Information Sciences xxx (2013) xxx–xxx

Contents lists available at SciVerse ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

A review on evolutionary algorithms in Bayesian network learning and inference tasks

Pedro Larrañaga [a], Hossein Karshenas [a,*], Concha Bielza [a], Roberto Santana [b]

[a] Computational Intelligence Group, Facultad de Informática, Universidad Politécnica de Madrid, Campus de Montegancedo, 28660 Boadilla del Monte, Madrid, Spain
[b] Intelligent System Group, Department of Computer Science and Artificial Intelligence, University of the Basque Country, Paseo Manuel de Lardizabal 1, 20080 San Sebastian–Donostia, Spain

A R T I C L E I N F O

Article history:
Received 14 May 2012
Received in revised form 19 October 2012
Accepted 21 December 2012
Available online xxxx

Keywords:
Probabilistic graphical model
Bayesian network
Evolutionary computation
Inference
Learning from data

A B S T R A C T

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. Bayesian networks are one of the most widely used class of these models. Some of the inference and learning tasks in Bayesian networks involve complex optimization problems that require the use of meta-heuristic algorithms. Evolutionary algorithms, as successful problem solvers, are promising candidates for this purpose. This paper reviews the application of evolutionary algorithms for solving some NP-hard optimization tasks in Bayesian network inference and learning.

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 [105], 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 to 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 [86] and automatic induction [23] from a set of samples.

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 [59] have been able to perform a more effective and diverse search of the vast solution space of difficult problems.

* Corresponding author. Tel.: +34 913363675; fax: +34 913524819.
E-mail addresses: pedro.larranaga@fi.upm.es (P. Larrañaga), hkarshenas@fi.upm.es (H. Karshenas), mcbielza@fi.upm.es (C. Bielza), roberto.santana@ehu.es (R. Santana).

0020-0255/$ - see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.ins.2012.12.051

Some of the most relevant inference and learning problems in Bayesian networks are formulated as the optimization of a function. These problems usually have an intractable complexity and therefore are a potential domain for the application of meta-heuristic methods. The aim of this paper is to review how evolutionary algorithms have been applied for solving some of the combinatorial problems existing in the inference and learning of Bayesian networks.

The paper is organized as follows. Section 2 introduces Bayesian networks and reviews some of the inference and learning methods proposed for them. Section 3 presents the framework of evolutionary algorithms and discusses how they work. The main review of how evolutionary algorithms are used in Bayesian network learning and inference is given in Section 4. Finally, Section 5 concludes the paper.

2. Bayesian networks

This section gives an introduction to Bayesian networks and how they are used for representing probability distributions in discrete, continuous, and hybrid environments. It then briefly reviews some of the methods for inference and learning of Bayesian networks. The terminology and concepts adopted and introduced in this section are later used in the presentation of evolutionary algorithms for learning and inference in Bayesian networks. For more information on Bayesian networks and PGMs in general, see Koller and Friedman [74], and Larrañaga and Moral [83].

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 combination 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 combination 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 $P(X_i = x_i | X_j = x_j)$ (or simply $P(x_i|x_j)$). Similarly, for continuous variables, the *joint density function* will be denoted as $p(\boldsymbol{x})$ and the *conditional density function* by $p(x_i|x_j)$. 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 $I(\boldsymbol{Y}, \boldsymbol{Z}|\boldsymbol{W})$), iff $\rho(\boldsymbol{y}|\boldsymbol{z}, \boldsymbol{w}) = \rho(\boldsymbol{y}|\boldsymbol{w})$, for all $\boldsymbol{y}$, $\boldsymbol{z}$ and $\boldsymbol{w}$.

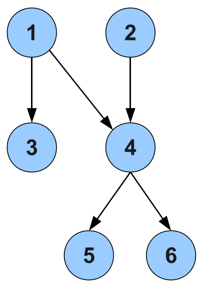
2.2. Bayesian network definition

A Bayesian network (BN) $\mathcal{B}(\mathcal{S}, \Theta)$ 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 independencies [30] between variables.
- A set of local parameters Θ representing the conditional probability distributions for the values of each variable given different value combinations of their parents, according to the structure $\mathcal{S}$.

Fig. 1a shows an example of a BN structure for a problem with six variables. For each variable X_i, $i = 1, \ldots, n$, structure $\mathcal{S}$ represents the assertion that X_i and its non-descendants, $ND(X_i)$, excluding its parents are conditionally independent given its parents, $\boldsymbol{Pa}_i$: i.e., $I(X_i, ND(X_i) \setminus \boldsymbol{Pa}_i | \boldsymbol{Pa}_i)$. This property is known as the Markov condition of BNs. Therefore, a BN encodes a factorization for the joint probability distribution of the variables

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



(a) Structure

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

(b) Conditional probability table

Fig. 1. An example of a Bayesian network structure and the parameters for one of its variables (X_4) assuming that $r_i = i + 1$.

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

A related notion in BNs is the so-called *Markov blanket* (MB) [107] of the variables. The MB of a variable in a BN consists of its parents, its children and the parents of its children (spouses). The important property of this subset is that a variable in the BN is only influenced by its MB. In other words, given its MB, a variable is conditionally independent of all other variables (excluding its MB): $I(X_i, \boldsymbol{X} \setminus MB(X_i)|MB(X_i))$.

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 $P_{\mathcal{B}}\left(x_i^k|\boldsymbol{pa}_i^j\right) \equiv \theta_{ijk}$ 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 computed as $q_i = \prod_{X_m \in \boldsymbol{Pa}_i} r_m$. The local parameters of the BN for the ith variable can be represented by $\boldsymbol{\theta}_i = ((\theta_{ijk})_{k=1}^{r_i})_{j=1}^{q_i}$. Fig. 1b shows an example of a conditional probability table for a discrete variable in a BN.

