Chapter 1

Benefits of Data Clustering in Multimodal Function Optimization via EDAs

J. M. Peña, J. A. Lozano, P. Larrañaga
Intelligent Systems Group
Dept. of Computer Science and Artificial Intelligence
University of the Basque Country
Donostia-San Sebastián, Spain
{ccbpepaj, ccploalj, ccplamup}@si.ehu.es

Abstract This chapter shows how Estimation of Distribution Algorithms (EDAs) can benefit from data clustering in order to optimize both discrete and continuous multimodal functions. To be exact, the advantage of incorporating clustering into EDAs is two-fold: to obtain all the best solutions rather than only one of them, and to alleviate the difficulties that affect many evolutionary algorithms when more than one global optimum exists. We propose the use of Bayesian networks and conditional Gaussian networks to perform such a data clustering when EDAs are applied to optimization in discrete and continuous multimodal domains, respectively. The dynamics and performance of our approach are shown by evaluating it on a number of symmetrical functions, some of them highly multimodal.

Keywords: EDAs, data clustering, Bayesian networks, conditional Gaussian networks

1. Introduction

Many optimization problems present several global optima. However, most evolutionary algorithms are essentially developed to capture only one of the set of best solutions in the problem domain considered. For instance, given a multimodal function with several equally sized global peaks, the simple genetic algorithm (GA) can converge to only one of them. Moreover, this peak is randomly chosen due to the well-known *genetic drift* (De Jong, 1975): the simple GA has no means to decide among the different global peaks, and only the stochastic variations due to the genetic operators can make the population drift to one of these peaks. Unfortunately, this behaviour is shared by most evolutionary algorithms. This is a drawback as we are interested in obtaining all the global optima of both discrete and continuous multimodal functions.

Furthermore, this interest is not only quantitative but also qualitative. That is, the existence of several global peaks is a major source of difficulties for many evolutionary algorithms. Basically, the problems appear because combining good solutions coming from different parts of the search space or basins often results in poor solutions. If this is the case, convergence may be very slow until the population drifts to one of the basins or, even worse, the algorithm may get stuck in a local optimum. Apparently, these problems are aggravated in algorithms based on building and simulating probabilistic graphical models (EDAs) (Pelikan and Goldberg, 2000).

Consequently, we propose taking advantage of data clustering in order to effectively and efficiently discover all the global optima of a given optimization problem by using EDAs. The work presented in this chapter is clearly inspired by the research of Pelikan and Goldberg (2000). In that work, the authors introduce for the first time data clustering as a tool to alleviate the problems derived from the existence of several optima in symmetrical discrete domains when EDAs are applied to perform the optimization. Roughly speaking, their motivation is to process separately, at each generation, the complementary parts of the solution space. Unlike Pelikan and Goldberg (2000), our proposal does not divide explicitly the solution space into different parts but it takes advantage of probabilistic graphical models that are able to represent the complexity of the solution space. The models considered in this chapter are *Bayesian networks (BNs)* for discrete domains (Pearl, 1988; Peña et al., 2000a; Peña et al., 2001a), and *conditional Gaussian networks (CGNs)* for continuous domains (Lauritzen, 1992; Lauritzen, 1996; Lauritzen and Wermuth, 1989; Peña et al., 2000b; Peña et al., 2000c; Peña et al., 2001a). Thus, we present a unified framework to tackle both discrete and continuous multimodal function optimization problems that extends the idea behind Pelikan and Goldberg (2000).

The remainder of this chapter is organized as follows. Section 2 is a presentation of the motivation and objectives of our proposal to combine EDAs with data clustering via BNs and CGNs to optimize multimodal functions. Section 3 introduces BNs and CGNs applied to data clustering. Section 4 deals with some practical issues related to our proposal. We present some experimental results of the performance achieved by our approach in Section 5. Finally, we draw conclusions in Section 6.

2. Data Clustering in Evolutionary Algorithms for Multimodal Function Optimization

The combination of data clustering and evolutionary algorithms has proven to be very successful as evidenced by the large body of research conducted in this direction. However, the purpose of this combination varies from work to work. According to our aims, two of the most relevant works that propose the use of clustering inside the evolutionary algorithms framework are Hocaoğlu and Sanderson (1997), and Pelikan and Goldberg (2000). Whereas the authors of the first paper focus on discovering all peaks of multimodal functions, the authors of the second paper emphasize the goodness of data clustering to overcome the difficulties that appear when optimizing symmetrical discrete functions. We are interested here in a combination of the goals of these two papers.

The paper by Hocaoğlu and Sanderson (1997) presents the *minimal representation size cluster GA (MRSC_GA)*, previously introduced in Hocaoğlu and Sanderson (1995), applied to multimodal function optimization and path planning. Using minimal representation size clustering (Segen and Sanderson, 1981), the initial population is clustered into multiple subpopulations that evolve separately for a number of iterations. The aim is that each subpopulation converges to one of the different peaks of the given multimodal function. Moreover, minimal representation size clustering ensures that no two subpopulations achieve the same optimum. Occasionally, the different subpopulations are merged to obtain a new population by applying an operator similar to the crossover operator to individuals selected from different subpopulations. The process is repeated with the multiple separated subpopulations resulting from the clustering of this new population. According to the results reported, the MRSC_GA exhibits good behaviour when applied to multimodal function optimization without being provided with a priori knowledge of the solution space.

On the other hand, Pelikan and Goldberg (2000) motivate the use of data clustering in evolutionary algorithms in general and in EDAs in particular, as a means to overcome the disrupting effects that symmetry creates. Some other works that study the problem of symmetry in the search space are Collard and Aurand (1994), Naudts and Naudts (1998), Van Hoyweghen (2000), Van Hoyweghen and Naudts (2000). Whereas the last three works focus on the negative influence on the dynamics of GAs that symmetry has, the former paper demonstrates how to take advantage of symmetry to reach the optima more quickly. However, in many cases, the GA proposed in Collard and Aurand (1994) is not a realistic approach as it needs to be provided with a priori knowledge about the function to optimize.

Among the different types of symmetry described by Van Hoyweghen and Naudts (2000), the work by Pelikan and Goldberg (2000) deals with what it is known as *symmetry on the alphabet* or *spin-flip symmetry*. An objective function contains spin-flip symmetry when every pair of bit-complementary individuals has identical objective function values. Some functions suffering from spin-flip symmetry are graph partitioning problems, random number partitioning problems, graph coloring problems and two-max functions. Thus, the work by Pelikan and Goldberg (2000) is limited to combinatorial optimization where the individuals are strings of binary variables. The difficulties that spin-flip symmetry creates are due to the fact that combining promising solutions coming from complementary parts of the solution space often results in poor solutions. This fact may slow down convergence or even make the algorithm get stuck in a local optimum.

The purpose of Pelikan and Goldberg (2000) is to distinguish, at each generation, the complementary parts of the solution space in order to break the symmetry by treating each part separately. To be exact, they propose clustering the genotypes of the selected individuals of each generation. Then, each cluster is processed separately in the learning and simulation steps to obtain some offspring that are incorporated into the original population. The process is repeated until convergence is reached. Working in such a way avoids combining solutions that belong to complementary basins of the search space and results in an improvement of performance. As a side effect, multiple optima are discovered. According to results reported for the UMDA (Mühlenbein, 1998) using the Forgy algorithm (Forgy, 1965) to carry out the clustering, this approach works very well on spin-flip symmetrical functions of considerable dimension.

Although it is not discussed by Hocaoğlu and Sanderson (1997), the use of multiple separated subpopulations in the MRSC_GA also helps to avoid the harmful effects that the existence of several peaks involves. These undesirable effects are similar to those addressed above in the context of spin-flip symmetry.

