = (x_1, \ldots, x_n)$ a possible value setting (configuration) for these variables. x_i denotes a possible value of X_i, the ith component of $\boldsymbol{X}$, and $\boldsymbol{y}$ denotes a possible value setting for the sub-vector $\boldsymbol{Y} = (X_{J_1}, \ldots, X_{J_k})$, $J = \{J_1, \ldots, J_k\} \subseteq \{1, \ldots, n\}$.

If all variables in $\boldsymbol{X}$ are discrete, $P(\boldsymbol{X} = \boldsymbol{x})$ (or simply $P(\boldsymbol{x})$) is used to denote the *joint probability mass* of a specific configuration $\boldsymbol{x}$ for the variables. The *conditional probability mass* of a specific value x_i of variable X_i given that $X_j = x_j$ is denoted by $P(X_i = x_i \mid X_j = x_j)$ (or simply $P(x_i \mid x_j)$). Similarly, for continuous variables, the *joint density function* will be denoted as $p(\boldsymbol{x})$ and the *conditional density function* by $p(x_i \mid x_j)$. When the nature of variables in $\boldsymbol{X} = (X_1, \ldots, X_n)$ is irrelevant, $\rho(\boldsymbol{x}) = \rho(x_1, \ldots, x_n)$ will be used to represent the generalized joint probability.

Let $\boldsymbol{Y}$, $\boldsymbol{Z}$ and $\boldsymbol{W}$ be three disjoint sub-vectors of variables. Then, $\boldsymbol{Y}$ is said to be *conditionally independent* of $\boldsymbol{Z}$ given $\boldsymbol{W}$ (denoted by $I(\boldsymbol{Y}, \boldsymbol{Z} \mid \boldsymbol{W})$), iff $\rho(\boldsymbol{y} \mid \boldsymbol{z}, \boldsymbol{w}) = \rho(\boldsymbol{y} \mid \boldsymbol{w})$, for any $\boldsymbol{y}$, $\boldsymbol{z}$ and $\boldsymbol{w}$.

2.2 Bayesian networks

A Bayesian network $\mathcal{B}(\mathcal{S}, \Theta)$ for a vector of variables $\boldsymbol{X} = (X_1, \ldots, X_n)$ consists of two components:

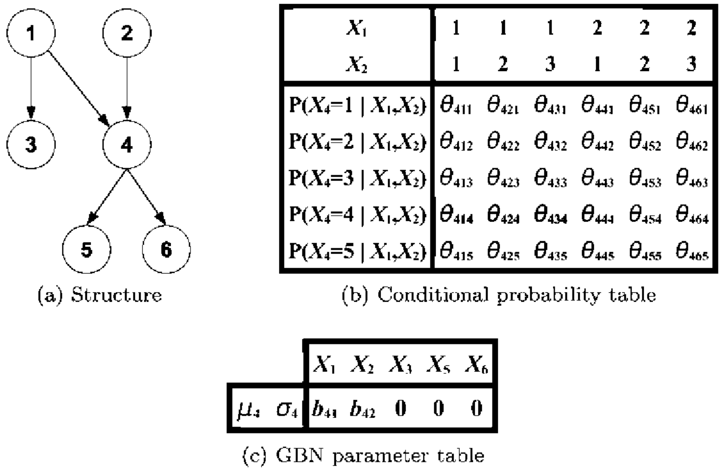
- A structure $\mathcal{S}$ represented by a directed acyclic graph (DAG), expressing a set of conditional (in)dependences (Dawid 1979) between variables.
- A set of local parameters Θ containing, for each variable, the conditional probability distribution of its values given different value settings for its parents, according to structure $\mathcal{S}$.

Figure 1(a) shows an example of Bayesian network structure for a problem with six variables. For each variable X_i, $i = 1, \ldots, n$, structure $\mathcal{S}$ represents the assertions that X_i and its non-descendants $\mathrm{ND}(X_i)$ (excluding its parents) are conditionally independent given its parents $\boldsymbol{Pa}_i$; i.e., $I(X_i, \mathrm{ND}(X_i) \setminus \boldsymbol{Pa}_i \mid \boldsymbol{Pa}_i)$. Therefore, a Bayesian network encodes a factorization for the joint probability distribution of the variables

$$\rho(\boldsymbol{x}) = \rho(x_1, \ldots, x_n) = \prod_{i=1}^{n} \rho_{\mathcal{B}}\big(x_i \mid \boldsymbol{pa}_i\big), \tag{1}$$

where $\boldsymbol{pa}_i$ denotes a possible value setting for the parents $\boldsymbol{Pa}_i$. Equation 1 states that the joint probability distribution of the variables represented by a Bayesian network can be computed as the product of each variable's univariate conditional probability distributions given the values of its parents. These conditional probabilities are stored as local parameters $\boldsymbol{\theta}_i$ in the Bayesian network.

In discrete domains, when a variable X_i has r_i possible values, $\{x_i^1, \ldots, x_i^{r_i}\}$, and, according to structure $\mathcal{S}$, its parents $\boldsymbol{Pa}_i$ have q_i possible combinations of values, $\{\boldsymbol{pa}_i^1, \ldots, \boldsymbol{pa}_i^{q_i}\}$, then $P_{\mathcal{B}}(x_i^k \mid \boldsymbol{pa}_i^j) \equiv \theta_{ijk}$ denotes the probability of X_i being in its kth value given that its parents are in their jth value combination. Since all variables are discrete, the number of possible value combinations for the parents can be easily



(a) Structure

X_1	1	1	1	2	2	2
X_2	1	2	3	1	2	3
P(X_4=1 | X_1,X_2)	θ_{411}	θ_{421}	θ_{431}	θ_{441}	θ_{451}	θ_{461}
P(X_4=2 | X_1,X_2)	θ_{412}	θ_{422}	θ_{432}	θ_{442}	θ_{452}	θ_{462}
P(X_4=3 | X_1,X_2)	θ_{413}	θ_{423}	θ_{433}	θ_{443}	θ_{453}	θ_{463}
P(X_4=4 | X_1,X_2)	θ_{414}	θ_{424}	θ_{434}	θ_{444}	θ_{454}	θ_{464}
P(X_4=5 | X_1,X_2)	θ_{415}	θ_{425}	θ_{435}	θ_{445}	θ_{455}	θ_{465}

(b) Conditional probability table

		X_1	X_2	X_3	X_5	X_6
μ_4	σ_4	b_{41}	b_{42}	0	0	0

(c) GBN parameter table

Fig. 1 (**a**) An example of a Bayesian network structure, showing two possible types of parameters for one of its variables (X_4); (**b**) Discrete variables, assuming that $r_i = i + 1$; (**c**) Continuous Gaussian variables

computed as $q_i = \prod_{X_m \in \boldsymbol{Pa}_i} r_m$. The local parameters of the Bayesian network for the ith variable can be represented by $\boldsymbol{\theta}_i = ((\theta_{ijk})_{k=1}^{r_i})_{j=1}^{q_i}$. Figure 1(b) shows an example of a conditional probability table for a discrete variable in a Bayesian network.

2.2.1 Gaussian Bayesian networks

For continuous domains, when it is assumed that $\boldsymbol{X} = (X_1, \ldots, X_n)$ is an n-dimensional Gaussian random variable, the Bayesian network used for encoding the joint density function is called a *Gaussian Bayesian network* (GBN) (Geiger and Heckerman 1994). The local probability density functions of the variables in a GBN is obtained using *linear regression models*. Specifically, the conditional density for variable X_i given the values of its parents $\boldsymbol{pa}_i$ is

$$p_{\mathcal{B}}(x_i \mid \boldsymbol{pa}_i) \rightsquigarrow \mathcal{N}\left(x_i; \mu_i + \sum_{X_m \in \boldsymbol{Pa}_i} b_{i,m}(x_m - \mu_m), \sigma_i^2\right), \tag{2}$$

where $\mathcal{N}(x; \mu, \sigma^2)$ represents a normal distribution with mean μ and standard deviation σ ($\sigma > 0$). Thus, the parameters of the local probability density function for every X_i can be determined by the triplet $\boldsymbol{\theta}_i = (\mu_i, \boldsymbol{b}_i, \sigma_i)$:

1. μ_i is the mean of variable X_i,
2. $\boldsymbol{b}_i$ is a vector of size $n - 1$, where every $b_{i,m}$ is a coefficient reflecting the strength of the linear relationship between X_m and X_i if $X_m \in \boldsymbol{Pa}_i$, and $b_{i,m} = 0$ otherwise, and
3. σ_i^2 is the variance of variable X_i.

Figure 1(c) shows an example of the parameters for a node of a Gaussian Bayesian network.

2.3 Learning Bayesian networks

This section briefly introduces Bayesian network learning methods, which will be useful later when we discuss probabilistic modeling in EDAs, and can give an idea of the complexity of model learning in general. The structure and conditional probabilities necessary for characterizing a Bayesian network can be provided either externally by experts, which is time consuming and error prone, or by automatic learning from a database of samples. The task of learning a Bayesian network can be divided into two subtasks:

- *structural learning*, i.e., identification of the topology of the Bayesian network, and
- *parametric learning*, estimation of the numerical parameters (conditional probabilities) for a given network topology.

The different methods proposed for inducing a Bayesian network from a dataset are usually classified by modeling type into two approaches:

1. methods based on detecting conditional (in)dependencies, also known as constraint-based methods, and
2. score+search methods.

2.3.1 Constrained-based methods

The input of these algorithms is a set of conditional (in)dependence relations between subsets of variables, which they use to build a Bayesian network that represents a large percentage (and, whenever possible, all) of these relations (Spirtes et al. 2001). The PC algorithm (Spirtes and Glymour 1991) is a well-known example of these methods. Typically, hypothesis tests are used to find conditional (in)dependencies from a dataset. Once the structure has been learned, the conditional probability distributions, required to fully specify the Bayesian network model are estimated from the dataset. The usual method for estimating the parameters is maximum likelihood estimation, although Laplace estimation and other Bayesian estimation approaches based on Dirichlet priors are also common.