2.3. Bayesian networks in machine learning

2.3.1. Supervised learning

In recent years, there has been a sizable increase in published research using BNs for supervised classification tasks [82]. Bayesian classifiers compute the class value with the highest posterior probability (c^*) to be assigned to each configuration of predictor values ($x_1, \ldots, x_n$):

$$c^* = \arg\max_c P(C = c|X_1 = x_1, \ldots, X_n = x_n) = \arg\max_c \rho(X_1 = x_1, \ldots, X_n = x_n|C = c)P(C = c). \quad (2)$$

Different Bayesian classifiers can be obtained depending on the factorization of $\rho(X_1 = x_1, \ldots, X_n = x_n|C = c)$. Fig. 2 shows examples of some Bayesian classifiers. *Naïve Bayes* (NB) [94] (Fig. 1a) is the simplest Bayesian classifier. It is built on the assumption that the predictor variables are conditionally independent given the class value

$$\rho(x_1, \ldots, x_n|c) = \prod_{i=1}^{n} \rho(x_i|c). \quad (3)$$

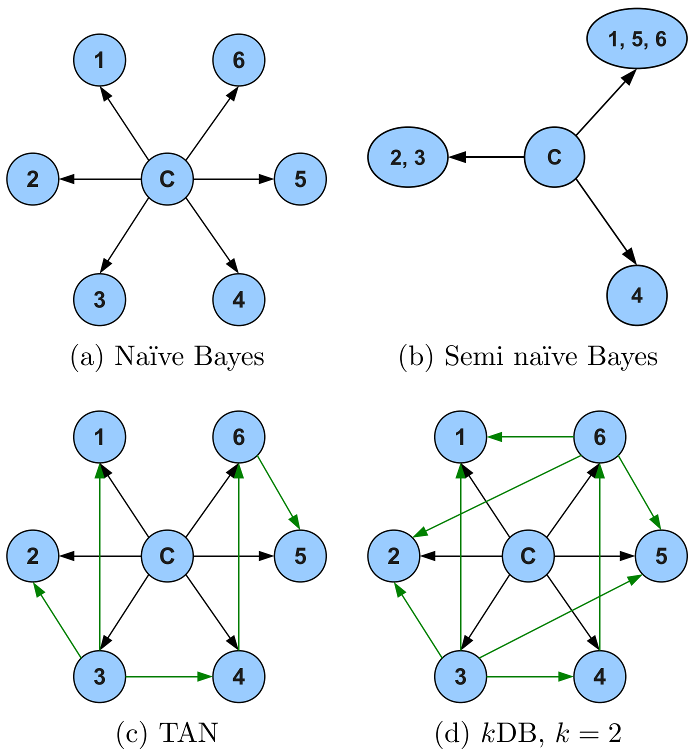


Fig. 2. Examples of different types of Bayesian classifier structures.

The *semi-naïve Bayes* (SNB) classifier [104] (Fig. 1b) considers a new variable type to avoid the conditional independence assumption of classical NB. These variables are formed by joining the original predictor variables, and their values are obtained from the Cartesian product of the values of the constituent variables. Pazzani [104] proposed a greedy wrapper approach for building a SNB classifier, where the irrelevant variables are removed from the model and the correlated variables are joined with their Cartesian product. The *tree augmented naïve Bayes* (TAN) classifier [43] (Fig. 1c) extends the structure of NB classifier by constructing a tree structure between predictor variables to account for their relationships. The *k-dependence Bayesian* (kDB) classifier [122] (Fig. 1d) also extends NB classifier with a more general structure allowing each variable to have *k* parents from the predictor variables. Bayesian classifiers can also be defined using the MB of the variables. Specifically, the MB of the class variable specifies the set of predictor variables affecting its posterior probability computation:

$$P(C|X_1,\ldots,X_n) = P(C|MB(C)). \tag{4}$$

2.3.2. Unsupervised learning

Another major area of machine learning employing BNs is *unsupervised learning* or *clustering*. The clustering of the data of an n-dimensional random variable $\boldsymbol{X} = (X_1, \ldots, X_n)$ should consider the structural constraint assumptions imposed by the data generation mechanism. In the case of BNs, the constraint states that there should be an edge from the random variable representing the cluster, C, to every predictor variable X_i. Thus, the factorization of the joint probability distribution for the $(n+1)$-dimensional random variable $(C, \boldsymbol{X})$ is given by

$$\rho_{\mathcal{B}}(c,\boldsymbol{x}) = P_{\mathcal{B}}(c)\prod_{i=1}^{n}\rho_{\mathcal{B}}(x_i|c,\boldsymbol{pa}_i). \tag{5}$$

Notice that this is similar to the factorization considered for BNs in supervised classification. The main difference, however, is that the value of variable C is unknown in clustering problems and has to be estimated using techniques like the *expectation–maximization* (EM) algorithm [33].

2.4. Inference in Bayesian networks

The BN paradigm is mainly used to reason in domains with intrinsic uncertainty. The reasoning inside the model, that is, the propagation of evidence through the model, depends on the structure reflecting the conditional independencies between the variables. Cooper [24] proved that this task is NP-hard in the general case of BNs with multiply connected structures. Generally speaking, the propagation of evidence involves assigning probabilities to the values of a subset of non-instantiated variables when the values of some other variables are known. The methods proposed for this task can be divided into two categories: (a) exact algorithms [86,107], and (b) approximate algorithms which include deterministic methods [13,38,65] and methods based on simulating samples from the BN [14,18,58,124]. For detailed information about these methods the reader can refer to [16,29,66].