2.1 Estimation of Mixture of Distributions Algorithm

The primary goal of this chapter is the proposal and evaluation of an enhancement of EDAs able to deal effectively and efficiently with both discrete and continuous multimodal function optimization problems. In the remainder of this chapter, this new EDA is referred to as *Estimation of Mixture of Distributions Algorithm (EMDA)*. In order to achieve our objective, the EMDA combines the benefits derived from the incorporation of data clustering into evolutionary algorithms that motivate Hocaoğlu and Sanderson (1997) with those that motivate Pelikan and Goldberg (2000). These benefits are:

- data clustering has proven to be an effective approach for overcoming the difficulties that multimodal function optimization involves for evolutionary algorithms in general and EDAs in particular, and
- data clustering is a reliable tool for obtaining all the global optima of multimodal functions. It should be noted that we aim to discover only global optima instead of local and global peaks as in Hocaoğlu and Sanderson (1997).

Unlike the work by Pelikan and Goldberg (2000), the EMDA does not rely on either the well-known Forgy algorithm or alternative techniques to carry out data clustering. Our proposal is true to the EDA paradigm as it is the model elicited from the selected individuals of each generation itself which encodes a probabilistic clustering of these individuals. Thus, the selected individuals are not explicitly divided into clusters to break the symmetry. On the contrary, the EMDA accepts the symmetry of the solution space because it takes advantage of probabilistic graphical models that are able to represent the complexity of the selected individuals. As a result, every model learnt at each iteration of the EMDA reveals the structure of the multimodal function that is being optimized, restricted to the selected individuals.

The models that we propose using are BNs when dealing with combinatorial optimization and CGNs when facing optimization problems in continuous domains. These two classes of probabilistic graphical models have been successfully applied to data clustering (Peña et al., 2000a; Peña et al., 2000b; Peña et al., 2000c; Peña et al., 2001a). Thus, the EMDA consists of the iteration of the same main steps as the generic EDA (Figure 1.1): selection of promising individuals from the current population, model learning from the selected individuals, and model sampling to obtain the offspring that are somehow incorporated into the current population to create the new population. Unsupervised learning of the BN or CGN should be provided with the number of clusters that we assume exist in the set of selected individuals. Ideally, this number should be the number of global optima of the function that we aim to optimize. If this is unknown, then it should be determined before starting the learning process involved in each iteration of the EMDA.

3. BNs and CGNs Applied to Data Clustering

One of the basic problems that arises in a great variety of fields, including pattern recognition, machine learning and statistics, is the so-called data clustering problem (Anderberg, 1973; Hartigan, 1975; Kaufman and Rousseeuw, 1990). Despite the different interpretations and expectations it gives rise to, the generic data clustering problem involves the assumption that, in addition to the observed variables or predictive attributes, there is a *hidden* variable. This last unobserved variable would reflect the cluster membership for every case in the database. Thus, the data clustering problem is also referred to as an example of learning from *incomplete data* due to the existence of such a hidden variable. Incomplete data represents a special case of *missing data* where all the missing entries are concentrated in a single variable: the hidden cluster variable. That is, we refer to a given database as incomplete when all the cases are unlabeled.

The fundamental data clustering problem aims to discover groups in data. Each of these groups is called a cluster, a region in which the density of objects is locally higher than in other regions. From the point of view adopted in this section, the data clustering problem may be defined as the inference of the joint generalized probability distribution for a given database. In the context of the EMDA, this database groups the selected individuals of the current iteration. Alternatively, data clustering can be viewed as a data partitioning problem. Given data, we can ask ourselves how the data can be split into different partitions dependent on a quality criterion (Pelikan and Goldberg, 2000).

3.1 Notation

We follow the usual convention of denoting variables by upper-case letters and their states by the same letters in lower-case. We use a letter or letters in bold-face upper-case to designate a set of variables and the same bold-face lower-case letter or letters to denote an assignment of a state to each variable in a given set. The joint generalized probability distribution of $\mathbf{X}$ is represented as $\rho(\mathbf{x})$. Additionally, $\rho(\mathbf{x} \mid \mathbf{y})$ denotes the generalized conditional probability distribution of $\mathbf{X}$ given $\mathbf{Y} = \mathbf{y}$. If all the variables in $\mathbf{X}$ are discrete, then $\rho(\mathbf{x}) = p(\mathbf{x})$ is the joint probability mass function of $\mathbf{X}$. Thus, $p(\mathbf{x} \mid \mathbf{y})$ denotes the conditional probability mass function of $\mathbf{X}$ given $\mathbf{Y} = \mathbf{y}$. If all the variables in $\mathbf{X}$ are continuous, then $\rho(\mathbf{x}) = f(\mathbf{x})$ is the joint probability density function of $\mathbf{X}$. Thus, $f(\mathbf{x} \mid \mathbf{y})$ denotes the conditional probability density function of $\mathbf{X}$ given $\mathbf{Y} = \mathbf{y}$.

When facing a data clustering problem, we assume the existence of a $(n+1)$-dimensional random variable $\mathbf{X}$ partitioned as $\mathbf{X} = (\mathbf{Y}, C)$ into a n-dimensional observed variable $\mathbf{Y}$ and a unidimensional discrete hidden variable C. In the particular case of every component Y_i of $\mathbf{Y}$ is discrete, the probabilistic graphical models that we aim to learn are called BNs. On the other hand, if every component Y_i of $\mathbf{Y}$ is continuous, then the probabilistic graphical models that we want to learn are called CGNs.

3.2 BNs for Data Clustering

Given a discrete random variable $\mathbf{X} = (\mathbf{Y}, C) = (Y_1, \ldots, Y_n, C)$, a BN for $\mathbf{X}$ (Pearl, 1988; Peña et al., 2000a; Peña et al., 2001a) is a graphical factorization of the joint probability distribution of $\mathbf{X}$. When applied to data clustering, a BN is defined by a directed acyclic graph $\mathbf{s}$ (model structure) determining the conditional (in)dependencies among the variables of $\mathbf{Y}$ and a set of local probability distributions. The model structure yields to a factorization of the joint probability distribution for $\mathbf{X}$ as follows:

$$p(\mathbf{x}) = p(c)p(\mathbf{y} \mid c) = p(c) \prod_{i=1}^{n} p(y_i \mid \mathbf{pa}(\mathbf{s})_i, c) \tag{1.1}$$

where $\mathbf{pa}(\mathbf{s})_i$ is the state of the parents of Y_i in $\mathbf{s}$, $\mathbf{Pa}(\mathbf{s})_i$, consistent with $\mathbf{x}$.

The local probability distributions of the BN are those in Equation 1.1 and we assume that they depend on a finite set of parameters $\boldsymbol{\theta}_{\mathbf{s}} \in \boldsymbol{\Theta}_{\mathbf{s}}$. Moreover, let $\mathbf{s}^h$ denote the hypothesis that the conditional (in)dependence assertions implied by $\mathbf{s}$ hold in the true joint probability distribution of $\mathbf{X}$. Therefore, Equation 1.1 can be rewritten as follows:

$$\begin{aligned} p(\mathbf{x} \mid \boldsymbol{\theta}_{\mathbf{s}}, \mathbf{s}^h) &= p(c \mid \boldsymbol{\theta}_{\mathbf{s}}, \mathbf{s}^h) p(\mathbf{y} \mid \boldsymbol{\theta}_{\mathbf{s}}^c, \mathbf{s}^h) \\ &= p(c \mid \boldsymbol{\theta}_{\mathbf{s}}, \mathbf{s}^h) \prod_{i=1}^{n} p(y_i \mid \mathbf{pa}(\mathbf{s})_i, \boldsymbol{\theta}_i^c, \mathbf{s}^h) \end{aligned} \tag{1.2}$$

where $\boldsymbol{\theta}_{\mathbf{s}}^c = (\boldsymbol{\theta}_1^c, \ldots, \boldsymbol{\theta}_n^c)$ denotes the parameters for the local probability distributions when $C = c$.

We limit our discussion to the case in which the local probability distributions of each variable of the BN consist of a set of multinomial distributions, one for each configuration of the parents and the cluster variable C.