2.3.2 Score+search methods

Constraint-based learning is quite an appealing approach as it is close to the semantics of Bayesian networks. However, most of the developed structure learning algorithms fall into the score+search method category. As the name implies, these methods have two major components:

1. a scoring metric that measures the quality of every candidate Bayesian network with respect to a dataset, and
2. a search procedure to intelligently move through the space of possible networks, as this space is enormous (see below for further discussion).

Scoring metrics Most of the popular scoring metrics are based on one of the following approaches: (i) penalized maximum likelihood, and (ii) marginal likelihood. *Penalized maximum likelihood* is computed as follows:

$$P_{\mathcal{B}}(\mathcal{D}) = \prod_{i=1}^{n} \prod_{j=1}^{q_i} \prod_{k=1}^{r_i} \left(\frac{N_{ijk}}{N_{ij}} \right)^{N_{ijk}} - f(N)\,\mathrm{dim}(\mathcal{B}), \tag{3}$$

where $\mathcal{D}$ is a dataset of N samples each consisting of n variables, N_{ij} is the number of samples in this dataset that have the jth value setting for the parents of the ith variable, and likewise N_{ijk} is the number of samples with the ith variable in its kth state and its parents in their jth configuration. $\mathrm{dim}(\mathcal{B})$ is the dimension (number of parameters needed to specify the model) of the Bayesian network. If the number of different states for the ith variable is given by r_i and the number of possible configurations for its parents is given by q_i, then the dimension of Bayesian network can be computed as $\mathrm{dim}(\mathcal{B}) = \sum_{i=1}^{n} q_i(r_i - 1)$. $f(N)$ is a non-negative penalization function depending on the size of the dataset. Popular scoring metrics like Akaike's information criterion (AIC) (Akaike 1974) and the Bayesian information criterion (BIC) (Schwarz 1978) differ as to their choice for this penalization function with values $f(N) = 1$ and $f(N) = \frac{1}{2}\log N$, respectively.

Assuming certain prior distributions for the parameters in the Bayesian network, the *marginal likelihood* of a specific network structure $\mathcal{S}$ given a dataset of samples, $P_{\mathcal{B}(\mathcal{S})}(\mathcal{D})$, can be computed in closed form (Cooper and Herskovits 1992; Heckerman et al. 1995). A common prior probability assumption is the Dirichlet distribution with parameters α_{ijk}, resulting in the following scoring metric (and assuming a uniform prior distribution for the structures) also known as the Bayesian Dirichlet equivalence (BDe) metric (Heckerman et al. 1995):

$$P_{\mathcal{B}(\mathcal{S})}(\mathcal{D}) = \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{\Gamma(\alpha_{ij})}{\Gamma(N_{ij} + \alpha_{ij})} \prod_{k=1}^{r_i} \frac{\Gamma(N_{ijk} + \alpha_{ijk})}{\Gamma(\alpha_{ijk})}, \tag{4}$$

where $\Gamma(v)$ is the Gamma function which for $v \in \mathbb{N}$ is given by $\Gamma(v) = (v-1)!$, and $\alpha_{ij} = \sum_{k=1}^{r_i} \alpha_{ijk}$. In the specific case where all Dirichlet distribution parameters are uniformly set to $\alpha_{ijk} = 1$, the resulting scoring metric is usually called K2 metric, initially proposed for use in the K2 algorithm (Cooper and Herskovits 1992).

Minimum description length (MDL) score (Rissanen 1978; Grünwald 1998) is another type of scoring metric based on information theory and data compression. This score, which is justified by Occam's razor principle less complex models, is closely related to the logarithm of the penalized maximum likelihood metric. In simple terms this metric can be described as follows. Suppose that the cost of encoding a set of data $\mathcal{D}$ with a model $\mathcal{B}$ is equal to the cost of describing the model plus the cost of describing the data with this model: $\mathrm{Cost}(\mathcal{B}) + \mathrm{Cost}(\mathcal{D} \mid \mathcal{B})$. Then the MDL score tries to select the model with the least total cost of description. Usually, the cost is expressed in terms of the number of bits required to represent the description.

A feature of scoring metrics that can greatly help the search algorithm is decomposability. With a decomposable metric, the score of a Bayesian network can be computed as the combination of scores obtained for smaller factors (e.g., a single variable). This property will allow the search algorithm to measure the effect of operations involving each factor independently of the effect of other network factors. The metrics introduced here are all decomposable.

Search methods Most of the proposed score+search algorithms search the *space of DAGs*, which represent feasible Bayesian network structures. The number of possible structures in this space for an n-dimensional variable is given by the following recursive formula (Robinson 1977):

$$\begin{aligned} f(n) &= \sum_{i=1}^{n}(-1)^{i+1}\binom{n}{i}2^{i(n-i)}f(n-i) \\ f(0) &= 1, \quad f(1) = 1 \end{aligned} \tag{5}$$

In fact it has been shown that searching this huge space for the optimal structure (according to a scoring metric) is NP-hard, even with a constrained maximum number of parents for each node (Chickering 1996; Chickering et al. 1994, 2004). Therefore, greedy local search techniques (Buntine 1991; Cooper and Herskovits 1992), as well as many heuristic search methods such as simulated annealing (Heckerman et al. 1995), tabu search (Bouckaert 1995) and evolutionary computation have been frequently employed for this purpose in the literature.

2.4 Markov networks

Markov networks are a type of probabilistic graphical models for representing symmetric influences between variables. A Markov network $\mathcal{M}(\mathcal{S}, \Phi_{\mathcal{C}})$ has two components:

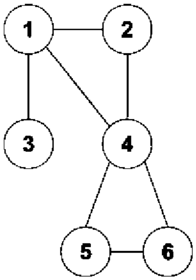
- an undirected graphical structure $\mathcal{S}$, where each variable is depicted by a node and the undirected edges represent homogeneous dependencies between the variables, and
- a set of factors (non-negative functions) $\Phi_{\mathcal{C}}$, each defined over a clique of $\mathcal{S}$, that express the compatibility of the values of their associated variables.

Figure 2 shows an exemplary Markov network structure and the parameters for one of its factors.

The set of factors $\Phi_{\mathcal{C}}$ can be used to define the joint probability distribution encoded in the Markov network. Let $\mathcal{C} = \{\boldsymbol{C}_1, \ldots, \boldsymbol{C}_\kappa\}$ be a set of cliques (complete subgraphs) of the Markov network structure, such that $\bigcup_{i=1}^{\kappa} \boldsymbol{C}_i = \boldsymbol{X}$. Then, the so-called *Gibbs* distribution $\rho_{\Phi_{\mathcal{C}}}$, parameterized by the set of factors $\Phi_{\mathcal{C}} = \{\phi_1(\boldsymbol{C}_1), \ldots, \phi_\kappa(\boldsymbol{C}_\kappa)\}$, that factorizes over the Markov network is given by

$$\rho_{\Phi_{\mathcal{C}}}(\boldsymbol{x}) = \frac{1}{Z}\prod_{i=1}^{\kappa}\phi_i(\boldsymbol{x}_{\boldsymbol{C}_i}), \tag{6}$$

where Z is a normalizing term, called the *partition function*, and is obtained by summing (or integrating for continuous domains) the unnormalized product of factors over all possible configurations of the variables. It should be noted that the Markov



(a) Structure

X_1	X_2	X_4	$\Phi(X_1, X_2, X_4)$
1	1	1	$\Phi_{1,2,4\text{-}1}$
1	2	2	$\Phi_{1,2,4\text{-}2}$
1	3	1	$\Phi_{1,2,4\text{-}3}$
1	1	2	$\Phi_{1,2,4\text{-}4}$
1	2	1	$\Phi_{1,2,4\text{-}5}$
1	3	2	$\Phi_{1,2,4\text{-}6}$
2	1	1	$\Phi_{1,2,4\text{-}7}$
2	2	2	$\Phi_{1,2,4\text{-}8}$
2	3	1	$\Phi_{1,2,4\text{-}9}$
2	1	2	$\Phi_{1,2,4\text{-}10}$
2	2	1	$\Phi_{1,2,4\text{-}11}$
2	3	2	$\Phi_{1,2,4\text{-}12}$

(b) Parameter table

Fig. 2 An example of a Markov network with a parameter table for the factor $\{X_1, X_2, X_4\}$. It is assumed that X_1, X_2 and X_4 respectively have 2, 3 and 2 possible states

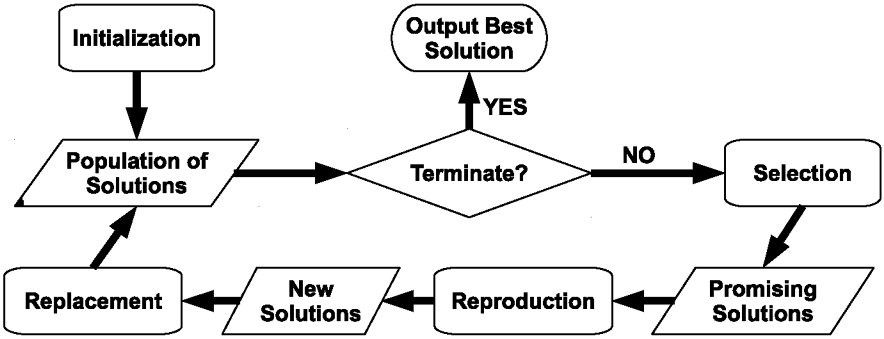


Fig. 3 Flowchart of a typical evolutionary algorithm

network factors do not necessarily correspond to probabilities or conditional probabilities.