Lauritzen and Spiegelhalter [86] proposed one of the most popular algorithms for exact inference. The first step of this algorithm is to moralize the network structure. In this step all variables with a common child are linked together and then all edge directions are removed. The resulting graph is called a moral graph. The second step of the algorithm is the so-called triangulation of the moral graph. A graph is triangulated if any cycle of length greater than 3 has a chord. This step is considered as the toughest step (in terms of computational complexity) of Lauritzen and Spiegelhalter's algorithm. The resulting structure is then used for evidence propagation and probability computation. For further explanation and details of this algorithm, see Lauritzen and Spiegelhalter [86].

The basic technique for triangulating a moral graph (see also Fig. 3) is through successive elimination of graph nodes. Before eliminating a node and its incident edges, we check that all of its adjacent nodes are directly connected to each other by adding the required edges to the graph. The nodes are chosen for elimination according to a given order of the variables. The quality of triangulation is measured by the weight of the triangulated graph $\mathcal{S}^t$

$$w(\mathcal{S}^t) = \log_2\left(\sum_{\boldsymbol{C}}\prod_{X_i\in\boldsymbol{C}} r_i\right), \tag{6}$$

where $\boldsymbol{C}$ denotes a maximal clique of the triangulated graph $\mathcal{S}^t$ composed of vertices X_i, each with r_i possible different states. The quality of triangulation is evidently fully determined by the order in which the nodes are eliminated. Hence, the search for an optimal triangulation is equivalent to the search for an optimal node elimination sequence, i.e., the search for an optimal permutation of nodes. Wen [136] demonstrated that the search for an optimal triangulation is NP-hard. Kjærulff [72] performed an empirical comparison of triangulation methods, obtaining the best results with the simulated annealing algorithm.

Instead of finding the probability of a subset of the variables in the BN, we sometimes need to find a value combination of these variables that results in the highest probability. The following two inference tasks are directly related to this requirement.

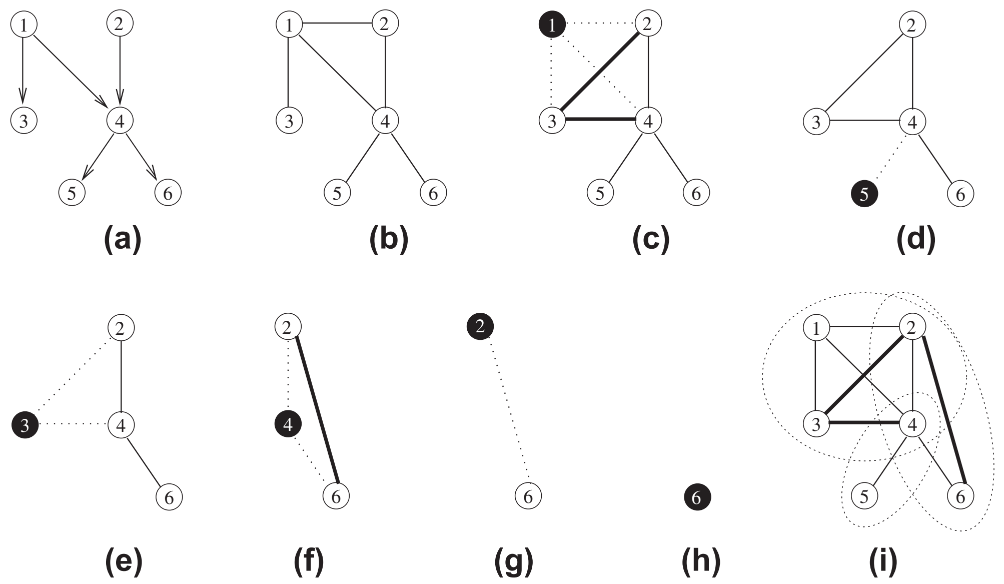


Fig. 3. An example of the triangulation algorithm. Nodes are eliminated in order: $X_1, X_5, X_3, X_4, X_2, X_6$ and it is assumed that $r_i = i + 1, i = 1, \ldots, 6$. (a) Initial DAG. (b) Related moral graph. (c) Eliminate X_1: $\mathbf{C}_1 = \{X_1, X_2, X_3, X_4\}$, added edges: $\{X_2, X_3\}, \{X_3, X_4\}$. (d) Eliminate X_5: $\mathbf{C}_2 = \{X_4, X_5\}$. (e) Eliminate X_3: $\mathbf{C}_3 = \emptyset$. (f) Eliminate X_4: $\mathbf{C}_4 = \{X_2, X_4, X_6\}$, added edges: $\{X_2, X_6\}$. (g) Eliminate X_2: $\mathbf{C}_5 = \emptyset$. (h) Eliminate X_6: $\mathbf{C}_6 = \emptyset$. (i) Total weight of the triangulated graph: $\log_2(2 \cdot 3 \cdot 4 \cdot 5 + 5 \cdot 6 + 3 \cdot 5 \cdot 7) = \log_2 255$.

2.4.1. Total abductive inference

Also known as the *most probable explanation* (MPE) problem [106], this type of inference finds the most probable value of each unobserved variable of the BN, given a set of observations ($\boldsymbol{X_O} = \boldsymbol{x_O}$). More formally, if $\boldsymbol{X_U} = \boldsymbol{X} \backslash \boldsymbol{X_O}$ is the set of unobserved variables, then the aim is to obtain the configuration $\boldsymbol{x_U}^*$ for $\boldsymbol{X_U}$ such that

$$\boldsymbol{x}_{\boldsymbol{U}}^{*} = \arg \max_{\boldsymbol{x_U}} \rho(\boldsymbol{x_U}|\boldsymbol{x_O}). \tag{7}$$

Searching for the MPE is just as complex (NP-hard) as probability propagation [125]. In fact, the MPE can be obtained by using probability propagation algorithms, where summation is replaced by a maximization operator in the final marginalization step [101].