3.3 CGNs for Data Clustering

A random variable $\mathbf{X} = (\mathbf{Y}, C) = (Y_1, \ldots, Y_n, C)$, where $\mathbf{Y}$ is continuous and C is discrete, is said to have a *conditional Gaussian distribution* (Lauritzen, 1992; Lauritzen, 1996; Lauritzen and Wermuth, 1989) if the distribution of $\mathbf{Y}$, conditioned on each state of C, is a multivariate normal distribution:

$$f(\mathbf{y} \mid C = c) \equiv \mathcal{N}(\mathbf{y}; \boldsymbol{\mu}(c), \boldsymbol{\Sigma}(c)) \tag{1.3}$$

whenever $p(c) = p(C = c) > 0$. Given $C = c$, $\boldsymbol{\mu}(c)$ is the n-dimensional mean vector, and $\boldsymbol{\Sigma}(c)$, the $n \times n$ variance matrix, is positive definite.

We define a CGN for $\mathbf{X}$ (Lauritzen, 1992; Lauritzen, 1996; Lauritzen and Wermuth, 1989; Peña et al., 2000b; Peña et al., 2000c; Peña et al., 2001a) as a probabilistic graphical model that encodes a conditional Gaussian distribution for $\mathbf{X}$. Thus, a CGN is defined by a directed acyclic graph $\mathbf{s}$ (model structure) determining the conditional (in)dependencies among the variables of $\mathbf{Y}$, a set of local probability density functions and a multinomial distribution for the variable C. The model structure yields to a factorization of the joint generalized probability density function for $\mathbf{X}$ as follows:

$$\rho(\mathbf{x}) = p(c)f(\mathbf{y} \mid c) = p(c)\prod_{i=1}^{n} f(y_i \mid \mathbf{pa(s)}_i, c) \tag{1.4}$$

where $\mathbf{pa(s)}_i$ is the state of the parents of Y_i in $\mathbf{s}$, $\mathbf{Pa(s)}_i$, consistent with $\mathbf{x}$.

The local probability density functions and the multinomial distribution of the CGN are those in the previous equation and we assume that they depend on a finite set of parameters $\boldsymbol{\theta}_{\mathbf{s}} \in \boldsymbol{\Theta}_{\mathbf{s}}$. Moreover, let $\mathbf{s}^h$ denote the hypothesis that the conditional (in)dependence assertions implied by $\mathbf{s}$ hold in the true joint generalized probability density function of $\mathbf{X}$. Therefore, Equation 1.4 can be rewritten as follows:

$$\begin{aligned} \rho(\mathbf{x} \mid \boldsymbol{\theta}_{\mathbf{s}}, \mathbf{s}^h) &= p(c \mid \boldsymbol{\theta}_{\mathbf{s}}, \mathbf{s}^h) f(\mathbf{y} \mid \boldsymbol{\theta}_{\mathbf{s}}^c, \mathbf{s}^h) \\ &= p(c \mid \boldsymbol{\theta}_{\mathbf{s}}, \mathbf{s}^h) \prod_{i=1}^{n} f(y_i \mid \mathbf{pa(s)}_i, \boldsymbol{\theta}_i^c, \mathbf{s}^h) \end{aligned} \tag{1.5}$$

where $\boldsymbol{\theta}_{\mathbf{s}}^c = (\boldsymbol{\theta}_1^c, \ldots, \boldsymbol{\theta}_n^c)$ denotes the parameters for the local probability density functions when $C = c$.

In order to encode a conditional Gaussian distribution for $\mathbf{X}$, each local probability density function of the CGN should be a linear-regression model. Thus, when $C = c$:

$$f(y_i \mid \mathbf{pa(s)}_i, \boldsymbol{\theta}_i^c, \mathbf{s}^h) \equiv \mathcal{N}(y_i; m_i^c + \sum_{y_j \in \mathbf{pa(s)}_i} b_{ji}^c (y_j - m_j^c), v_i^c) \tag{1.6}$$

where $\mathcal{N}(y; \mu, \sigma^2)$ is a univariate normal distribution with mean μ and standard deviation σ ($\sigma > 0$). Given this form, a missing arc from Y_j to Y_i implies that $b_{ji}^c = 0$ in the linear-regression model. When $C = c$, the local parameters are $\boldsymbol{\theta}_i^c = (m_i^c, \mathbf{b}_i^c, v_i^c)$, $i = 1, \ldots, n$, where $\mathbf{b}_i^c = (b_{1i}^c, \ldots, b_{i-1i}^c)^t$ is a column vector.

The interpretation of the components of the local parameters $\boldsymbol{\theta}_i^c$, $i = 1, \ldots, n$, is as follows: given $C = c$, m_i^c is the unconditional mean of Y_i, v_i^c is the conditional variance of Y_i given $\mathbf{Pa(s)}_i$, and b_{ji}^c, $j = 1, \ldots, i-1$, is a linear coefficient reflecting the strength of the relationship between Y_j and Y_i.

3.4 Unsupervised Learning of BNs and CGNs

One of the methods for learning BNs and CGNs from incomplete data is the well-known Bayesian Structural EM (BS-EM) algorithm developed by Friedman (1998). Due to its good performance, this algorithm has received special attention in the literature and has motivated several variants of itself (Meilă and Jordan, 1998; Peña et al., 1999; Peña et al., 2000a; Peña et al., 2001b; Thiesson et al., 1998). We use the BS-EM algorithm for explanatory purposes as well as in our experiments presented in Section 5.

EMDA
Generate M individuals at random to create the initial population ($\mathbf{d}_0$)
$l = 0$
Repeat until the stopping criterion is met
$l++$
Let $\mathbf{d}_{l-1}^{Se}$ group the N ($N \leq M$) individuals selected from $\mathbf{d}_{l-1}$ according to the selection method
Let $\rho_l(\mathbf{x})$ represent the joint generalized probability distribution of $\mathbf{x}$ encoded by a BN or CGN learnt from $\mathbf{d}_{l-1}^{Se}$ via the BS-EM algorithm
Generate the offspring by sampling T individuals from $\rho_l(\mathbf{x})$
Let $\mathbf{d}_l$ be the new population created by replacing part of $\mathbf{d}_{l-1}$ with part of the offspring by using the replacement method

where the BS-EM algorithm is as follows:

BS-EM algorithm
loop $l = 0, 1, \ldots$
Run the EM algorithm to get the MAP parameters $\widehat{\boldsymbol{\theta}}_{\mathbf{s}_l}$ for $\mathbf{s}_l$ given $\mathbf{o}$
Perform search over model structures, evaluating each one by

$$\begin{aligned} Score(\mathbf{s} : \mathbf{s}_l) &= E[\log \rho(\mathbf{h}, \mathbf{o}, \mathbf{s}^h) \mid \mathbf{o}, \widehat{\boldsymbol{\theta}}_{\mathbf{s}_l}, \mathbf{s}_l^h] \\ &= \textstyle\sum_{\mathbf{h}} p(\mathbf{h} \mid \mathbf{o}, \widehat{\boldsymbol{\theta}}_{\mathbf{s}_l}, \mathbf{s}_l^h) \log \rho(\mathbf{h}, \mathbf{o}, \mathbf{s}^h) \end{aligned}$$

Let $\mathbf{s}_{l+1}$ be the model structure with the highest score
if $Score(\mathbf{s}_l : \mathbf{s}_l) = Score(\mathbf{s}_{l+1} : \mathbf{s}_l)$ **then** return $(\mathbf{s}_l, \widehat{\boldsymbol{\theta}}_{\mathbf{s}_l})$

Figure 1.1 Schematics of the EMDA (top) and the BS-EM algorithm (bottom).

When applying the BS-EM algorithm in a data clustering problem, we assume that we have a database of N cases, $\mathbf{d} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, where every case is represented by an assignment to n of the $n+1$ variables involved in the problem domain. So, there are $(n+1)N$ random variables that describe the database. The N cases of the database correspond to the selected individuals at each iteration of the EMDA. Let $\mathbf{O}$ denote the set of observed variables, that is, the nN variables that have assigned values. Similarly, let $\mathbf{H}$ denote the set of hidden or unobserved variables, that is, the N variables that reflect the unknown cluster membership of each case of $\mathbf{d}$.