3 Estimation of distribution algorithms

Over the last few decades several types of evolutionary algorithms (EAs), like genetic algorithms (GA) (Holland 1975), evolutionary strategies (ES) (Rechenberg 1973), evolutionary programming (EP) (Fogel 1966) and genetic programming (Cramer 1985; Koza 1992) have been proposed. All of these meta-heuristic algorithms, more or less, follow the same framework adopted from natural evolution. Figure 3 shows the common steps taken by a typical evolutionary algorithm for solving a problem. Given a fitness function that evaluates the quality of solutions, the algorithm iteratively evolves a population of candidate solutions to the problem. New offspring solutions are reproduced from the fitter solutions of the population (survival of the fittest principle) by applying genetic operators, i.e. crossover and mutation.

The simple and easy to understand mechanism of EAs along with their successful application to different problems in a variety of domains, has brought a lot of attention and interest to these algorithms. A key to the success of EAs is the identification, preservation and effective combination of the fitter partial solutions to the problem during evolution (Harik et al. 1999). However, it has been shown that the operators used in traditional EAs fail to properly accomplish this task when certain characteristics are present in the problem. A main reason for this shortcoming is that these algorithms do not properly consider the dependencies and relationships between the variables of the problem, and are not able to thoroughly exploit the information obtained so far, up to the current stage of the search, in order to speed up convergence. There are properties like non-linearity, ill-conditioning and deception in real world problems that without considering these types of regularities can pose significant challenges to traditional EAs.

As it was discussed in Sect. 2, probabilistic modeling offers a systematic way of acquiring this kind of regularities, and therefore can help to achieve a quick, accurate and reliable problem solving (Goldberg 2002; Pelikan et al. 2002). For this purpose, instead of genetic operators used in traditional EAs, new candidate solutions to the problem in each iteration are generated using the following two steps:

1. Estimating a probabilistic model based on the statistics collected from the set of candidate solutions, and
2. Sampling the learnt probabilistic model.

In this way the problem regularities encoded in the probabilistic model are used when generating new solutions, thus trying to overcome the shortcomings of traditional EAs. The incorporation of probabilistic modeling into EAs, has led to a new paradigm in evolutionary computation, usually referred to as *estimation of distribution algorithms* (EDAs). Algorithm 1 shows the basic steps of an EDA. Worthy of note is that probabilistic models could also be used in EAs for other purposes, for instance to make decisions on the application of mutation operators, to assess the influence of the different EA parameters on the algorithm behavior, or even to implement local optimization procedures.

Implicitly, EDAs assume that it is possible to model the promising areas of the search space, and to use this model to guide the search for the optimal solution(s). The probabilistic model learnt in EDAs captures an abstract representation of the features shared by the selected solutions and encodes the different patterns of interactions between subsets of the problem variables. The advantage of EDAs over other non-model-based EAs in dealing with the problems that contain important interactions among their variables, together with the capacity to solve different types of problems in a robust and scalable manner (Lozano et al. 2006; Pelikan 2005), has greatly popularized these algorithms. There are also many EDA implementations available online, like the EDA toolbox for Matlab® (Santana et al. 2010), which can be adopted for specific uses. Santana (2011) surveys some of the available softwares.

Because of the different nature of both optimization and probabilistic modeling in discrete and continuous domains, EDAs developed for each of these domains

```
ESTIMATION OF DISTRIBUTION ALGORITHM
Inputs:
    A representation of solutions,
    An objective function $f$
$P_0 \leftarrow$ Generate initial population according to the given representation
$F_0 \leftarrow$ Evaluate each individual $\boldsymbol{x}$ of $P_0$ using $f$
$g \leftarrow 1$
while termination criteria not met do
    $S_g \leftarrow$ Select a subset of $P_{g-1}$ according to $F_{g-1}$ using a selection mechanism
    $\hat{\rho}_g(\boldsymbol{x}) \leftarrow$ Estimate the probability of solutions in $S_g$
    $Q_g \leftarrow$ Sample $\hat{\rho}_g(\boldsymbol{x})$ according to the given representation
    $H_g \leftarrow$ Evaluate $Q_g$ using $f$
    $P_g \leftarrow$ Replace $Q_g$ in $P_{g-1}$ according to $F_{g-1}$ and $H_g$
    $F_g \leftarrow$ Update $F_{g-1}$ according to the solutions in $P_g$
    $g \leftarrow g+1$
end while
Output: The best solution in $P_{g-1}$
```

Algorithm 1: The basic steps of an estimation of distribution algorithm

also have differences depending on the representation type they use for the problem. Therefore, each of these two categories are discussed separately in Sects. 3.1 and 3.2. Note that this paper does not intend to give an exhaustive list of all proposed EDAs. Rather it tries to review the different probabilistic models and machine learning methods employed in EDAs. Another common way of categorizing EDAs is by the complexity of the probabilistic models they use. In general, one of the rationales in EDA development has been to find a satisfactory trade-off between the complexity of the probabilistic models they use and how accurately these models represent particular optimization problem characteristics. This is another factor taken into account here in reviewing EDAs. Moreover, for readability we use the algorithm acronyms. Table 1 lists the algorithms full names.

3.1 Discrete EDAs

Early EDAs were developed for discrete and especially binary domains, as it is a common practice in EAs to represent problem solutions with bit strings. *Univariate EDAs*, such as PBIL (Baluja 1994), cGA (Harik et al. 1999) and UMDA (Mühlenbein and Paaß 1996), assume that all variables are independent and thus their joint probability can be factorized as a product of univariate marginal probabilities. The probabilistic model in this case consists of separate nodes containing the probability distribution for each of the problem variables. While some algorithms (e.g., PBIL and UMDA) learn the model from a population of solutions, others (e.g., cGA) update the model using only a few individuals. Consequently, these algorithms are the simplest EDAs and thanks to their simplicity, univariate EDAs are particularly suitable for the theoretical analysis of EDA behavior (González et al. 2002; Zhang 2004).

Table 1 Full name of EDAs and the models they use. Discrete EDAs are shown on a *white*, continuous on a *green* and mixed discrete-continuous on a *blue background*, respectively (Color table online)

Algorithm	Complete name	Model used
Univariate		
PBIL	Population Based Incremental Learning	–
UMDA	Univariate Marginal Distribution Algorithm	–
PBIL_C	Continuous PBIL [redacted]	[redacted]
cGA	Compact Genetic Algorithm	–
UMDA_C	Continuous UMDA	–
Bivariate		
HEDA	Histogram-based EDA [redacted]	Marginal Histograms [redacted]
MIMIC	Mutual Information Maximizing Input Clustering	Chain
COMIT	Combining Optimizers with Mutual Information Trees	Tree
BMDA	Bivariate Marginal Distribution Algorithm	Forest
MIMIC_C	Continuous MIMIC [redacted]	Chain [redacted]
CEDA	Copula-based EDA	Copula Functions
Multivariate		
FDA	Factorized Distribution Algorithm	Factor Graph
EBNA	Estimation of Bayesian Network Algorithm	Bayesian Network
BOA	Bayesian Optimization Algorithm	Bayesian Network
EGNA	Estimation of Gaussian Network Algorithm [redacted]	Gaussian Bayesian Network
IDEA	Iterated Density Estimation Evolutionary Algorithm	Gaussian Markov Network
EMNA	Estimation of Multivariate Normal distribution Algorithm	Gaussian Markov Network
MBOA	Mixed BOA [redacted]	Decision Graphs [redacted]
MN-FDA	Markov Network based FDA	Markov Network
[redacted]OPEDA	MultiObjective Parzan-based EDA [redacted]	Mixture of Kernels
rBOA	Real-coded BOA	Gaussian Bayesian Network
EBCOA	Evolutionary Bayesian Classifier based Optimization Algorithm	Bayesian Network Classifiers
MN-EDA	Markov Network EDA	Markov Network
UEBNA	Unsupervised EBNA	Bayesian Network
EcGA	Extended cGA	Marginal Product Model
BGMMEDA	Boosting Gaussian Mixture Model based EDA	Mixture of Gaussian Markov Networks
DEUM	Distribution Estimation Using Markov Random Fields	Markov Network
[redacted]MA-ES	Covariance Matrix Adaptation Evolutionary Strategy	Gaussian Markov Network
MARLEDA	Markovian Learning EDA	Markov Network
EDNA	Estimation of Dependency Network Algorithm	Dependency Network
RM-MEDA	Regulatory Model-based Multiobjective EDA [redacted]	Mixture of hyperplanes [redacted]
KEDA	Kernel density-based EDA	Mixture of Gaussian Kernels
AffEDA	Affinity propagation EDA	Marginal Product Model
[redacted]	Model Building Growing Neural Gas [redacted]	Mixture of Gaussian Markov Networks
LTGA	Linkage Tree GA	Hierarchical Dependency Tree
JGBN-EDA	Joint GBN-based EDA [redacted]	Gaussian Bayesian Network

To extend the modeling capability of EDAs, *bivariate models* were used in EDAs. Bivariate models can represent pairwise dependencies between variables using efficient learning methods. MIMIC (De Bonet et al. 1997) uses a chain structured probabilistic model where the probability distribution of all the variables except the head node is conditioned on the value of the variable preceding them in the chain. The structure of the probabilistic model in COMIT (Baluja and Davies 1997) is a tree, while it is generalized to a forest of trees (dependency graph) in BMDA (Pelikan and Mühlenbein 1999).