2.4.2. Partial abductive inference

Also known as the *maximum a posteriori* (MAP) problem, this type of inference outputs the most probable configuration for just a subset of the variables in BN, known as the *explanation set*. If $\boldsymbol{X_E} \subset \boldsymbol{X_U}$ is used to denote the explanation set, then the aim is to obtain the configuration $\boldsymbol{x_E}^*$ for $\boldsymbol{X_E}$ such that

$$\boldsymbol{x}_{\boldsymbol{E}}^{*} = \arg \max_{\boldsymbol{x_E}} \rho(\boldsymbol{x_E}|\boldsymbol{x_O}). \tag{8}$$

This problem can be reformulated using an MPE problem, and marginalizing over all variables in $\boldsymbol{X_R} = \boldsymbol{X_U} \backslash \boldsymbol{X_E}$. Hence, finding the MAP is more complex than the MPE problem since it can have an intractable complexity (NP-hard) even for cases in which the MPE can be computed in polynomial time (e.g., polytrees) [103].

2.5. Learning Bayesian networks

The structure and conditional probabilities necessary for characterizing a BN can be provided either externally by experts, which is time consuming and prone to error, or by automatic learning from a database of samples. The task of learning a BN can be divided into two subtasks:

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

The different methods proposed for inducing a BN from a dataset are usually classified by modeling type into two approaches [10,28,56,100]:

1. methods based on detecting conditional independencies, also known as constraint-based methods, and
2. score+search methods.

2.5.1. Constrained-based methods

The input of these algorithms is a set of conditional independence relations between subsets of variables, which they use to build a BN that represents a large percentage (and, whenever possible, all) of these relations [129]. The PC algorithm [128] is a well-known example of these methods. Typically, hypothesis tests are used to find conditional independencies from a dataset. Once the structure has been learned, the conditional probability distributions, required to fully specify the BN 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.

2.5.2. Score+search methods

Constraint-based learning is quite an appealing approach as it is close to the semantics of BNs. 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 BN 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. In discrete domains, *Penalized maximum likelihood* is computed as follows:

$$P_{\mathcal{B}}(\mathcal{D}) = \prod_{i=1}^{n}\prod_{j=1}^{q_i}\prod_{k=1}^{r_i}\left(\frac{N_{ijk}}{N_{ij}}\right)^{N_{ijk}} - f(N)dim(\mathcal{B}), \tag{9}$$

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 combination 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. $dim(\mathcal{B})$ is the dimension (number of parameters needed to specify the model) of the BN. 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 BN can be computed as $dim(\mathcal{B}) = \sum_{i=1}^{n} q_i(r_i - 1)$. $f(N)$ is a non-negative penalization function depending on the size of the dataset. Popular scoring metrics like Akaike's information criterion (AIC) [1] and the Bayesian information criterion (BIC) [123] 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 BN, 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 [23,57]. 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 [57]:

$$P_{\mathcal{B}(\mathcal{S})}(\mathcal{D}) = \prod_{i=1}^{n}\prod_{j=1}^{q_i}\frac{\Gamma(\alpha_{ij})}{\Gamma(N_{ij} + \alpha_{ij})}\prod_{k=1}^{r_i}\frac{\Gamma(N_{ijk} + \alpha_{ijk})}{\Gamma(\alpha_{ijk})}, \tag{10}$$

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 (see below).

Minimum description length (MDL) score [50,114] is another type of scoring metric based on information theory and data compression. This score, which is justified by Occam's razor principle favoring less complex models, is closely related to the logarithm of the penalized maximum likelihood score. In simple terms this metric can be described as follows. Suppose that the cost of encoding a dataset $\mathcal{D}$ with a model $\mathcal{B}$ is equal to the cost of describing the model plus the cost of describing the data with this model: $Cost(\mathcal{B}) + Cost(\mathcal{D}|\mathcal{B})$. 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 BN 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.

Spaces and search methods. Most of the proposed score+search algorithms search the *space of DAGs*, which represent feasible BN structures. The number of possible structures in this space for an n-dimensional variable is given by the following recursive formula [115]:

$$\begin{aligned} f(n) &= \sum_{i=1}^{n}(-1)^{i+1}\binom{n}{i}2^{i(n-i)}f(n-i), \\ f(0) &= 1, \quad f(1) = 1. \end{aligned} \tag{11}$$

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 [19–21]. Therefore, greedy local search techniques like K2 algorithm [9,23], as well as many heuristic search methods such as simulated annealing [57], tabu search [8] and evolutionary computation (see Section 4) have been frequently employed for this purpose in the literature.

The K2 algorithm receives as input a total ordering of the variables which can have a big influence on its result. Thus, finding a good ordering of the variables is crucial for the algorithm success. On this ground, the *space of variable orderings* (permutations), rather than the space of DAGs, can be searched to obtain orderings that can result in higher-scoring networks.

Besides the previous two spaces, another possibility is to search the *space of equivalence classes* of BNs [22], when the scoring metric complies with the equivalence property. Two DAGs are said to be Markov equivalent if they encode the same statistical model, i.e., the same set of conditional independence statements. This model can be represented by a partial DAG (PDAG), where some of the edges are undirected. A metric that assigns equal scores to Markov equivalent BNs is said to comply with the equivalence property. Using this algebraic relation (which is reflexive, symmetric and transitive), the space of equivalence classes can be searched for the best BN. The BDe metric mentioned above is a Markov equivalence-compliant scoring metric.

3. Evolutionary algorithms

Over the last few decades several types of evolutionary algorithms (EAs), like genetic algorithms (GAs) [59], evolutionary strategies (ESs) [112], evolutionary programming (EP) [42] and genetic programming (GP) [27,75] have been proposed. They are considered as important meta-heuristic algorithms for solving many real-world problems. Fig. 4 shows the common framework of a typical evolutionary algorithm.