For learning BNs and CGNs from incomplete data, the BS-EM algorithm performs a search over the space of models based on the well-known *EM algorithm* (Dempster et al., 1977; McLachlan and Krishnan, 1997) and direct optimization of the Bayesian score. This results in an attempt to maximize the expected Bayesian score at each iteration instead of the true Bayesian score. As shown in Figure 1.1 (bottom), the BS-EM algorithm is comprised of two steps: an optimization of the BN or CGN parameters, usually by means of the EM algorithm, and a structural search for model selection.

To completely specify the BS-EM algorithm, we have to decide on the structural search procedure (the second step of the BS-EM algorithm of Figure 1.1). The usual approach is to perform a greedy hill-climbing search over BN or CGN structures considering all possible additions, removals and reversals of a single arc at each point in the search. This structural search procedure is desirable as it exploits the decomposition properties of BNs and CGNs, and the factorization properties of the Bayesian score for complete data. However, any structural search procedure that exploits these properties can be used. The log marginal likelihood of the expected complete data is usually chosen as the score to guide the structural search.

Direct application of the BS-EM algorithm as it appears in Figure 1.1 (bottom) may result in a unrealistic and inefficient solution because the computation of $Score(\mathbf{s} : \mathbf{s}_l)$ implies a huge computational expense as it takes account of every possible completion of the database. It is common to use a relaxed version of the presented algorithm that only considers the most likely completion of the database to compute $Score(\mathbf{s} : \mathbf{s}_l)$ instead of considering every possible completion. Thus, this relaxed version of the BS-EM algorithm is comprised of the iteration of a parametric optimization for the current model, and a structural search once the database has been completed with the most likely completion by using the best estimate of the joint generalized probability distribution of the data so far (current model). The completion is achieved by calculating the posterior probability distribution of the cluster variable C for each case of the database, $p(c \mid \mathbf{y}_i, \widehat{\boldsymbol{\theta}}_{\mathbf{s}_l}, \mathbf{s}_l^h)$. The case is assigned to the cluster where the maximum of this posterior probability distribution of C is reached. We use this relaxed version in our experiments.

It should be noted that the learnt model does not provide us with an explicit partition of the selected individuals into clusters but with an encoding of the joint generalized probability distribution of these individuals. Thus, the clusters determined by the learnt BN or CGN should be understood as probabilistic clusters. Instead of belonging to a particular cluster, each selected individual $\mathbf{y}_i$ implies a probability distribution for C, $p(c \mid \mathbf{y}_i, \widehat{\boldsymbol{\theta}}_{\mathbf{s}_l}, \mathbf{s}_l^h)$, that represents a probabilistic clustering of the individual.

4. Further Considerations About the EMDA

4.1 Sampling the Learnt Model

In order to generate the offspring to create the new population, the learnt model must be sampled. By doing this, the number of offspring produced by each probabilistic cluster is determined by the marginal probability distribution of the cluster variable C in the learnt model. This implies that the number of offspring sampled by each probabilistic cluster is somehow proportional to its size.

According to preliminary experiments, a decisive factor in performing efficient and effective multimodal function optimization is to keep in the population a reasonable number of individuals representing each of the basins of the global optima in order to avoid losing any of them as the optimization progresses. Based on these preliminary results, the model that the EMDA samples to generate the offspring in our experimental evaluation is not exactly the model learnt but a slightly modified one. As we are interested in discovering all global optima and they are equally sized peaks, the EMDA samples the same number of individuals from each of the probabilistic clusters encoded by the learnt model. This is equivalent to modifying the marginal probability distribution of C in the learnt model to be a uniform distribution. The effects of other sampling alternatives on the performance of the EMDA need to be studied. For instance, we could explore sampling a number of individuals from each probabilistic cluster proportional to its average fitness. Sampling a number of offspring from each probabilistic cluster inversely proportional to its size or to its average fitness could also be an alternative to consider. This last option aims to sample more individuals from the clusters with the least number of representatives or with the representatives with the worst average fitness, resulting in a positive discrimination of these probabilistic clusters that can aid their recovery.

4.2 Members of the EMDA Family

The EMDA relies on unsupervised learning of BNs and CGNs in order to obtain all the global optima while avoiding the harmful effects of multimodality. However, the reader should be aware of the existence of other probabilistic graphical models that could also provide us with the same benefits as BNs and CGNs when considered under the EMDA paradigm. For intance, Peña et al.(2001b) and Thiesson et al. (1998) present some probabilistic graphical models for data clustering that are more flexible than BNs and CGNs. Thus, the EMDA leads us to a family of EMDAs where the difference between the distinct members of this family consists of the class of probabilistic graphical models used to perform the clustering of the selected individuals.

It should be noted that the structure of a learnt BN or CGN for data clustering is independent of the value of the cluster variable C, and so, the model structure is the same for all values of C. However, the parameters of the local probability distributions depend on the value of C and they may differ for different values of C. It is interesting to note that the constraint of having a single model structure for every value of C can be relaxed by considering models more flexible than BNs and CGNs. An example is the class of what are known as *mixtures of DAG (MDAG) models* (Thiesson et al., 1998). Roughly speaking, MDAG models represent a generalization of BNs and CGNs applied to data clustering where different model structures for the different values of the variable C are allowed.

Despite having received less attention than BNs and CGNs in the recent literature, MDAG models appear to be more appropriate than BNs and CGNs under the EMDA paradigm for the optimization of multimodal functions. The explanation is straightforward. The advantage of EMDA over other members of the EDA paradigm is that the model learnt from the selected individuals, a BN or a CGN, is able to capture the multimodality of the function being optimized. Ideally, each probabilistic cluster of the learnt model corresponds to one of the several global optima that the function has. However, every probabilistic cluster involve the same set of conditional (in)dependence assertions as the rest, independently of the global optimum that is being modeled by that particular probabilistic cluster. This is because BNs and CGNs when applied to data clustering have a single model structure for all the values of C. On the other hand, MDAG models offer enough flexibility to encode a different set of conditional (in)dependencies for each probabilistic cluster. This fact together with the possibility of having different parameters for each probabilistic cluster make MDAG models desirable paradigms for multimodal function optimization. However, many of these problems lead us to make use of the EMDA whilst discarding the consideration of more flexible models such as MDAG models. That is, the solution space of many problems restricted to the selected individuals can be perfectly modeled by unsupervised learning of BNs or CGNs. Thus, the use of models more flexible than these would not contribute to an improvement in the performance of the algorithm. See, for instance, the 14 problems that we use in our experimental evaluation of Section 5 (most of these are taken from the existing literature).

Finally, another example of an evolutionary algorithm that may be seen as a member of the EMDA family is the *Adaptative Mixture Model Algorithm (AMix)* (Gallagher et al., 1999). The enhancement that this algorithm proposes consists of the use of a Gaussian mixture to model the joint probability distribution of the selected individuals of each generation. Moreover, the number of components in the mixture is allowed to vary during the execution of the algorithm as new data points are available. This is based on whether the Mahalanobis distance between the existing model and these new points is greater or not than a prespecified threshold. Gallagher et al. (1999) factorize each component of the Gaussian mixture into univariate Gaussian distributions, and then correlations among the different variables are not modeled. As far as we know, the AMix algorithm has not been studied in multimodal function optimization problems. However, it can be easily applied to them by simply considering that every component in the Gaussian mixture models the probability distribution of the selected individuals that belong to the basin of one of the global optima.

For the sake of conciseness, the remainder of this chapter considers the EMDA as it appears in Figure 1.1, i.e., the models learnt from the selected individuals are BNs for combinatorial optimization and CGNs for optimization in continuous domains. It is beyond the scope of this chapter to compare the different instances of what we have named the EMDA family. Moreover, BNs and CGNs are well-established classes of probabilistic graphical models that have been studied in depth. Besides, the use of models more flexible and, thus, more complex than BNs and CGNs to perform the clustering of the selected individuals (e.g. MDAG models) would also imply enlarging the optimization process as their unsupervised learning is usually computationally more expensive.

5. Experimental Results

This section is devoted to the experimental evaluation of the EMDA for combinatorial optimization as well as optimization in continuous domains. For this purpose, we use the UMDA (Mühlenbein, 1998) and the EBNA (Larrañaga et al., 2000b) as benchmarks for combinatorial optimization, and the UMDA$_c$ and the EGNA (Larrañaga et al., 2000a) for optimization in continuous domains. The comparison between the results achieved by the EMDA and those achieved by the benchmarks allows us to draw some conclusions about the efficiency and effectiveness of the EMDA.