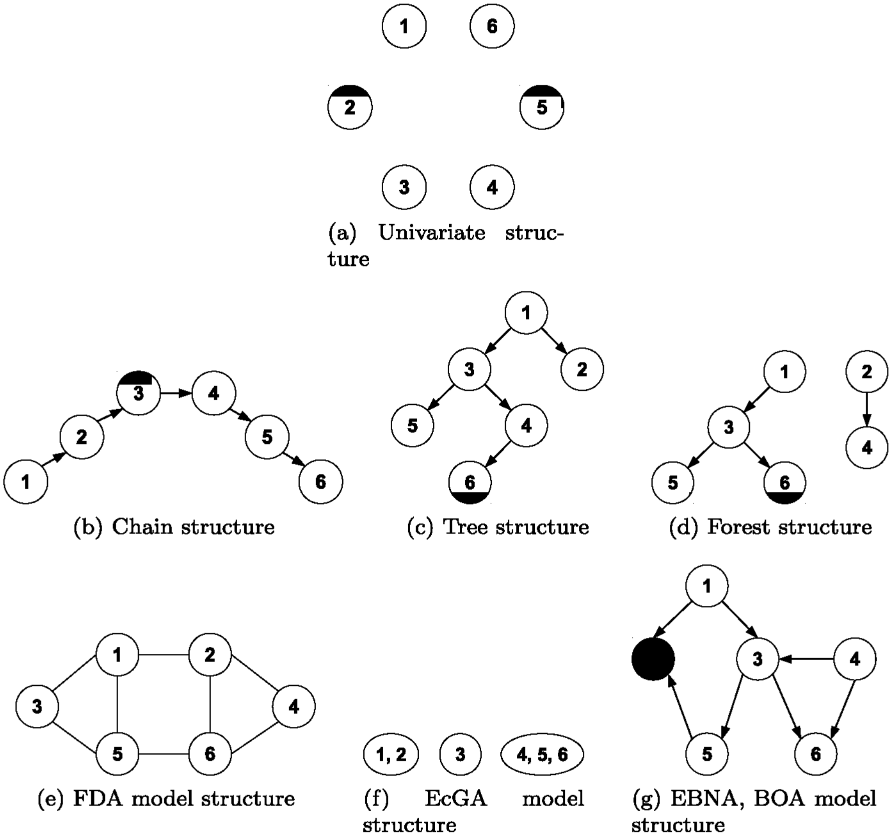
In univariate and bivariate EDAs, the probabilistic model structure is either fixed or is very restricted. Therefore, while they can be efficiently applied to separable problems (without any dependency) or to problems with low degrees of dependency among the variables, they might still rapidly lose their efficiency when applied to more complicated problems, with larger number of variable interactions. A further attempt to improve EDAs is to use models that can capture dependencies between an arbitrary number of variables. Thus the joint probability distribution can be decomposed into factors involving several variables of the problem. Of course, this more flexible modeling by *multivariate EDAs*, capable of learning complex structures, comes at the cost of a greater computational effort. Figure 4 shows some examples of possible model structures learnt by EDAs.

3.1.1 Multivariate EDAs

FDA (Mühlenbein and Mahnig 1999; Mühlenbein et al. 1999) gives a factorization of the joint probability distribution for a class of problems known as additively decomposable functions. EcGA (Harik et al. 2006) factorizes the joint probability distribution into a number of marginal distributions defined over non-overlapping subsets of variables in a probabilistic model called marginal product model. An MDL scoring metric is used to search for the proper partitioning of the variables.

EBNA (Etxeberria and Larrañaga 1999) and BOA (Pelikan et al. 1999) learn a Bayesian network from the selected set of solutions in every generation. While both of the algorithms use a greedy local search method to explore the space of possible network structures, EBNA measures the quality of the networks using the BIC metric and BOA utilizes the BDe metric to score them. BOA is also further extended to hierarchical BOA (Pelikan 2005) by incorporating diversity-preserving techniques and an improved representation for Bayesian network parameters with decision graphs. An improved version of FDA, known as learning FDA, is also proposed that uses Bayesian networks to dynamically learn the interdependent variables (Mühlenbein and Mahnig 1999). Thanks to the powerful probabilistic model that these algorithms use, they can be applied to solve many difficult problems (Larrañaga et al. 2000a; Pelikan and Hartmann 2006).

Because of model learning complexity, Markov network-based EDAs (Santana 2003; Wang and Wang 2004; Shakya 2006; Alden 2007) are usually applied to applications where the structure of the optimization problem is known and can be easily represented using an undirected graphical model. However, an approximation of the probability distribution, like Kikuchi approximations (Santana 2005), can also be estimated to obtain the factorization of problem variables. The use of this type of



(a) Univariate structure

(b) Chain structure

(c) Tree structure

(d) Forest structure

(e) FDA model structure

(f) EcGA model structure

(g) EBNA, BOA model structure

Fig. 4 Examples of different types of model structures used in EDAs

probabilistic models for optimization is still topic of active research and very recently Shakya and Santana (2012) have reviewed new developments in this type of EDAs.

EDNA (Gámez et al. 2007) uses dependency networks (Heckerman et al. 2001) to model the problem structure. Based on a heuristic approximation, this algorithm uses second-order statistics (similar to bivariate EDAs) for model learning. Dependency networks can represent cyclic dependencies between variables which cannot be encoded in Bayesian networks. However, this property prevents the application of sampling techniques used for Bayesian networks. Like Markov network-based EDAs, therefore, EDNA uses relatively complex Gibbs sampling (Geman and Geman 1984) procedures to generate new solutions from the probabilistic model.

3.1.2 Other modeling types

Apart from basic probabilistic models, EDAs have also used other modeling techniques. *Mixture of models* increases the flexibility of joint probability estimation in EDAs, especially for solving multi-modal optimization problems (containing several optima). Pelikan and Goldberg (2000) studied the effect of clustering the set of selected solutions on the performance of UMDA. UEBNA (Peña et al. 2005) represents the mixture with a Bayesian network which is learnt by the structural expectation maximization algorithm (Dempster et al. 1977). Santana et al. (2009b) discussed several topics concerning model building in discrete EDAs.

A promising technique is to take into account the fitness information of individuals in the modeling process. EBCOA (Miquélez et al. 2004) adopts this idea, first introduced by Michalski (2000), by dividing the set of selected individuals into different classes according to their fitness values. The probabilistic model of this algorithm can be any of the Bayesian network classifiers (naïve Bayes, semi-naïve Bayes, tree augmented naïve Bayes) with the class node corresponding to the different classes of fitness values (after discretization). Another related idea is to learn EDA probability distributions from both low and high fitness individuals (Hong et al. 2009). Valdez-Peña et al. (2009) have extended this idea by estimating a distribution for selection operators and used it to identify which solutions would be used for model learning.

Probabilistic models in EDAs can also be used to estimate or predict the fitness values of the new solutions. This approach can be especially useful for problems with a very difficult and time-consuming fitness function. Fitness inheritance modeling has been incorporated into EcGA (Sastry et al. 2004) and BOA (Pelikan and Sastry 2004) to estimate the fitness value of new individuals and, consequently, reduce the total number of function evaluations consumed by the algorithm to reach the optimum. Brownlee et al. (2008) and Brownlee (2009) used a Markov network-based fitness model in DEUM (Shakya 2006) to predict the fitness value of the new individuals and also compare their correlations to the true fitness values.

Propagation methods, used for inference in PGMs, also have applications in EDAs. Mendiburu et al. (2007) used total abductive inference to find the most probable Bayesian network configuration in EBNA as one of the solutions generated from the probabilistic model in order to improve the EDA sampling procedure. Lima et al. (2009) used loopy belief propagation in a local search for BOA as a way to compute optimal local configurations of the problem. An analysis of EBNA performance at different stages of evolution is also given by computing the most probable configuration at each generation (Echegoyen et al. 2009). AffEDA (Santana et al. 2010) uses affinity propagation (Frey and Dueck 2006), another probabilistic-modeling-inspired propagation algorithm, to obtain non-overlapping factorizations of the joint probability distribution.

Otherwise, there have been attempts at combining EDAs with other optimization algorithms, like differential evolution (Sun et al. 2005) and artificial immune systems (de Castro and Zuben 2009), to achieve better optimization performance. Recently, Bengoetxea and Larrañaga (2010) and Ahn et al. (2012) have proposed very similar frameworks for combining EDAs with particle swarm optimization, in continuous and discrete domains respectively. The central idea of these techniques is to combine the global search ability of EDAs with better local exploitation of other methods. LTGA (Thierens 2011) takes advantage of structure learning in EDAs and genetic recombination in GAs by building a linkage tree using an agglomerative hierarchical clustering algorithm based on mutual information as a distance metric. Using this linkage tree the algorithm determines the crossing point of the parents when applying the crossover operator to the selected solutions for generating new solutions.

3.2 Continuous EDAs

The usual choice, adopted by most EAs in continuous domain optimization, is to assume a Gaussian distribution for problem variables. Many of the early continuous EDAs as well as their recent improvements are also based on this assumption. PBIL$_{\mathrm{C}}$ (Sebag and Ducoulombier 1998) extends its discrete version to continuous domains by updating a vector of independent Gaussian distributions. UMDA$_{\mathrm{C}}$ (Larrañaga et al. 1999, 2000b) uses maximum likelihood estimation to learn the parameters of the Gaussian distribution for each variable from the population of solutions. MIMIC$_{\mathrm{C}}$ (Larrañaga et al. 1999, 2000b) learns the chain structured probabilistic model for continuous variables by adapting the concept of (conditional) entropy for univariate and bivariate Gaussian distributions.

EGNA (Larrañaga et al. 1999, 2000b; Larrañaga and Lozano 2001) can be considered as the continuous version of EBNA based on a GBN. Two approaches have been proposed for learning the network structure in this algorithm: (i) starting from a complete DAG, likelihood ratio hypothesis tests are used to decide whether the edge between two nodes should be excluded from the network; (ii) performing a greedy local search in the space of possible DAGs using a scoring metric like BGe (continuous version of the BDe metric) or BIC. EBCOA has also been extended to continuous domains (Miquélez et al. 2006) by building Bayesian classifiers that assume Gaussian distributions for the variables given the class variable value. Karshenas et al. (2011) proposed learning a joint GBN consisting of both variables and objectives in their JGBN-EDA for multi-objective optimization.