3.1. Genetic algorithms

GAs are perhaps the most well-known and widely used EAs. Since their introduction [59], they have received an increasing amount of attention and interest, and numerous works have studied their different aspects. A typical GA works by evolving a population of candidate solutions to the problem across a number of generations in order to obtain better solutions. Solutions are usually represented as binary strings, the same as the representation of information in machine language. The algorithm selects a subset of fitter solutions from the population according to a selection mechanism, e.g. tournament selection, as the parents. These parent solutions are used to reproduce new offspring solutions by applying genetic operators like crossover and mutation. The newly generated solutions then compete with the solutions in the population for survival according to their fitness.

The simple and easy to understand mechanism of GAs with their simple solution representation method, has led them to be heavily utilized for optimization in a vast variety of domains, from engineering tasks [46] to medicine [2]. They have been also extensively used in multi-objective [32], uncertain and dynamic [47] domains under the general term of evolutionary algorithms. Despite their simple mechanics, several works have also studied the performance of these algorithms from a theoretical point of view [53,60]. For further information on these algorithms see [48,49].

3.2. Genetic programming

The objective of GP is to evolve functions or computer programs to obtain a desired functionality. The main difference of GP and GA is in the way that the solutions are represented. The usual representation used to encode the solutions in GP is tree structures, where operations are shown as intermediate nodes and operands as terminal nodes of the tree. GP evolves a

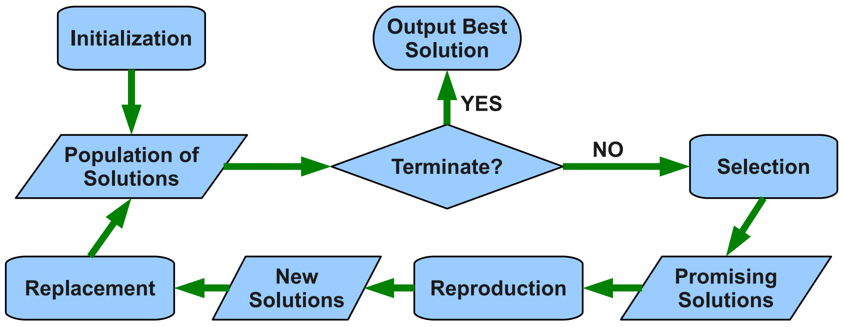


Fig. 4. Flowchart of a typical evolutionary algorithm.

population of these trees in the general framework of EAs, trying to generate programs that can better achieve the required functionality. In a broader perspective, GP can be used for automatic generation of new content.

To deal with program tree representation of solutions, the genetic operators used to reproduce new solutions should be adapted accordingly. Crossover usually involves switching the compatible branches of two solutions. In mutation the values of specific tree nodes or a branch of the tree is changed (while respecting the compatibility of the whole solution). Therefore, the solutions in the population can have different sizes. GP and EP are closely related and usually used interchangeably, with the latter putting more emphasis on mutation in the generation of new solutions. The interested reader is referred to the series of books by Koza [75–78].

3.3. Estimation of distribution algorithms

A relatively new paradigm in evolutionary computation is the use of probabilistic modeling in the EA framework for optimization. The resulting algorithms which are usually called estimation of distribution algorithms (EDAs) [81,96], replace the traditional reproduction mechanism of EAs, i.e. genetic operators, with probabilistic model estimation, followed by sampling individuals from the estimated model. Implicitly, EDAs assume that it is possible to model the promising areas of the search space, and use this model to guide the search for the optimum. The probabilistic model learnt in EDAs captures an abstract representation of the features shared by the selected solutions. Such a model can capture different patterns of interactions between subsets of the variables and can conveniently use this knowledge to sample new solutions.

Probabilistic modeling gives EDAs an advantage over other non-model based EAs by allowing them to deal with problems containing important interactions among the problem components. This, together with their capacity to solve different types of problems in a robust and scalable manner [90,108], has popularized these algorithms, which are sometimes even referred to as competent GAs [49,109] to differentiate them from traditional GA algorithms.

A common way of categorizing EDAs is according to the complexity of the probabilistic models they use. Based on this criterion, EDAs are usually classified as follows:

- Univariate EDAs: all variables are considered to be independent during model estimation. Some of the algorithms in this class include: compact GA (cGA) [54], population based incremental learning (PBIL) [3] and univariate marginal distribution algorithm (UMDA) [96].
- Bivariate EDAs: only mutual dependencies are considered between variables when estimating the probabilistic model. Mutual information maximizing input clustering (MIMIC) [31] and combining optimizers with mutual information trees (COMIT) [4] are examples of the algorithms in this class.
- Multivariate EDAs: the probabilistic model used in the algorithm, can potentially consider any number of dependencies between the variables. Factorized distribution algorithm (FDA) [95] and estimation of Bayesian network algorithm (EBNA) [39] are some of these algorithms.

It is important to note that the more flexible modeling offered in multivariate EDAs comes at the cost of a greater computational effort. Although probabilistic models were first built into GAs, the idea was soon adopted by other types of EAs, like GP.

3.4. Complementary methods

In order to improve the performance of EAs in optimization, several methods have been proposed which modify or add to the general framework of EAs. Here we briefly introduce two methods which are used in some of the works related to BN learning and inference, presented later on.

3.4.1. Hybridization

Hybridization of an algorithm usually refers to the case where this algorithm is used in conjunction with a different type of method, and thus can cover various types of hybridization between different algorithmic frameworks. The most likely type of hybridization for EAs, is to use a local search method for improving new solution reproduction, which is sometimes referred to as memetic algorithms [55]. In these algorithms, after generating a new solution using genetic operators, its local neighborhood is searched for fitter solutions using a local search method like hill climbing. This type of hybridization can improve the exploitation ability of EAs in search for optimal solution(s).