Whereas the UMDA and the UMDA$_c$ are classic EDA instances, the EBNA and the EGNA have shown good performance on discrete and continuous optimization problems. Moreover, these two last algorithms are close in spirit to the EMDA as they are also based on learning and simulation of probabilistic graphical models. However, neither the EBNA nor the EGNA use probabilistic graphical models to perform data clustering, but they carry out supervised learning of them instead. The EBNA and the EGNA instances considered in this section make use of the B algorithm (Buntine, 1991) together with the BIC score (Schwarz, 1978) for the former and the BGe score (Geiger and Heckerman, 1995) for the latter, in order to perform learning of the probabilistic graphical models at each generation. We refer the interested reader to the original works for more details.

In order to keep the cost of the optimization process carried out by the EMDA as low as possible, we propose limiting the BS-EM algorithm to learning *Tree Augmented Naive Bayes (TANB) models* (Friedman and Goldszmidt, 1996; Meilă, 1999; Peña et al., 2000a; Peña et al., 2000c). This is a sensible as well as usual decision to reduce the otherwise large search spaces of BNs and CGNs. TANB models constitute a class of compromise BNs and CGNs defined by the following condition: predictive attributes may have at most one other predictive attribute as a parent. Thus, TANB models represent an interesting trade-off between efficiency and effectiveness, that is, a balance between the cost of the learning process and the quality of the learnt models (Peña et al., 2000a; Peña et al., 2000c).

It is well-known that TANB models are able to solve efficiently data clustering problems of considerable size. Moreover, they avoid the difficulties of learning densely connected BNs and CGNs, and the painfully slow probabilistic inference when working with these. Also, generation of the offspring from the learnt model is accelerated when this is a TANB model as few conditional dependencies among the variables are allowed.

5.1 General Considerations

In this subsection we discuss some decisions that are common to the EDA instances used in our experimental comparison (UMDA, UMDA$_c$, EBNA, EGNA and EMDA): selection method, population replacement method and stopping condition.

The four algorithms considered use truncation selection as the selection method, i.e, the best individuals of the current population according to their objective function values are selected. The way in which the new population is created consists of the replacement of the worst individuals of the current population by all the offspring.

The algorithms are stopped when the relative difference between the sum of the objective function values of all the individuals of the populations of two successive generations is less than a fixed value here referred to as precision.

The particular values for the population size, the number of selected individuals, the number of generated offspring and the precision may vary from objective function to objective function. We find it convenient to use a regular grammar to clearly identify these values. Thus, each of the objective functions used has an optimization scheduling represented as (α, β, γ, δ) where α is the size of the population, β is the number of selected individuals, γ is the size of the offspring and δ is the precision.

As noted earlier, unsupervised learning of the BN or CGN should be provided with the number of clusters that we assume exist in the set of selected individuals. In our experiments, this number is set to the number of global optima of the function that we aim to optimize.

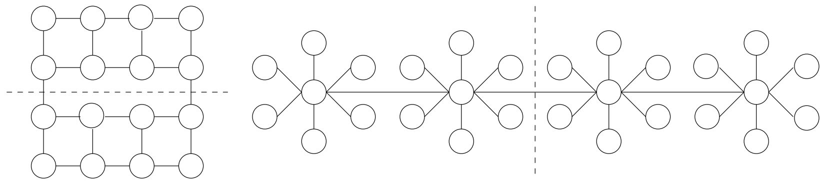

Figure 1.2 Graph structures for $F_{grid16}(\mathbf{x})$ (left) and $F_{cat28}(\mathbf{x})$ (right). Dashed lines indicate the optimal cuts.

5.2 Combinatorial Optimization

This subsection evaluates the EMDA as applied to combinatorial optimization. Most of the problems considered can be found in Pelikan and Goldberg (2000) and Pelikan et al. (2000). All the problems are defined in $\{0, 1\}^n$, i.e., the set of binary individuals of length n.

We limit our current evaluation to multimodal functions that show spin-flip symmetry in the solution space. This class of multimodal functions represents a set of challenging problems for many EDAs and GAs (Naudts and Naudts, 1998; Pelikan and Goldberg, 2000; Van Hoyweghen, 2000; Van Hoyweghen and Naudts, 2000).

5.2.1 Problems.

Two-max Problems. These are two simple spin-flip symmetrical functions:

$$F_{two-max}(\mathbf{x}) = |\frac{n}{2} - u|$$

$$F_{two-max2}(\mathbf{x}) = \begin{cases} F_{two-max}(\mathbf{x}) - 5 & \text{if } F_{two-max}(\mathbf{x}) > 5 \\ F_{two-max}(\mathbf{x}) & \text{otherwise} \end{cases}$$

where u is the sum of the bits in $\mathbf{x}$ and n is the length of $\mathbf{x}$. The objective is to maximize the functions. For both functions, there are two global optima: $\mathbf{x}_1^*=(0, \ldots, 0)$ and $\mathbf{x}_2^*=(1, \ldots, 1)$ with fitness equal to $\frac{n}{2}$ for $F_{two-max}$, and equal to $\frac{n}{2} - 5$ for $F_{two-max2}$. In our case, $n = 50$. The optimization scheduling for both functions is (2000, 1000, 1999, 0). $F_{two-max2}$ is considered more difficult than $F_{two-max}$ as it has two local optima in addition to the two global optima.

Graph Bisection Problems. Graph bisection problems aim to split the set of nodes of a given graph structure into two equally sized subsets so that the number of edges between the two subsets is minimized. We use two grid-like

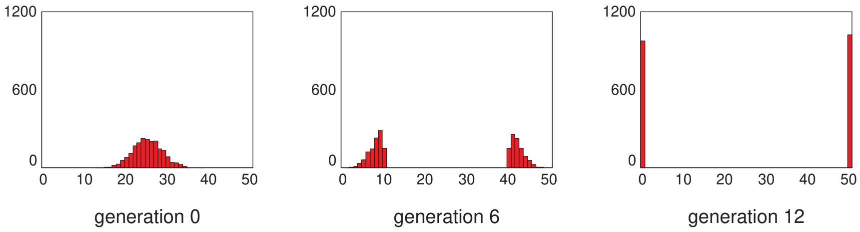

Figure 1.3 Dynamics of the EMDA in the $F_{two-max}$ problem. The horizontal axis represents the number of ones in a solution whereas the vertical axis denotes the number of corresponding solutions in the population of the generation indicated.

graph structures cut in halves and connected by two edges, with sizes $n = 16, 36$ resulting in the problems $F_{grid16}(\mathbf{x})$ and $F_{grid36}(\mathbf{x})$, respectively. Also, we consider three so-called caterpillar graph structures with sizes $n = 28, 42, 56$ that result in $F_{cat28}(\mathbf{x})$, $F_{cat42}(\mathbf{x})$ and $F_{cat56}(\mathbf{x})$, respectively. Figure 1.2 shows the graph structures for $F_{grid16}(\mathbf{x})$ and $F_{cat28}(\mathbf{x})$.

Each bit of a given individual represents one node of the graph structure. The value of the bit classifies the corresponding node into one of the two subsets. It should be noted that only individuals with equal numbers of zeroes and ones represent feasible solutions. Thus, some individuals may need to be repaired. Although more specialized repair operators might be considered, we make use of a simple randomized repair operator: a unfeasible solution is converted into a feasible one by randomly picking a number of the bits that are in the majority and changing them to their complementary values.

The fitness of a given individual is calculated as the size of the graph structure minus the number of edges connecting the two subsets of nodes encoded in the individual. Thus, the objective is to maximize. The global optima have an objective function value equal to $n - 2$ for $F_{grid16}(\mathbf{x})$ and $F_{grid36}(\mathbf{x})$, and equal to $n-1$ for $F_{cat28}(\mathbf{x})$, $F_{cat42}(\mathbf{x})$ and $F_{cat56}(\mathbf{x})$. It is easy to see that these five problems are spin-flip symmetrical problems and, thus, the global optima are represented by complementary individuals. The optimization scheduling is (2000, 1000, 1999, 0) for the five graph bisection problems.