IDEA (Bosman and Thierens 2000b; Bosman and Thierens 2000a) and EMNA (Larrañaga and Lozano 2001) learn a full multivariate normal distribution (MND) from the set of selected solutions. The inverse covariance matrix or precision matrix of this distribution corresponds to a type of Markov network depicting pairwise (in)dependencies between variables. While EMNA uses maximum likelihood estimation, IDEA employs Kullback-Leibler divergence in conjunction with a greedy search algorithm, as well as likelihood ratio statistical hypothesis tests. Further improvements of IDEA have been proposed by scaling the diminishing variances (Grahl et al. 2006) and shifting the distribution mean (Bosman and Grahl 2008; Bosman et al. 2008).

3.2.1 Mixture of distributions

An extended version of IDEA (Bosman and Thierens 2001) uses a mixture of normal distributions over clusters of solutions, obtained by applying a clustering algorithm before learning mixture components. rBOA (Ahn et al. 2004) first learns a GBN to obtain a decomposition of the problem variables into smaller subproblems. Then, a separate mixture of GBNs is learnt for each of the subproblems by clustering the solutions in that subproblem. In BGMMEDA (Li et al. 2006), instead of clustering the samples, a boosting technique is applied to estimate a Gaussian mixture model.

MB-GNG (Martí et al. 2011) adopts growing neural gas, a specific single-layer neural network, to determine the location of the components of the mixture of Gaussian distributions. This model learning algorithm is sensitive to, and therefore does not neglect, outliers and is able to automatically adapt its topology while decreasing the accumulated error of the network nodes. The multi-model EDA framework (Weise et al. 2011) extends these mixture methods by applying traditional EA recombination operators to the individual models learnt for each of the clusters in order to improve search space exploration.

RM-MEDA (Zhang et al. 2008) learns a piece-wise continuous manifold for multi-objective optimization using the local principle component analysis algorithm. Each model component consists of a hyper-rectangle with a Gaussian noise.

3.2.2 Other modeling approaches

Pošík (2008, 2009a) proposed the use of *Cauchy* distribution for the purpose of preventing premature convergence. Since the moments of an n-dimensional variable with multivariate Cauchy distribution are not defined, the mean vector and covariance matrix of a Gaussian distribution are computed instead. For sampling new solutions, the scaling factor of the Cauchy distribution is used to obtain isotropically distributed new solutions.

More recently some EDAs have employed *copula* theory to relax the Gaussian assumption for the variables. Copula-based EDAs (CEDAs) (Salinas-Gutiérrez et al. 2009; Wang et al. 2009; Wang and Zeng 2010; Cuesta-Infante et al. 2010) use the copula function for estimating the joint probability distribution of the variables according to Sklar's theorem. The copula function only uses the marginal univariate probabilities to compute the joint probability distribution. This reduces the computational complexity of model learning. Two-dimensional elliptical copulas as well as Archimedean and empirical copulas and their extensions to higher dimensions are studied in the literature. These copula functions will serve as the problem dependency structure when sampling new solutions from the learnt model. In each generation the algorithm selects or constructs a copula function after estimating the univariate marginal distributions and then, generates new samples according to the copula distribution.

CMA-ES (Hansen 2006) incorporates model estimation into evolutionary strategies which mainly deal with continuous domain optimization. The algorithm learns an MND as its probabilistic model to generate new solutions. The probabilistic model estimated in each generation is a combination of information collected over several generations, taking into account the path that the optimizer has traversed in the search space. Instead of estimating a new probabilistic model in each generation, the algorithm *adapts* the model during evolution. Thus, the algorithm is able to use smaller population sizes for optimization by spanning model learning over several generations. Because of such an adaptation strategy, some researchers do not completely consider this algorithm as an EDA (Pošík 2009b). It is worth to note that similar techniques have been proposed for improving the efficiency of EDAs in optimization (Pelikan et al. 2008; Bosman et al. 2008).

3.2.3 Non-parametric probabilistic models

Other probabilistic models that estimate a non-parametric distribution for the variables have also been used in continuous EDAs. IDEA, for example, has employed other models, apart from MND, like normal kernel distribution (a Gaussian kernel for each sample) or histograms in its framework (Bosman and Thierens 2000a, 2000b).

Cho and Zhang (2002) proposed a continuous EDA that learns a mixture of factor analyzers using the EM algorithm. They also employed a more complicated mixture of variational Bayesian independent component analyzers in a later study (Cho and Zhang 2004). MOPEDA (Costa and Minisci 2003) applies a Parzen estimator that convolves the empirical estimation obtained from a finite data set with a squared integrable kernel function in order to reduce the variance of the probability distribution estimation. Both Gaussian and Cauchy kernels are used alternatively during evolution to utilize their intrinsic complementary characteristics. In KEDA (Luo and Qian 2009), the width of each kernel is dynamically computed during the optimization.

Histogram-based EDAs (HEDAs) discretize each variable's values by dividing their range to a number of bins. Tsutsui et al. (2001) proposed two types of marginal histogram models: (i) a fixed-width histogram (FWH) where the domain of each variable is divided into a fixed number of bins whose height may differ depending on the variable values; (ii) a fixed-height histogram (FHH) where all bins have an equal value generation probability but can have different widths. Consequently, there will be more bins in denser regions and thus modeling will be more accurate.

Ding et al. (2008) proposed two improvements to this histogram modeling in their HEDA. They introduced a surrounding effect, where the values of each bin can affect the values of its surrounding bins using a special surrounding factor. They also employed a shrinkage strategy whereby the height of the bin containing the best value of the variable can exceed a predefined threshold. PBIL_C is also extended with histograms (Xiao et al. 2009), combining the original updating rule with bin updating, where the bins reaching a predefined height are divided.

Histogram modeling has also been applied to optimization in permutation domains (Tsutsui 2002; Tsutsui et al. 2006) using two different types of models. The first is an edge histogram matrix where each entry indicates the frequency of two permutation values occurring adjacent to each other in the population. The second is a node histogram matrix that encodes the frequency at which a special value in the permutation occurs at a specific location in the solution. Specific sampling algorithms are developed for these models where a new value is generated according to the value of adjacent permutation locations or the position for which the value is going to be generated.

3.3 Discrete-continuous EDAs

MBOA (Očenášek and Schwarz 2002; Očenášek et al. 2004) adopts binary classification and regression decision trees to solve mixed discrete-continuous optimization problems. The algorithm uses a BDe-like scoring metric to build a decision tree for each variable to encode its related probability distribution. The decision trees allow the algorithm to build individual models (like Gaussian kernels) for specific regions of the search space, stored in different tree leaves.

3.4 Discussion

Table 1 gives a summary of the presented EDAs and their probabilistic models. The algorithms are divided into three different classes according to the complexity of their probabilistic models: univariate, bivariate and multivariate EDAs. Within each class, the algorithms are ordered chronologically to show how the use of probabilistic models in EDAs has evolved during time.

Initial EDAs mainly considered about the probability distribution of individual variables, in order to perform a more effective search for the solutions of separable problems. In this kind of problems the optimal value of each variable can be obtained regardless of the value of the other variables. In other terms, the way that the value of each variable influences the fitness of the whole solution is not affected by the values of other variables at all.

The limitations of univariate EDAs, brought up the need for a more advanced probabilistic modeling. The main advantage of these new probabilistic models is that they consider a kind of structure for the problem, reflecting an estimation of the interactions between variables. Some algorithms put certain constraints on the structures to be considered, e.g. bivariate EDAs can only consider mutual interactions. Some others like FDA consider a fixed structure given beforehand (e.g. for a specific class of problems), and try to find the best parameter estimation that fits this structure. But most of EDAs try to learn the structure dynamically during evolution. A number of algorithms require the variables to be clustered into completely disjoint dependence groups (e.g. EcGA and AffEDA), whereas in others overlapping groups of dependent variables is allowed (e.g. MIMIC and EBNA).

Because of their ability to represent complex patterns of interactions between the problem variables, the use of multivariate probabilistic models has become dominant in EDAs. Usually these algorithms perform a kind of *structure learning* to estimate the probabilistic model, which is very time-consuming in comparison to other parts of the algorithm. Therefore, in practice an upper bound is imposed on the order of interactions that is considered in the structure learning. This restriction can also be imposed implicitly, e.g. using penalized scoring metrics as discussed in Sect. 2.3 for learning Bayesian networks.

The choice of the type of EDA to be used, depends very much on the problem. If the problem at hand is linear, or the variables are not believed to be strongly dependent, then one should use univariate EDAs since they are computationally more efficient. On the contrary, if we are dealing with a problem that has high order of interactions between its variables, then EDAs that use probabilistic models with higher representational capability should be used in order to be able to reach the optimal solution(s) of the problem. The structures estimated by these EDAs can also give a better understanding of unknown problems. Several works have studied the accuracy of these structures and the information we can obtain from them (Lima et al. 2007; Karshenas et al. 2009; Santana et al. 2009a).

In reality, one should compromise between the computational complexity and the optimization capability of these algorithms when applying them to different problems. Based on this observation, there has been many efforts to increase the efficiency of EDAs while keeping their complexity at an acceptable range. Techniques like parallelization and hybridization are introduced in the literature which are usually referred to as efficiency enhancement techniques (Pelikan 2005).

3.5 Model-based genetic programming

Although probabilistic models were first built into genetic algorithms, the idea was soon adopted also in genetic programming (GP). In GP the objective is to evolve functions or computer programs that are able to solve a given problem. The usual representation used to encode the solutions are tree structures. The variation operators (crossover and mutation) are adapted to work with this representation. Because of this complex representation, model learning and sampling can be a challenging task. However, several GPs based on probabilistic modeling have been proposed in the literature.

Probabilistic incremental program evolution (PIPE) (Sałustowicz and Schmidhuber 1997) is a GP algorithm based on univariate factorization of program distribution. A probabilistic prototype tree (PPT) model encodes the probability distribution and is later used to generate new program trees at each generation of the algorithm. Extended compact genetic programming (ECGP) (Sastry and Goldberg 2003) incorporates the use of marginal product distributions into the context of GP. Also based on a PPT model, ECGP constructs a factorization of the tree program distribution equal to the product of marginal distributions. Each marginal distribution is associated with a subtree of the PPT. The structure of the factorization is learned using a greedy algorithm, similar to EcGA.