3.4.2. Cooperative coevolution

Co-evolutionary algorithms are an extension to the original EAs and are specially designed for optimization problems in complex systems. In coevolution, the fitness of each solution is determined by the way it interacts with other solutions in the population. Basically, two types of coevolution can be considered: competition and cooperation. In competitive coevolution [131], the increase in the fitness of an individual negatively affects the fitness of other solutions. Cooperative coevolution, on the other hand, rewards those solutions that have better collaboration with other solutions [111]. In this type of coevolution, usually the problem is decomposed into a number of subproblems and the individuals in the population represent solutions to these subproblems. Therefore these sub-solutions need to cooperate with each other to obtain complete solutions with higher fitness values. The sub-solutions can be either evolved in different populations or in a single population, known as Parisian approach [102].

4. Evolutionary algorithms in Bayesian network inference and learning

4.1. Triangulation of the moral graph

As mentioned in Section 2.4, the output of the triangulation algorithm used in the exact inference of BNs depends entirely on the order in which the graph nodes are eliminated. The problem of searching for an optimal node elimination sequence resembles the much researched traveling salesman problem (TSP). The aim of both problems is to find an optimal variable ordering. One important difference, however, is that only the relative order is important in the standard TSP, whereas the absolute order also matters in the node elimination problem. Taking these ideas, Larrañaga [80] applied a GA with crossover and mutation operators adapted for the TSP path representation. They achieved competitive results compared to simulated annealing, the best method to date [72].

More sophisticated recombination operators are a way to enhance the search for optimal variable ordering. Wang et al. [135] proposed an adaptive GA able to self-adapt the crossover and mutation operators probabilities, and provided a ranking-based selection operator that adjusts the pressure of selection according to the population evolution. Recently, Dong et al. [37] proposed a new GA based on a new rank-preserving crossover operator and a twofold mutation mechanism that utilizes the minimum fill weight heuristic.

Another alternative to improve search efficiency for this problem has been to use probabilistic modeling. Romero and Larrañaga [119] proposed an approach based on recursive EDAs (REDAs) for both discrete and continuous representation of the variables. REDAs partition the set of vertices (that are to be ordered) into two subsets. In each REDA call, the vertices in the first subset are fixed, whereas the other subset of variables is evolved with a standard EDA. In the second call, the subsets switch roles.

Several criteria are proposed for searching for the optimal node elimination order, from which most of the works try to minimize the weight of the corresponding triangulated graph (Eq. (6)). According to the abovementioned works, GAs can obtain results comparable with simulated annealing, for which very good results have been reported. A very close behavior is seen when using REDAs, with improved convergence speed. The comparison with other types of optimization algorithms that use other optimization criteria also show that GAs minimizing graph weight can find better node elimination orders, provided that proper operators and parameters are used.

4.2. Total and partial abductive inference

EAs have also been used to search for the MPE in a BN. Gelsema [45] used a GA where each individual is a configuration of the unobserved variables, i.e., a string of integers. Rojas-Guzmán [118] employed a GP where each individual represents the whole BN with all the nodes in the explanation set instantiated to one of their possible states. Mengshoel [93] used a GA coupled with his proposed probabilistic crowding replacement to perform a more efficient search for the MPE. Sriwachirawat and Auwatanamongkol [130] proposed a GA for solving the more complex problem of finding the *k* MPEs [101].

de Campos et al. [12] proposed a GA for approximate partial abductive inference (MAP) given an evidence set. The individuals in the GA population represent a possible configuration only for the variables in the explanation set (a subset of unobserved variables). The proposed algorithm is also able to find the *k* MPEs of the explanation set. Discrete EDAs with different degrees of model complexity (UMDA, MIMIC and EBNA) are also used to find the MAP [11].

The common trend in finding the *k* MPEs is to search in the space of possible value combinations for unobserved variables. The reported results show that if these value combinations are represented on the original BN structure, better results can be obtained with evolutionary search in less time. GAs can reach high probable explanations faster than conventional methods

Table 1
Application of evolutionary algorithms to inference in Bayesian networks.

Task	Reference	Representation	Algorithm
Triangulation	Larrañaga et al. [80]	Permutation of variables	GA
	Wang et al. [135]	Permutation of variables	GA
	Romero and Larrañaga [119]	Permutation of variables	REDA
	Dong et al. [37]	Permutation of variables	GA
MPE	Gelsema [45]	Value combination for variables	GA
	Rojas-Guzmán and Kramer [118]	Graph	GP
	Mengshoel [93]	Value combination for variables	GA
	Sriwachirawat and Auwatanamongkol [130]	Value combination for variables	GA
MAP	de Campos et al. [12]	Value combination for variables	GA
	de Campos et al. [11]	Value combination for variables	UMDA, MIMIC, EBNA

like max flow propagation [101]. Furthermore, the probabilistic modeling of EDAs can speed up the convergence compared to GA, especially when using probabilistic models with high descriptive abilities (e.g. EBNA) [11]. Table 1 summarizes the algorithms for some BN inference tasks.

4.3. Structure search in Bayesian network learning

Finding the correct BN structure is an important part of the learning process which also directly affects BN parameter learning. Heuristic search algorithms and especially EAs can be a promising approach to this problem as the cardinality of search space is huge. The reviewed methods are divided into three categories depending on the space where they perform the search for finding the best network. The methods are also listed in Table 2.

4.3.1. DAG space

Larrañaga et al. [85] proposed a GA that encodes the connectivity matrix of the BN structure in its individuals. The algorithm, which uses a marginal likelihood metric to score the network structures, considers two different approaches. In the first approach there is a total ordering assumption between the variables (parents before children), and thus the variation operators (one-point crossover and bit mutation) are closed operators. This reduces the cardinality of the search space. In the second approach, there is no such assumption, and the algorithm should deal with a larger space. In this case, a repairing operator is needed to ensure that the variation operators are closed.