In addition to the difficulties derived from their symmetrical nature, these graph bisection problems present another source of problems for GAs and EDAs due to the fact that there are many local optima and only two global optima (Pelikan and Goldberg, 2000; Pelikan et al., 2000; Schwarz and Ocenasek, 1999).

5.2.2 Results. Figure 1.3 shows the dynamics of the EMDA in the $F_{two-max}$ problem. The histograms summarize the number of solutions (vertical axis) in the populations of generations 0, 6 and 12 with the number of ones denoted by the horizontal axis. As previously stated, the two global optima are

Table 1.1 Performance of the UMDA, EBNA and EMDA in the discrete domains considered. The numbers of evaluations and runtimes are average values over 10 independent runs. The numbers of times that each optima is reached summarize the final results of these 10 runs.

Problem	*UMDA*	*EBNA*	*EMDA*
$F_{two-max}$	35183 eval.	31185 eval.	25988 eval.
	20 sec.	160 sec.	107 sec.
	(5, 5)	(7, 3)	(10, 10)
$F_{two-max2}$	49176 eval.	40181 eval.	27987 eval.
	33 sec.	207 sec.	194 sec.
	(6, 4)	(4, 6)	(10, 10)
F_{grid16}	57972 eval.	60971 eval.	23989 eval.
	59 sec.	46 sec.	39 sec.
	(5, 5)	(4, 6)	(10, 10)
F_{grid36}	81560 eval.	120941 eval.	38182 eval.
	68 sec.	336 sec.	161 sec.
	(5, 4)	(9, 1)	(10, 10)
F_{cat28}	50776 eval.	53374 eval.	25988 eval.
	34 sec.	80 sec.	82 sec.
	(1, 8)	(4, 6)	(10, 10)
F_{cat42}	66968 eval.	96953 eval.	33184 eval.
	58 sec.	310 sec.	189 sec.
	(2, 3)	(4, 4)	(10, 10)
F_{cat56}	87957 eval.	120741 eval.	39981 eval.
	91 sec.	776 sec.	359 sec.
	(3, 4)	(1, 2)	(10, 10)

complementary and are on the left-most and right-most sides of the histograms. It is clear that, as the optimization progresses, the population drifts to both sides as the BN learnt at each iteration of the EMDA is able to capture this division of the selected individuals. Finally, both global optima are discovered and seen in the population of the last generation of the EMDA. Moreover, the individuals of this last population are almost equally distributed between both global peaks.

Table 1.1 summarizes the results achieved when applying the EMDA to each of the 7 combinatorial problems presented in the previous subsection. Additionally, this table reports the results reached by the UMDA and the EBNA in these problems for comparison. For each problem and each evolutionary algorithm three results are given: the number of evaluations of the objective function until the stopping criterion is met, the runtime of the optimization process (in seconds), and the number of times that each of the global optima of the objective function is discovered. The first two results are average values over 10 independent runs, but the third result summarizes these 10 runs by a pair (η, ξ), where η and ξ are the number of times that $\mathbf{x}_1^*$ and $\mathbf{x}_2^*$ are obtained respectively. Obviously, the UMDA and the EBNA are able to reach at most one global optima per run while the EMDA could reach both optima in each run.

From Table 1.1 we can conclude that the EMDA shows a more effective and efficient behaviour than the other two algorithms for the problems chosen. It is specially appealing that, for all the objective functions, the EMDA needs less evaluations of the objective function than the UMDA and the EBNA to reach convergence without degrading the quality of the obtained solutions (the two global optima of every function are obtained in every run). Except in the two-max problems, the EBNA needs more evaluations than the UMDA to converge.

As expected, the number of evaluations has a decisive influence on the runtime of the optimization process measured in seconds. Here, the optimization process using the EBNA is the slowest of the three. On the other hand, the UMDA is the quickest although the number of evaluations that it needs to converge in any of the 7 problems is much larger than the number of evaluations needed by the EMDA. Obviously, this is due to the unsupervised learning of BNs that the EMDA performs which is known to be a difficult and, sometimes, computationally expensive process. However, the runtime of the EMDA in this set of problems is considered reasonable.

Looking at Table 1.1 we discover that to converge and obtain the two complementary global optima of any of the 7 functions using the EMDA involves less evaluations than to converge and obtain only one using the UMDA or the EBNA. Thus, these results confirm what we already proposed: the incorporation of probabilistic clustering into EDAs is not only interesting because it allows all the global optima to be obtained, but also because it deals with symmetry in a natural way. That is, it avoids the combination of good solutions coming from complementary parts of the solution space which often results in poor solutions that slow down convergence. We categorized this dual interest in developing the EMDA as quantitative and qualitative, i.e., a gain in effectiveness together with a gain in efficiency.

From the point of view of effectiveness measured as the number of global optima recovered by each algorithm for each function, the EMDA outperforms the two benchmarks. This is not surprising as this was part of the motivation that led us to propose it. Specifically, the EMDA always discovers the global optima independently of the actual problem. Moreover, the individuals of the last population of every run of the EMDA for any of the 7 domains are equally distributed between the two global optima. On the other hand, the UMDA and the EBNA suffer the effects of the symmetry and the existence of several peaks, specially in the caterpillar graph bisection problems. Their harmful effects can be observed in the fact that, for the 7 functions chosen, the UMDA and the EBNA need a larger number of evaluations to converge and discover at most one global optima per run, than the EMDA for reaching convergence and discover both global optima. In addition to this, it can be appreciated from Table 1.1 that the caterpillar graph bisection problems are extremely difficult for the UMDA and the EBNA. These two algorithms exhibit a poor performance in those problems as they get stuck in local optima in 9 out of the 30 runs performed for the three caterpillar graph bisection problems. The EMDA exhibits a unbeatable behaviour in these particular problems.

These results prove the goodness of the EMDA in particular and the combination of EDAs and probabilistic clustering in general, in alleviating the disrupting effects of spin-flip symmetry and in obtaining all the global optima in the objective function.

5.3 Optimization in Continuous Domains

This subsection evaluates the EMDA as applied to optimization in continuous domains. We limit our current evaluation to multimodal functions that show symmetry in the solution space. Specifically, we consider that a function $F(\mathbf{x})$ exhibits symmetry in the solution space with respect to $\mathbf{a}$ when $F(\mathbf{a}+\mathbf{x}) = F(\mathbf{a}-\mathbf{x})$ for all $\mathbf{x}$ in the domain. As in the discrete case, this class of multimodal functions represents a set of challenging problems since they involve the same harmful effects on many evolutionary algorithms as spin-flip symmetrical functions.

5.3.1 Problems.

Two-max Problems. These are two simple symmetrical functions similar to the discrete two-max problems introduced in the evaluation of the EMDA in combinatorial optimization problems:

$$F_{two-max}(\mathbf{x}) = |\sum_{i=1}^{n} x_i| \qquad -5 \leq x_i \leq 5 \qquad i = 1, \ldots, n$$

$$F_{two-max2}(\mathbf{x}) = \begin{cases} F_{two-max}(\mathbf{x}) - 30 & \text{if } F_{two-max}(\mathbf{x}) > 30 \\ F_{two-max}(\mathbf{x}) & \text{otherwise} \end{cases}$$

$$-10 \leq x_i \leq 10 \qquad i = 1, \ldots, n$$

where the objective is to maximize these functions. For both functions, there are two global optima: $\mathbf{x}_1^*$=(-5, ... , -5) and $\mathbf{x}_2^*$=(5, ... , 5) for $F_{two-max}$ with fitness equal to $5n$, and $\mathbf{x}_1^*$=(-10, ... , -10) and $\mathbf{x}_2^*$=(10, ... , 10) for $F_{two-max2}$ with fitness equal to $10n-30$. In our case, $n = 10$. The optimization scheduling for both functions is (2000, 1000, 1999, 1). $F_{two-max2}$ is considered more difficult than $F_{two-max}$ as it has two local optima in addition to the two global optima.