The use of Bayesian networks for GP was proposed by Yanai and Iba (2003). This estimation of distribution programming approach is based on the use of the PPT. The conditional probabilities between the nodes of the PPT are computed for the purpose of representing a wider class of probability distributions than PIPE and ECGP. Recently, Hasegawa and Iba (2008) proposed a Bayesian network modeling approach for GP that significantly reduces the size of the conditional probability tables. The algorithm also requires fewer samples to construct the Bayesian network from the selected solutions.

Another type of GP-EDAs are based on the use of grammars. These algorithms (Shan et al. 2006; McKay et al. 2010; Bosman and de Jong 2008) depart from the traditional uses of probabilistic graphical models since the probability distributions are often associated with the grammar rules and their different contexts of application. For example, Bosman and de Jong (2008) estimate the distribution of programming trees based on the subtrees that actually occur in the data. The representation specifies a set of rules whose expansion leads to trees, and the probability distributions are defined on these rules. For a good review of these algorithms, see Shan et al. (2006) and McKay et al. (2010).

4 Conclusions

Probabilistic graphical models are a useful and effective way of dealing with uncertainty in data. They have been studied at length over the last three decades, and many methods have been proposed to automate their learning and inference. They have also been successfully used in machine learning tasks. Special-purpose versions of these probabilistic tools have been proposed for dealing with continuous variables and domains with mixed discrete-continuous variables.

One of the disciplines that has greatly taken advantage of probabilistic modeling is evolutionary computation, resulting in a new paradigm, namely, estimation of distribution algorithms. Although this is a relatively new paradigm, numerous studies have investigated its different aspects, and several types of algorithms have been proposed based on this paradigm. These algorithms cover both discrete and continuous domains, and within each domain probabilistic models with different complexities have been used in these algorithms.

EDAs are still topic of intensive research, and every year many new works related to the theory or application of these algorithms are published. New studies are trying to extend the application of these algorithms to other domains like multi-objective, noisy or dynamic problems. Nevertheless, because of the close relationship that these algorithms have with probabilistic modeling, any new development in the learning or inference of probabilistic models can help to achieve competent problem optimization with EDAs.

References

Ahn, C.W., An, J., Yoo, J.C.: Estimation of particle swarm distribution algorithms: combining the benefits of PSO and EDAs. Inf. Sci. **192**, 109–119 (2012)

Ahn, C., Ramakrishna, R., Goldberg, D.: Real-coded Bayesian optimization algorithm: bringing the strength of BOA into the continuous world. In: 6th Annual Conference on Genetic and Evolutionary Computation (GECCO'04), pp. 840–851. Springer, Berlin (2004)

Akaike, H.: A new look at the statistical model identification. IEEE Trans. Autom. Control **19**(6), 716–723 (1974)

Alden, M.E.: MARLEDA: effective distribution estimation through Markov random fields. Ph.D. Thesis, The University of Texas at Austin (2007)

Baluja, S.: Population-based incremental learning: a method for integrating genetic search based function optimization and competitive learning. Tech. Rep. CMU-CS-94-163, Carnegie-Mellon University (1994)

Baluja, S., Davies, S.: Using optimal dependency-trees for combinational optimization. In: 14th International Conference on Machine Learning, pp. 30–38. Morgan Kaufmann, San Mateo (1997)

Bengoetxea, E., Larrañaga, P.: EDA-PSO: a hybrid paradigm combining estimation of distribution algorithms and particle swarm optimization. In: Swarm Intelligence. Lecture Notes in Computer Science, vol. 6234, pp. 416–423. Springer, Berlin (2010)

Bosman, P.A.N., Grahl, J.: Matching inductive search bias and problem structure in continuous estimation of distribution algorithms. Eur. J. Oper. Res. **185**(3), 1246–1264 (2008)

Bosman, P.A.N., Grahl, J., Thierens, D.: Enhancing the performance of maximum-likelihood Gaussian EDAs using anticipated mean shift. In: 10th International Conference on Parallel Problem Solving from Nature (PPSN X), pp. 133–143. Springer, Berlin (2008)

Bosman, P.A.N., Thierens, D.: Advancing continuous IDEAs with mixture distributions and factorization selection metrics. In: Optimization by building and using probabilistic models (OBUPM) Workshop at the Genetic and Evolutionary Computation Conference (GECCO'01), pp. 208–212. ACM, New York (2001)

Bosman, P.A.N., de Jong, E.: Adaptation of a success story in GAs: Estimation-of-distribution algorithms for tree-based optimization problems. In: Success in Evolutionary Computation. Studies in Computational Intelligence, vol. 92, pp. 3–18. Springer, Berlin (2008)

Bosman, P.A.N., Thierens, D.: Linkage information processing in distribution estimation algorithms. In: Genetic and Evolutionary Computation Conference (GECCO'99), pp. 60–67. Morgan Kaufmann, San Mateo (1999)

Bosman, P.A.N., Thierens, D.: Continuous iterated density estimation evolutionary algorithms within the IDEA framework. In: Genetic and Evolutionary Computation Conference (GECCO'00) Workshop, pp. 197–200 (2000a)

Bosman, P.A.N., Thierens, D.: Expanding from discrete to continuous estimation of distribution algorithms: the IDEA. In: 6th International Conference on Parallel Problem Solving from Nature (PPSN VI), pp. 767–776. Springer, Berlin (2000b)

Bouckaert, R.R.: Bayesian belief networks: from construction to inference. Ph.D. Thesis, Universiteit Utrecht, Faculteit Wiskunde en Informatica (1995)

Brownlee, A., McCall, J., Zhang, Q., Brown, D.: Approaches to selection and their effect on fitness modelling in an estimation of distribution algorithm. In: IEEE Congress on Evolutionary Computation (CEC 2008)—IEEE World Congress on Computational Intelligence, pp. 2621–2628. IEEE Comput. Soc., Los Alamitos (2008)

Brownlee, A.E.I.: Multivariate Markov networks for fitness modelling in an estimation of distribution algorithm. Ph.D. Thesis, The Robert Gordon University. School of Computing (2009)

Buntine, W.: Theory refinement on Bayesian networks. In: 7th Conference on Uncertainty in Artificial Intelligence (UAI'91), vol. 91, pp. 52–60. Morgan Kaufmann, San Mateo (1991)

Chickering, D.: Learning Bayesian networks is NP-complete. In: Learning from Data: Artificial Intelligence and Statistics V. Lecture Notes in Statistics, vol. 112, pp. 121–130. Springer, Berlin (1996)

Chickering, D., Geiger, D., Heckerman, D.: Learning Bayesian networks is NP-hard. Tech. Rep. MSR-TR-94-17, Microsoft Research (1994)

Chickering, D., Heckerman, D., Meek, C.: Large-sample learning of Bayesian networks is NP-hard. J. Mach. Learn. Res. **5**, 1287–1330 (2004)

Cho, D.Y., Zhang, B.T.: Evolutionary optimization by distribution estimation with mixtures of factor analyzers. In: IEEE Congress on Evolutionary Computation (CEC'02), vol. 2, pp. 1396–1401. IEEE Comput. Soc., Los Alamitos (2002)

Cho, D.Y., Zhang, B.T.: Evolutionary continuous optimization by distribution estimation with variational Bayesian independent component analyzers mixture model. In: Parallel Problem Solving from Nature (PPSN VIII). Lecture Notes in Computer Science, vol. 3242, pp. 212–221. Springer, Berlin (2004)

Cooper, G., Herskovits, E.: A Bayesian method for the induction of probabilistic networks from data. Mach. Learn. **9**(4), 309–347 (1992)

Costa, M., Minisci, E.: MOPED: a multi-objective Parzen-based estimation of distribution algorithm for continuous problems. In: Evolutionary Multi-Criterion Optimization. Lecture Notes in Computer Science, vol. 2632, p. 71. Springer, Berlin (2003)

Cramer, N.L.: A representation for the adaptive generation of simple sequential programs. In: First International Conference on Genetic Algorithms, pp. 183–187. Erlbaum, Hillsdale (1985)

Cuesta-Infante, A., Santana, R., Hidalgo, J.I., Bielza, C., Larrañaga, P.: Bivariate empirical and n-variate Archimedean copulas in estimation of distribution algorithms. In: IEEE Congress on Evolutionary Computation (CEC'10) (2010)

Dawid, A.P.: Conditional independence in statistical theory. J. R. Stat. Soc. B **41**(1), 1–31 (1979)

De Bonet, J., Isbell, C., Viola, P.M.: Finding optima by estimating probability densities. Adv. Neural Inf. Process. Syst. **9**, 424–430 (1997)

de Castro, P.A.D., Zuben, F.J.V.: BAIS: a Bayesian artificial immune system for the effective handling of building blocks. Inf. Sci. **179**(10), 1426–1440 (2009)

Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. J. R. Stat. Soc. B **39**(1), 1–38 (1977)

Ding, N., Zhou, S., Sun, Z.: Histogram-based estimation of distribution algorithm: a competent method for continuous optimization. J. Comput. Sci. Technol. **23**(1), 35–43 (2008)

Echegoyen, C., Mendiburu, A., Santana, R., Lozano, J.: Analyzing the probability of the optimum in EDAs based on Bayesian networks. In: IEEE Congress on Evolutionary Computation (CEC'09), pp. 1652–1659 (2009)

Etxeberria, R., Larrañaga, P.: Global optimization using Bayesian networks. In: Second Symposium on Artificial Intelligence (CIMAF-99), pp. 332–339 (1999)

Fogel, L.J.: Artificial Intelligence Through Simulated Evolution. Wiley, New York (1966)

Frey, B.J., Dueck, D.: Mixture modeling by affinity propagation. In: Advances in Neural Information Processing Systems, vol. 18, pp. 379–386. MIT Press, Cambridge (2006)

Frydenberg, M.: The chain graph Markov property. Scand. J. Stat. **17**(4), 333–353 (1990)