To overcome the requirement for a repairing operator, [40] used the fuse-DAGs algorithm [92] to guarantee that the crossover operator satisfies the closure property. Larrañaga et al. [84] hybridized two versions of a GA with a local search operator to obtain better structures. Myers et al. [99] extended the use of GAs for BN learning to domains with missing data, simultaneously evolving the structure of the BN and the missing data in separate populations. At each generation the new solutions generated in both populations are used to compute the BDe score of each network structure.

cotta and Muruzábal [25] built phenotypic information into gene-based and allele-based recombination operators in a GA to search for the best structure according to a penalized marginal likelihood scoring metric. Using guidelines on how GAs work [53] and van Dijk et al. 35] designed a GA where the recombination operator tries to prevent the disruption of the good BN substructures obtained so far in the population. The algorithm uses an MDL metric as the fitness function for scoring the network structures, and a repairing operator to ensure that structures are acyclic.

Blanco et al. [6] compared the performance of GAs with two univariate EDAs, namely, UMDA and PBIL, using three different scoring metrics. The reported results, both with and without a total ordering assumption between variables, showed that EDAs are able to obtain better or comparable network structures. Kim et al. [71] used fitness sharing in an EA to obtain a

Table 2
Application of evolutionary algorithms to learning Bayesian networks.

Space	Reference	Representation	Algorithm
DAGs	Larrañaga et al. [85]	Connectivity matrix	GA
	Larrañaga et al. [84]	Connectivity matrix	GA+local search
	Etxeberria et al. [40]	Connectivity matrix	GA
	Myers et al. [99]	Connectivity matrix	GA
	Wong et al. [137]	Graph	EP
	Tucker et al. [134]	Edge-time tuples	EP
	Cotta and Muruzãbal [25]	Connectivity matrix	GA
	van Dijk et al. [35]	Connectivity matrix	GA
	Blanco et al. [6]	Connectivity matrix	GA, PBIL, UMDA
	Tucker et al. [133]	Set of spatial points	GA
	Wong and Leung [139]	Connectivity matrix	EP
	Wong et al. [138]	Connectivity matrix	Cooperative coevolution
	Kim et al. [71]	Connectivity matrix	GA
	Mascherini and Stefanini [91]	Connectivity matrix	GA
	Jia et al. [68]	String of possible parents	Immune GA
	Ross and Zuviria [121]	String of possible parents	Multiobjective GA
	Hanzelka [52]	Connectivity matrix	GA+local search
	Barrie~re et al. [5]	Connectivity matrix	Cooperative coevolution
PDAGs	Muruzãbal and Cotta [98]	Graph	EP
	Cotta and Muruzãbal [26]	Graph	EP
	van Dijk and Thierens [34]	Connectivity matrix	GA+local search
	Jia et al. [67]	Connectivity matrix	Immune GA
Orderings	Larrañaga et al. [79]	Permutation	GA
	Habrant [51]	Permutation	GA
	Hsu et al. [61]	Permutation	GA
	Romero et al. [120]	Permutation	UMDA, MIMIC
	Kabli et al. [69]	Chain permutation	GA
	Lee et al. [88]	Permutation+connectivity matrix	GA

diverse population of BN structures. The BNs learnt at the end of evolution are then combined according to Bayes' rule for providing a more robust inference.

Hanzelka [52] also proposed a hybridization of GA with local search methods performed on single solutions under the term of *Lamarckian evolution*. It uses a Chi-squared test to determine the edge that should be removed for repairing the structure. After GA terminates, an exhaustive search is conducted in the most promising search subspace obtained.

Barrière [5] proposed an EA which uses a cooperative co-evolution strategy to evolve a population of conditional independence assertions. The scoring criteria is the Chi-squared test. At the end of evolution, the best conditional independence assertions found (partly stored in an archive) are used to build the structure of the BN.

The EP algorithm proposed by Wong et al. [137] is based on a set of mutation operators and uses the MDL metric to search for good BN structures. Because of its flexibility in representing the structures without any encoding, no further assumptions on the ordering of the variables are needed in order to apply the mutation operators. The proposed algorithm is extended by first introducing a merge operator and then hybridizing it using the two-phase constraint-based method [139]. In the dependency analysis phase, conditional independencies among the variables are used to reduce the size of the DAG search space. In the search phase, good BN models are generated using an EA. Replacement of the EA with a cooperative co-evolution algorithm is also studied in [138].

Most of the works that consider learning BNs by searching in the space of possible DAG structures use a string representation of the connectivity matrix. In this representation the order of variables is important or else a repair operator will be necessary to ensure valid DAG structures after applying genetic operators. Because of this, some methods simultaneously search for variable orderings and topology of BN whereas some others use structure-aware operators to ensure the validity of the resulting DAGs. A similar representation is the list of parents of each variable, leading to solutions of varying sizes. If GP is used, DAG structures can be directly evolved and the reported results show better performance of this approach, in terms of the final structure score, its closeness to the original structure and computational time needed for the search [137].

Another point is the importance of local search or in general higher exploitation which is shown to result in better BN structures. Significant improvement has been reported when the initial search space is reduced by incorporating information about the conditional independencies between variables [139]. These information is gathered in a pre-evolution phase by performing conditional independence tests, usually with small order to keep the computational complexity of the whole algorithm small. Comparison of different EAs with some standard methods like K2 algorithm or simple deterministic methods like hill climbing show that, especially when the size of the learning datasets increases, EAs are able to estimate better structures and usually have a faster convergence.