Mixture of Normal Distributions Problems. The first example of this class of problems that we consider can be defined as the joint probability density function of a mixture of two normal distributions with different mean vectors:

$$F_{mix1}(\mathbf{x}) = 0.5\,\mathcal{N}(\mathbf{x};\boldsymbol{\mu}_1,\boldsymbol{\Sigma}) + 0.5\,\mathcal{N}(\mathbf{x};\boldsymbol{\mu}_2,\boldsymbol{\Sigma})$$

where $\mathcal{N}(\mathbf{x};\boldsymbol{\mu},\boldsymbol{\Sigma})$ is a multivariate normal distribution with n-dimensional mean vector $\boldsymbol{\mu}$ and $n \times n$ variance matrix $\boldsymbol{\Sigma}$. In our problem, $\boldsymbol{\mu}_1$ = (-1, ... , -1) and $\boldsymbol{\mu}_2$ = (1, ... , 1). Moreover, we consider that the variance matrix is diagonal with all the elements of the diagonal equal to 1. There are two global optima: $\mathbf{x}_1^*$=(-1, ... , -1) and $\mathbf{x}_2^*$=(1, ... , 1).

We also use two more examples of the mixture of normal distributions problems here denoted as F_{mix2} and F_{mix3}. They are similar to F_{mix1} but in these cases the non-zero elements of the variance matrix are equal to 4 for F_{mix2} and equal to 9 for F_{mix3}. The two global optima of F_{mix2} are approximately $\mathbf{x}_1^*$=(-0.99, ... , -0.99) and $\mathbf{x}_2^*$=(0.99, ... , 0.99). For F_{mix3} the global optima are around $\mathbf{x}_1^*$=(-0.52, ... , -0.52) and $\mathbf{x}_2^*$=(0.52, ... , 0.52).

The objective for the three functions is maximization, and $-3 \leq x_i \leq 3$ for $i = 1, \ldots, n$. In our case $n = 10$. The optimization scheduling is (2000, 1000, 1999, 10^{-8}) for the first function and (2000, 1000, 1999, 10^{-10}) for the other two. It is easy to see that the three functions have been introduced in an increasing order of difficulty.

Shekel's foxholes problems. We consider two instances of the well-known multimodal Shekel's foxholes problem (De Jong, 1975). The first instance is as follows:

$$F_{Shekel1}(\mathbf{x}) = -\sum_{j=1}^{m} \frac{1}{||\mathbf{x} - \mathbf{x}_j^*||^2 + c_j}$$

where m is the number of global optima and c_j, $j = 1, \ldots, m$, is a coefficient that determines the height of each of the global peaks. The objective is minimization. In our case, $m = 2$, $c_1 = c_2 = 0.001$, $\mathbf{x}_1^*$=(1, ... , 1) and $\mathbf{x}_2^*$=(3, ... , 3). Thus, the value of the objective function in the global minima is equal to -1000. Moreover, $0 \leq x_i \leq 4$ for $i = 1, \ldots, n$. The dimension of the problem is $n = 5$. The optimization scheduling is (2000, 1000, 1999, 50).

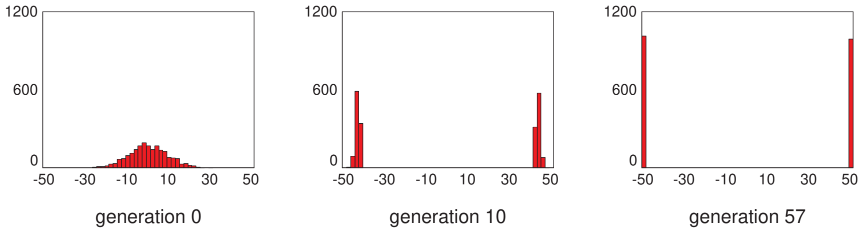

Figure 1.4 Dynamics of the EMDA in the continuous $F_{two-max}$ problem. The horizontal axis represents the sum of the genes of a solution whereas the vertical axis denotes the number of corresponding solutions in the population of the generation indicated.

We refer to the second instance of the Shekel's foxholes problem as $F_{Shekel2}$. In this case $m = 3$, $c_1 = c_2 = c_3 = 0.001$, $\mathbf{x}_1^*=(1, \ldots, 1)$, $\mathbf{x}_2^*=(4, \ldots, 4)$ and $\mathbf{x}_3^*=(7, \ldots, 7)$. The value of the objective function in the global minima is equal to -1000, and $0 \leq x_i \leq 8$ for $i = 1, \ldots, n$. The dimension of the problem is $n = 5$. The optimization scheduling is (5000, 1000, 500, 50). The objective is also minimization.

5.3.2 Results. Figure 1.4 shows the dynamics of the EMDA in the continuous $F_{two-max}$ problem. The histograms summarize the number of solutions (vertical axis) in the populations of generations 0, 10 and 57 whose sum of genes is equal to the value denoted by the horizontal axis. The two global optima are on the left-most and right-most sides of the histograms. Thus, it is clear that, as the EMDA progresses, the population drifts to both sides since the CGN learnt at each iteration is able to capture this division of the selected individuals. Finally, both global optima are discovered and seen in the population of the last generation of the EMDA. Moreover, the individuals of this last population are almost equally distributed between both global peaks.

Table 1.2 summarizes the results achieved when applied the EMDA to each of the 7 optimization problems presented in the previous subsection. Additionally, this table reports the results reached by the UMDA$_c$ and the EGNA in these problems for comparison. For each problem and each evolutionary algorithm three results are given: the number of evaluations of the objective function until the stopping criterion is met, the runtime of the optimization process (in seconds), and the number of times that each of the global optima of the objective function is discovered. The first two results are average values over 10 independent runs. The third result is encoded using the same system as in Section 5.2.2.

Roughly speaking, the results achieved for the continuous domains repeat the patterns that appear for combinatorial optimization. Let us analyze in detail

Table 1.2 Performance of the UMDA$_c$, EGNA and EMDA in the continuous domains considered. The numbers of evaluations and runtimes are average values over 10 independent runs. The numbers of times that each optima is reached summarize the final results of these 10 runs.

Problem	*UMDA$_c$*	*EGNA*	*EMDA*
$F_{two-max}$	104149 eval.	161120 eval.	115343 eval.
	36 sec.	104 sec.	152 sec.
	(4, 6)	(5, 5)	(10, 10)
$F_{two-max2}$	169516 eval.	185108 eval.	141130 eval.
	84 sec.	142 sec.	172 sec.
	(2, 8)	(6, 4)	(10, 10)
F_{mix1}	90556 eval.	92955 eval.	78962 eval.
	58 sec.	58 sec.	80 sec.
	(7, 3)	(5, 5)	(10, 10)
F_{mix2}	77562 eval.	88357 eval.	63769 eval.
	53 sec.	105 sec.	61 sec.
	(8, 2)	(4, 6)	(10, 10)
F_{mix3}	50776 eval.	57972 eval.	44179 eval.
	20 sec.	47 sec.	50 sec.
	(0, 0)	(3, 7)	(10, 10)
$F_{Shekel1}$	77162 eval.	102750 eval.	56773 eval.
	30 sec.	49 sec.	37 sec.
	(5, 5)	(5, 5)	(10, 10)
$F_{Shekel2}$	40000 eval.	39900 eval.	42850 eval.
	7 sec.	18 sec.	85 sec.
	(0, 10, 0)	(0, 10, 0)	(10, 10, 10)

the results summarized in Table 1.2. Except in $F_{two-max}$ (only for UMDA$_c$) and $F_{Shekel2}$, the EMDA needs a smaller number of evaluations of the objective function than the UMDA$_c$ and the EGNA to achieve convergence. Thus, the EMDA exhibits a more efficient behaviour than the other two evolutionary algorithms. Despite this, the EMDA usually takes a larger runtime than the other two algorithms. Again, the reason is the unsupervised learning of CGNs performed by the EMDA. However, its runtime is considered reasonable.