Gámez, J., Mateo, J., Puerta, J.E.: Estimation of dependency networks algorithm. In: Bio-inspired Modeling of Cognitive Tasks. Lecture Notes in Computer Science, vol. 4527, pp. 427–436. Springer, Berlin (2007)

Geiger, D., Heckerman, D.: Learning Gaussian networks. In: 10th Conference on Uncertainty in Artificial Intelligence (UAI'94), pp. 235–243 (1994)

Geman, S., Geman, D.: Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. IEEE Trans. Pattern Anal. Mach. Intell. **6**(6), 721–741 (1984)

Goldberg, D.E.: The Design of Innovation: Lessons from and for Competent Genetic Algorithms. Kluwer Academic, Norwell (2002)

González, C., Lozano, J., Larrañaga, P.: Mathematical modelling of UMDAc algorithm with tournament selection. Behaviour on linear and quadratic functions. Int. J. Approx. Reason. **31**(3), 313–340 (2002)

Grahl, J., Bosman, P.A.N., Rothlauf, F.: The correlation-triggered adaptive variance scaling IDEA. In: 8th Annual Conference on Genetic and Evolutionary Computation (GECCO'06), pp. 397–404. ACM, New York (2006)

Grünwald, P.: The minimum description length principle and reasoning under uncertainty. Ph.D. Thesis, University of Amsterdam (1998)

Hansen, N.: The CMA evolution strategy: a comparing review. In: (Lozano et al. 2006), pp. 75–102 (2006)

Harik, G., Cantú-Paz, E., Goldberg, D., Miller, B.: The gambler's ruin problem, genetic algorithms, and the sizing of populations. Evol. Comput. **7**(3), 231–253 (1999)

Harik, G.R., Lobo, F.G., Sastry, K.: Linkage learning via probabilistic modeling in the Extended Compact Genetic Algorithm (ECGA). In: (Pelikan et al. 2006), pp. 39–61 (2006). Chap. 3

Harik, G., Lobo, F., Goldberg, D.: The compact genetic algorithm. IEEE Trans. Evol. Comput. **3**(4), 287–297 (1999)

Hasegawa, Y., Iba, H.: A Bayesian network approach to program generation. IEEE Trans. Evol. Comput. **12**(6), 750–764 (2008)

Heckerman, D., Geiger, D., Chickering, D.: Learning Bayesian networks: the combination of knowledge and statistical data. Mach. Learn. **20**(3), 197–243 (1995)

Heckerman, D., Chickering, D.M., Meek, C., Rounthwaite, R., Kadie, C.: Dependency networks for inference, collaborative filtering, and data visualization. J. Mach. Learn. Res. **1**, 49–75 (2001)

Holland, J.: Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor (1975)

Hong, Y., Zhu, G., Kwong, S., Ren, Q.: Estimation of distribution algorithms making use of both high quality and low quality individuals. In: IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'09), pp. 1806–1813. IEEE Comput. Soc., Los Alamitos (2009)

Karshenas, H., Nikanjam, A., Helmi, B.H., Rahmani, A.T.: Combinatorial effects of local structures and scoring metrics in Bayesian optimization algorithm. In: First ACM/SIGEVO Summit on Genetic and Evolutionary Computation (GEC'09), pp. 263–270. ACM, New York (2009)

Karshenas, H., Santana, R., Bielza, C., Larrañaga, P.: Multi-objective optimization with joint probabilistic modeling of objectives and variables. In: Evolutionary Multi-Criterion Optimization. Lecture Notes in Computer Science, vol. 6576, pp. 298–312. Springer, Berlin (2011)

Koller, D., Friedman, N.: Probabilistic Graphical Models: Principles and Techniques. MIT Press, Cambridge (2009)

Koza, J.R.: Genetic Programming: on the Programming of Computers by Means of Natural Selection. MIT Press, Cambridge (1992)

Larrañaga, P., Etxeberria, R., Lozano, J., Pena, J.: Optimization by learning and simulation of Bayesian and Gaussian networks. Tech. Rep. EHU-KZAAIK-IK-4/99, Intelligent Systems Group, Department of Computer Science and Artificial Intelligence, University of the Basque Country (1999)

Larrañaga, P., Etxeberria, R., Lozano, J., Peña, J.: Combinatonal optimization by learning and simulation of Bayesian networks. In: 16th Conference on Uncertainty in Artificial Intelligence (UAI'00), pp. 343–352. Morgan Kaufmann, San Mateo (2000a)

Larrañaga, P., Etxeberria, R., Lozano, J., Peña, J.: Optimization in continuous domains by learning and simulation of Gaussian networks. In: Conference on Genetic and Evolutionary Computation (GECCO'00) Workshop Program, pp. 201–204. Morgan Kaufmann, San Mateo (2000b)

Larrañaga, P., Lozano, J. (eds.): Estimation of Distribution Algorithms: a New Tool for Evolutionary Computation. Kluwer Academic, Norwell (2001)

Larrañaga, P., Moral, S.: Probabilistic graphical models in artificial intelligence. Appl. Soft Comput. **11**(2), 1511–1528 (2011)

Lauritzen, S.L., Spiegelhalter, D.J.: Local computations with probabilities on graphical structures and their application to expert systems. J. R. Stat. Soc. B **50**(2), 157–224 (1988)

Li, B., Zhong, R.T., Wang, X.J., Zhuang, Z.Q.: Continuous optimization based-on boosting Gaussian mixture model. In: 18th International Conference on Pattern Recognition (ICPR'06), vol. 1, pp. 1192–1195 (2006)

Lima, C., Pelikan, M., Goldberg, D., Lobo, F., Sastry, K., Hauschild, M.: Influence of selection and replacement strategies on linkage learning in BOA. In: CEC 2007, IEEE Congress on Evolutionary Computation, pp. 1083–1090 (2007)

Lima, C., Pelikan, M., Lobo, F., Goldberg, D.: Loopy substructural local search for the Bayesian optimization algorithm. In: Engineering Stochastic Local Search Algorithms. Designing, Implementing and Analyzing Effective Heuristics. Lecture Notes in Computer Science, vol. 5752, pp. 61–75. Springer, Berlin (2009)

Lozano, J., Larrañaga, P., Inza, I., Bengoetxea, E. (eds.): Towards a New Evolutionary Computation: Advances on Estimation of Distribution Algorithms. Studies in Fuzziness and Soft Computing, vol. 192. Springer, Berlin (2006)

Luo, N., Qian, F.: Evolutionary algorithm using kernel density estimation model in continuous domain. In: 7th Asian Control Conference (ASCC'09), pp. 1526–1531 (2009)

Martí, L., García, J., Berlanga, A., Coello, C.A.C., Molina, J.M.: MB-GNG: addressing drawbacks in multi-objective optimization estimation of distribution algorithms. Oper. Res. Lett. **39**(2), 150–154 (2011)

McKay, R., Hoai, N., Whigham, P., Shan, Y., O'Neill, M.: Grammar-based genetic programming: a survey. Genet. Program. Evol. Mach. **11**, 365–396 (2010)

Mendiburu, A., Santana, R., Lozano, J.A.: Introducing belief propagation in estimation of distribution algorithms: a parallel framework. Tech. Rep. EHU-KAT-IK-11-07, Intelligent Systems Group, University of the Basque Country (2007)

Michalski, R.S.: Learnable evolution model: evolutionary processes guided by machine learning. Mach. Learn. **38**, 9–40 (2000)

Miquélez, T., Bengoetxea, E., Larrañaga, P.: Evolutionary computation based on Bayesian classifiers. Int. J. Appl. Math. Comput. Sci. **14**(3), 335–350 (2004)

Miquélez, T., Bengoetxea, E., Larrañaga, P.: Evolutionary Bayesian classifier-based optimization in continuous domains. In: 6th International Conference on Simulated Evolution and Learning (SEAL'06). Lecture Notes in Computer Science, vol. 4247, pp. 529–536. Springer, Berlin (2006)

Mühlenbein, H., Mahnig, T.: FDA–A scalable evolutionary algorithm for the optimization of additively decomposed functions. Evol. Comput. **7**(4), 353–376 (1999)

Mühlenbein, H., Mahnig, T., Ochoa Rodríguez, A.: Schemata, distributions and graphical models in evolutionary optimization. J. Heuristics **5**(2), 215–247 (1999)

Mühlenbein, H., Paaß, G.: From recombination of genes to the estimation of distributions I. Binary parameters. In: 4th International Conference on Parallel Problem Solving from Nature (PPSN IV). Lecture Notes in Computer Science, vol. 1141, pp. 178–187. Springer, Berlin (1996)

Očenášek, J., Schwarz, J.: Estimation distribution algorithm for mixed continuous-discrete optimization problems. In: Intelligent Technologies: Theory and Applications: New Trends in Intelligent Technologies, pp. 227–232. IOS Press, Amsterdam (2002)

Očenášek, J., Kern, S., Hansen, N., Koumoutsakos, P.: A mixed Bayesian optimization algorithm with variance adaptation. In: Parallel Problem Solving from Nature (PPSN VIII). Lecture Notes in Computer Science, vol. 3242, pp. 352–361. Springer, Berlin (2004)

Pearl, J.: Bayesian networks: a model of self-activated memory for evidential reasoning. In: 7th Conference of the Cognitive Science Society, pp. 329–334 (1985)

Pelikan, M., Goldberg, D., Lobo, F.: A survey of optimization by building and using probabilistic models. Comput. Optim. Appl. **21**(1), 5–20 (2002)

Pelikan, M., Mühlenbein, H.: The bivariate marginal distribution algorithm. In: Advances in Soft Computing-Engineering Design and Manufacturing, pp. 521–535 (1999)

Pelikan, M., Sastry, K., Cantú-Paz, E. (eds.): Scalable Optimization via Probabilistic Modeling: from Algorithms to Applications. Springer, Berlin (2006)