In addition to being the most efficient, the results of Table 1.2 confirm that the EMDA is also the most effective of the three algorithms considered: it always discovers all the global optima that exist in the 7 functions chosen. Moreover, except for $F_{two-max}$ (only for UMDA$_c$) and $F_{Shekel2}$, the EMDA is able to converge to all the global optima in all the runs in a number of evaluations smaller than the UMDA$_c$ and the EGNA whereas these algorithms discover at most just one of the existing optima. Thus, the EMDA reveals once again its benefits for multimodal function optimization from a qualitative as well as a quantitative point of view. To reinforce the effective behaviour shown by the EMDA, we should add that the individuals of the last population of every run of the EMDA for any of the 7 domains are equally distributed between the existing global optima. On the other hand, the UMDA$_c$ suffers the effects of the symmetry of the solution space when dealing with F_{mix3} and it is unable to achieve any of the global optima of this function in the 10 runs carried out.

Finally, we should conclude that, as seen in the combinatorial optimization problems previously considered, the EMDA when applied to optimization in continuous domains fulfills all its objectives.

6. Conclusions

The main contribution of this chapter has been the introduction of a new member of the EDA family: the EMDA (Estimation of Mixture of Distributions Algorithm). The motivation that has led us to the EMDA was two-fold. First, we wanted to obtain all the global optima when facing both discrete and continuous multimodal function optimization problems. Second, the optimization process needed to be efficient in addition to effective, i.e., it had to be able to overcome the difficulties derived from the existence of several global peaks in the function to optimize.

The main steps of the EMDA are the same as in any other EDA: selection of promising individuals, model learning and model sampling to generate a new population. The improvement of the EMDA over other EDAs is the model to be learnt at each iteration. This model is intended to capture the multimodality of the function to be optimized by clustering the selected individuals according to their genotypes. This avoids the harmful effects of multimodality as individuals from different parts of the search space are treated separately. Furthermore, each cluster should ideally evolve to a different global peak.

Unlike other works that divide the set of selected individuals of each generation into a set of clusters, the EMDA does not perform such an explicit partition of the selected individuals. The EMDA makes use of two well-known classes of probabilistic graphical models to cluster the selected individuals: BNs for combinatorial optimization and CGNs for continuous optimization. This makes the EMDA fit EDAs in a natural way as well as representing a unified framework for combinatorial as well as continuous multimodal function optimization.

Empirical evaluation of the EMDA for combinatorial as well as continuous optimization has been limited to some symmetrical functions. The functions chosen are known to be difficult problems for many evolutionary algorithms. This point has been confirmed by the results reported: the EMDA has outperformed the UMDA, UMDA$_c$, EBNA and EGNA in the number of evaluations of the objective functions needed to converge and all the global optima were discovered for all the problems considered. This proves that the EMDA is able to deal with multimodal functions and discover all existing global optima while alleviating the harmful effects that the existence of several global peaks implies for many other evolutionary algorithms.

Acknowledgments

This work was supported by the Spanish Ministry of Education and Culture under AP97 44673053 grant.

References

Anderberg, M. R. (1973). *Cluster Analysis for Applications.* Academic Press.

Buntine, W. (1991). Theory refinement in Bayesian networks. In *Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence*, pages 52–60. Morgan Kaufmann, Inc.

Collard, P. and Aurand, J. P. (1994). DGA: An efficient genetic algorithm. In *Proceedings of the European Conference on Artificial Intelligence 1994*, pages 487–492. John Wiley & Sons, Inc.

De Jong, K. A. (1975). *An analysis of the behavior of a class of genetic adaptive systems.* Doctoral Dissertation. University of Michigan.

Dempster, A., Laird, N., and Rubin, D. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, 39:1–38.

Forgy, E. (1965). Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications. *Biometrics*, 21:768.

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

Friedman, N. and Goldszmidt, M. (1996). Building classifiers using Bayesian networks. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pages 1277–1284. AAAI Press.

Gallagher, M., Frean, M., and Downs, T. (1999). Real-valued Evolutionary Optimization using a Flexible Probability Density Estimator. In *Proceedings of the Genetic and Evolutionary Computation Conference 1999*, pages 840–846.

Geiger, D. and Heckerman, D. (1995). Learning Gaussian Networks. In *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, pages 235–243.

Hartigan, J. A. (1975). *Clustering Algorithms.* John Wiley & Sons, Inc.

Hocaoğlu, C. and Sanderson, A. C. (1995). Evolutionary speciation using minimal representation size clustering. In *Evolutionary Programming IV: Proceedings of the Fourth Annual Conference on Evolutionary Programming*, pages 187–203. MIT Press.

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

Kaufman, L. and Rousseeuw, P. (1990). *Finding Groups in Data.* John Wiley & Sons, Inc.

Larrañaga, P., Etxeberria, R., Lozano, J. A., and Peña, J. M. (2000a). 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, Inc.

Larrañaga, P., Etxeberria, R., Lozano, J. A., and Peña, J. M. (2000b). Optimization in continuous domains by learning and simulation of Gaussian networks. In *Genetic and Evolutionary Computation Conference 2000. Proceedings of the Program Workshops*, pages 201–204. Morgan Kaufmann, Inc.

Lauritzen, S. L. (1992). Propagation of probabilities, means and variances in mixed graphical association models. *Journal of the American Statistical Association*, 87(420):1098–1108.

Lauritzen, S. L. (1996). *Graphical Models.* Clarendon Press.

Lauritzen, S. L. and Wermuth, N. (1989). Graphical models for associations between variables, some of which are qualitative and some quantitative. *The Annals of Statistics*, 17:31–57.

McLachlan, G. J. and Krishnan, T. (1997). *The EM Algorithm and Extensions.* John Wiley & Sons, Inc.

Meilă, M. (1999). *Learning with Mixtures of Trees.* Doctoral Dissertation. Massachusetts Institute of Technology.

Meilă, M. and Jordan, M. I. (1998). Estimating dependency structure as a hidden variable. *Neural Information Processing Systems*, 10:584–590.

Mühlenbein, H. (1998). The Equation for Response to Selection and its Use for Prediction. *Evolutionary Computation*, 5:303–346.

Naudts, B. and Naudts, J. (1998). 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. Lectures Notes in Computer Science.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems.* Morgan Kaufmann, Inc.

Pelikan, M. and Goldberg, D. E. (2000). Genetic Algorithms, Clustering, and the Breaking of Symmetry. In *Proceedings of Parallel Problem Solving from Nature VI*, pages 385–394. Springer-Verlag. Lectures Notes in Computer Science.

Pelikan, M., Goldberg, D. E., and Sastry, K. (2000). Bayesian Optimization Algorithm, Decision Graphs, and Occam's Razor. Technical Report IlliGAL No. 2000020, Illinois.

Peña, J. M., Lozano, J. A., and Larrañaga, P. (1999). Learning Bayesian networks for clustering by means of constructive induction. *Pattern Recognition Letters*, 20(11-13):1219–1230.

Peña, J. M., Lozano, J. A., and Larrañaga, P. (2000a). An improved Bayesian structural EM algorithm for learning Bayesian networks for clustering. *Pattern Recognition Letters*, 21(8):779–786.

Peña, J. M., Lozano, J. A., and Larrañaga, P. (2000b). Learning conditional Gaussian networks for data clustering via edge exclusion tests. Submitted.

Peña, J. M., Lozano, J. A., and Larrañaga, P. (2000c). Performance evaluation of compromise conditional Gaussian networks for data clustering. Submitted.

Peña, J. M., Lozano, J. A., and Larrañaga, P. (2001a). 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*. Morgan Kaufmann, Inc.

Peña, J. M., Lozano, J. A., and Larrañaga, P. (2001b). Learning recursive Bayesian multinets for data clustering by means of constructive induction. *Machine Learning*, to appear.

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

Schwarz, J. and Ocenasek, J. (1999). 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.

Segen, J. and Sanderson, A. C. (1981). Model inference and pattern discovery by minimal representation method. Technical Report CMU-RI-TR-82-2, Carnegie Mellon University.

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

Van Hoyweghen, C. (2000). Detecting Spin-flip Symmetry in Optimization Problems. *Theoretical Aspects of Evolutionary Computing.*

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