Pelikan, M., Sastry, K., Goldberg, D.: Sporadic model building for efficiency enhancement of the hierarchical BOA. Genet. Program. Evol. Mach. **9**(1), 53–84 (2008)

Pelikan, M.: Hierarchical Bayesian Optimization Algorithm: Toward a New Generation of Evolutionary Algorithms, 1st edn. Studies in Fuzziness and Soft Computing, vol. 170. Springer, Berlin (2005)

Pelikan, M., Goldberg, D.: Genetic algorithms, clustering, and the breaking of symmetry. In: Parallel Problem Solving from Nature (PPSN VI). Lecture Notes in Computer Science, vol. 1917, pp. 385–394. Springer, Berlin (2000)

Pelikan, M., Goldberg, D.E., Cantú-Paz, E.B.: The Bayesian optimization algorithm. In: Conference on Genetic and Evolutionary Computation (GECCO'99), vol. 1, pp. 525–532. Morgan Kaufmann, San Mateo (1999)

Pelikan, M., Hartmann, A.: Searching for ground states of Ising spin glasses with hierarchical BOA and cluster exact approximation. In: (Pelikan et al. 2006), pp. 333–349 (2006)

Pelikan, M., Sastry, K.: Fitness inheritance in the Bayesian optimization algorithm. In: Conference on Genetic and Evolutionary Computation (GECCO'04). Lecture Notes in Computer Science, vol. 3103, pp. 48–59. Springer, Berlin (2004)

Peña, J.M., Lozano, J.A., Larrañaga, P.: Globally multimodal problem optimization via an estimation of distribution algorithm based on unsupervised learning of Bayesian networks. Evol. Comput. **13**(1), 43–66 (2005)

Pošík, P.: Preventing premature convergence in a simple EDA via global step size setting. In: 10th International Conference on Parallel Problem Solving from Nature (PPSN X). Lecture Notes in Computer Science, vol. 5199, pp. 549–558. Springer, Berlin (2008)

Pošík, P.: BBOB-benchmarking a simple estimation of distribution algorithm with cauchy distribution. In: 11th Annual Conference Companion on Genetic and Evolutionary Computation (GECCO'09), pp. 2309–2314. ACM, New York (2009a)

Pošík, P.: Stochastic local search techniques with unimodal continuous distributions: a survey. In: EvoWorkshops on Applications of Evolutionary Computing (EvoWorkshops'09), pp. 685–694. Springer, Berlin (2009b)

Rechenberg, I.: Evolutionsstrategie-Optimierung Technischer Systeme nach Prinzipien der Biologischen Evolution. Ph.D. Thesis, reprinted by Fromman-Holzboog (1973)

Rissanen, J.: Modeling by shortest data description. Automatica **14**(5), 465–471 (1978)

Robinson, R.: Counting unlabeled acyclic digraphs. In: Combinatorial Mathematics V. Lecture Notes in Mathematics, vol. 622, pp. 28–43. Springer, Berlin (1977)

Salinas-Gutiérrez, R., Hernández-Aguirre, A., Villa-Diharce, E.: Using copulas in estimation of distribution algorithms. In: Advances in Artificial Intelligence (MICAI'09). Lecture Notes in Computer Science, vol. 5845, pp. 658–668. Springer, Berlin (2009)

Sałustowicz, R.P., Schmidhuber, J.: Probabilistic incremental program evolution: stochastic search through program space. In: 9th European Conference on Machine Learning (ECML'97). Lecture Notes in Computer Science, vol. 1224, pp. 213–220. Springer, Berlin (1997)

Santana, R., Bielza, C., Lozano, J., Larrañaga, P.: Mining probabilistic models learned by EDAs in the optimization of multi-objective problems. In: 11th Annual Conference on Genetic and Evolutionary Computation (GECCO'09), pp. 445–452. ACM, New York (2009a)

Santana, R., Larrañaga, P., Lozano, J.: Research topics in discrete estimation of distribution algorithms based on factorizations. Memet. Comput. **1**(1), 35–54 (2009b)

Santana, R., Larrañaga, P., Lozano, J.: Learning factorizations in estimation of distribution algorithms using affinity propagation. Evol. Comput. **18**(4), 515–546 (2010)

Santana, R.: A Markov network based factorized distribution algorithm for optimization. In: 14th European Conference on Machine Learning (ECML'03). Lecture Notes in Computer Science, vol. 2837, pp. 337–348. Springer, Berlin (2003)

Santana, R.: Estimation of distribution algorithms with Kikuchi approximations. Evol. Comput. **13**, 67–97 (2005)

Santana, R.: Estimation of distribution algorithms: from available implementations to potential developments. In: 13th Annual Conference Companion on Genetic and Evolutionary Computation (GECCO'11), pp. 679–686. ACM, New York (2011)

Santana, R., Bielza, C., Larrañaga, P., Lozano, J.A., Echegoyen, C., Mendiburu, A., Armañanzas, R., Shakya, S.: Mateda-2.0: estimation of distribution algorithms in MATLAB. J. Stat. Softw. **35**(7), 1–30 (2010)

Sastry, K., Goldberg, D.E.: Probabilistic model building and competent genetic programming. In: Genetic Programming Theory and Practice, pp. 205–220. Kluwer Academic, Norwell (2003). Chap. 13

Sastry, K., Pelikan, M., Goldberg, D.: Efficiency enhancement of genetic algorithms via building-block-wise fitness estimation. In: IEEE Congress on Evolutionary Computation (CEC'04), vol. 1, pp. 720–727 (2004)

Schwarz, G.: Estimating the dimension of a model. Ann. Stat. **6**(2), 461–464 (1978)

Sebag, M., Ducoulombier, A.: Extending population-based incremental learning to continuous search spaces. In: 5th International Conference on Parallel Problem Solving from Nature (PPSN V). Lecture Notes in Computer Science, vol. 1498, pp. 418–427. Springer, Berlin (1998)

Shakya, S.: DEUM: a framework for an estimation of distribution algorithm based on Markov random fields. Ph.D. Thesis, The Robert Gordon University (2006)

Shakya, S., Santana, R. (eds.): Markov Networks in Evolutionary Computation. Adaptation, Learning, and Optimization, vol. 14. Springer, Berlin (2012)

Shan, Y., McKay, R., Essam, D., Abbass, H.: a survey of probabilistic model building genetic programming. In: (Pelikan et al. 2006), pp. 121–160 (2006)

Spirtes, P., Glymour, C.: An algorithm for fast recovery of sparse causal graphs. Soc. Sci. Comput. Rev. **9**(1), 62–72 (1991)

Spirtes, P., Glymour, C., Scheines, R.: Causation, Prediction, and Search, 2nd edn. MIT Press, Cambridge (2001)

Sun, J., Zhang, Q., Tsang, E.: DE/EDA: a new evolutionary algorithm for global optimization. Inf. Sci. **169**(3–4), 249–262 (2005)

Thierens, D.: The linkage tree genetic algorithm. In: Parallel Problem Solving from Nature (PPSN XI). Lecture Notes in Computer Science, vol. 6238, pp. 264–273. Springer, Berlin (2011)

Tsutsui, S., Pelikan, M., Goldberg, D.: Node histogram vs. edge histogram: a comparison of pmbgas in permutation domains. Tech. Rep. 2006009, Missouri Estimation of Distribution Algorithms Laboratory (MEDAL), Department of Mathematics and Computer Science, University of Missouri–St. Louis (2006)

Tsutsui, S.: Probabilistic model-building genetic algorithms in permutation representation domain using edge histogram. In: Parallel Problem Solving from Nature (PPSN VII). Lecture Notes in Computer Science, vol. 2439, pp. 224–233. Springer, Berlin (2002)

Tsutsui, S., Pelikan, M., Goldberg, D.E.: Evolutionary algorithm using marginal histogram in continuous domain. In: Optimization by Building and Using Probabilistic Models (OBUPM) Workshop—Conference on Genetic and Evolutionary Computation (GECCO'01), pp. 230–233 (2001)

Valdez-Peña, S.I., Hernández-Aguirre, A., Botello-Rionda, S.: Approximating the search distribution to the selection distribution in EDAs. In: 11th Annual Conference on Genetic and Evolutionary Computation (GECCO'09), pp. 461–468. ACM, New York (2009)

Wang, L.F., Zeng, J.C.: Estimation of distribution algorithm based on copula theory. In: Exploitation of Linkage Learning in Evolutionary Algorithms. Evolutionary Learning and Optimization, vol. 3, pp. 139–162. Springer, Berlin (2010)

Wang, L.F., Zeng, J.C., Hong, Y.: Estimation of distribution algorithm based on Archimedean copulas. In: First ACM/SIGEVO Summit on Genetic and Evolutionary Computation (GEC'09), pp. 993–996. ACM, New York (2009)

Wang, X., Wang, H.: Evolutionary optimization with Markov random field prior. IEEE Trans. Evol. Comput. **8**(6), 567–579 (2004)

Weise, T., Niemczyk, S., Chiong, R., Wan, M.: A framework for multi-model EDAs with model recombination. In: Applications of Evolutionary Computation. Lecture Notes in Computer Science, vol. 6624, pp. 304–313. Springer, Berlin (2011)

Xiao, J., Yan, Y., Zhang, J.: HPBILc: a histogram-based EDA for continuous optimization. Appl. Math. Comput. **215**(3), 973–982 (2009)

Yanai, K., Iba, H.: Estimation of distribution programming based on Bayesian network. In: IEEE Congress on Evolutionary Computation (CEC'03), vol. 3, pp. 1618–1625 (2003)

Zhang, Q.: On stability of fixed points of limit models of univariate marginal distribution algorithm and factorized distribution algorithm. IEEE Trans. Evol. Comput. **8**(1), 80–93 (2004)

Zhang, Q., Zhou, A., Jin, Y.: RM-MEDA: a regularity model based multiobjective estimation of distribution algorithm. IEEE Trans. Evol. Comput. **12**(1), 41–63 (2008)