4.3.2. Equivalence class space

To eliminate redundancy in the DAG space, van Dijk and Thierens [34] extended their initial representation to PDAGs to perform the search in the equivalence class space. They also studied the effect of hybridizing the algorithm with local search methods. Jia et al. [67] proposed an immune GA, hybridizing principles of artificial immune systems (based on immunology) [17] with GAs, to search this space. They employed conditional independence tests for extracting variables independence information prior to the evolutionary search of GA and use it (as immune vaccines) in order to reduce the search space.

Muruzábal and Cotta [98] proposed an EP algorithm to perform the search in the equivalence class space. The algorithm uses some mutation operators to move between Markov equivalent classes [22] according to a BDe metric. Cotta and Muruzábal [26] compared three versions of EP algorithms that perform the search in the equivalence class space, either directly or with a restriction on the operators (inclusion boundary [15]).

Two milestone works have paved the way for most other approaches proposed for searching in equivalence space: the equivalence class aware operators that allow moving between different classes when applied to any PDAG member of that class; and the inclusion boundary property of the operators which when preserved can prevent the search from falling into local optima. The greedy search in this space results in faster convergence compared with searching in the DAG space. However, the size of search space is still exponential in the number of variables. Many of the EAs that are proposed for performing the search in this space involve converting back and forth PDAGs to DAGs which is a computationally expensive operation. Hybridizing EAs with local search has been reported to improve the results [34].

4.3.3. Ordering space

Larrañaga et al. [79] used the TSP-inspired permutation representation (Section 4.1) to search for the best ordering between the variables with a GA. The K2 algorithm was applied on each ordering to evaluate the quality of different orderings. They compared the performance of different combinations of crossover and mutation operators. Using the same representation and evaluation scheme, Habrant [51] proposed improved mutation and crossover operators to search for the best BN structure in the real-world problem of time series prediction in finance. Similarly, Hsu et al. [61] proposed a GA based on the *order crossover* operator to search in the permutation space. The fitness of each BN obtained by the K2 algorithm from an ordering is measured according to its inference quality (using cross-validation).

The chainGA [69] assumes a chain structure between the variables in the ordering which it evaluates using the K2 metric in order to bypass the need for the time-consuming K2 algorithm. At the end of evolution however, the K2 algorithm is applied to the best orderings to obtain a good structure. The algorithm is also applied to the real-world problem of prostate cancer management [70]. Lee et al. [88] proposed a novel representation of BN structure composed of *dual* chromosomes: a node ordering chromosome and a connectivity matrix chromosome according to its dual (ordering). They applied the proposed GA, with the special crossover and mutation operators developed for this representation, to a number of real-world problems involving learning BNs.

Romero et al. [120] applied two types of discrete- and continuous-encoded EDAs (UMDA and MIMIC) to obtain the best ordering for the K2 algorithm. For discrete encoding they used a bijective mapping to represent possible orderings of n variables with $n-1$ random variables. The simulation step is adapted in order to output a valid permutation of the variables. This adaptation is not necessary for continuous encoding, where each n-dimensional real vector can be transformed into a valid permutation of the n variables.

An important decision to make when performing the search in the space of orderings is how to evaluate different candidate orderings. Some of the proposed methods use the K2 algorithm for this purpose which results in a high fitness evaluation cost. Approximating the quality of a solution with a less computationally expensive method, can greatly reduce the overall running time of the algorithms (e.g. as in chainGA). However, this can also cause the quality of the learned BNs to reduce. The reported results on datasets with small number of variables show that the evolutionary search with GA obtains results comparable to those of exhaustive search of all possible orderings, while only visiting a small percentage of the whole solution space.

4.4. Learning other types of Bayesian networks

BNs have also been used for reasoning in continuous domains. In this type of domains it is usually assumed that the vector of variables follow a Gaussian distribution, and therefore the resulting BN is called a *Gaussian Bayesian network* (GBN) [44]. GBNs differ from discrete BNs only in the way they represent the parameters. A further extension is to use BNs for domains with mixed variables, i.e. containing both discrete and continuous variables. The resulting BN is called a *conditional Gaussian Bayesian network* (CGN) [87]. Special precautions have to be taken when dealing with domains consisting of both discrete and continuous variables to ensure certain conditions for the structure and parameters of the learned CGN. In this domain, Mascherini and Stefanini [91] proposed a mixed GA to search for the optimal CGN, where invalid structures are corrected by deleting inadmissible arcs at random. An extension of the BDe metric is used to measure the fitness of the model for the mixed domain dataset.

In another application, BNs have been used to model time series data. Basically, a natural choice for modeling time series data is to use directed graphical models which can appropriately capture the forward flow of time. If all arcs of the model are directed, both within and between different time slices while the structure is unchanged, the resulting model is called a *dynamic Bayesian network* (DBN) [97]. Several works have used EAs to learn DBN structures from data. Tucker et al. [134] use an EP algorithm to seed the population of a GA that evolves over the structures of DBNs. Zuviria [121] use a multi-objective GA, where the multi-objective criteria are network likelihood and structural complexity scores. Tucker et al. [133] propose evolutionary and non-evolutionary operators for learning BN structures from spatial time series data. Jia et al. [68] apply their immune GA for learning DBNs.

Many of the methods proposed for learning the structure of normal BNs can be adapted to learn DBNs. Simplifying assumptions, e.g. no edges between the nodes in the same time slice, can greatly reduce the computational complexity of learning DBNs. The BNs learned with simple GAs are not so satisfactory, sometimes worse than hill climbing. Because of this, complementary techniques have been employed in the reported works, including generating non-random initial population with GP, incorporation of additional operators in the recombination process, like the add vaccine of immune GA, and performing a multi-objective search instead of single objective to take into account several criteria when learning DBNs. All of these complementary techniques have been reported to yield better DBNs in terms of the scoring metric or structural accuracy compared with simple GA.

4.5. Learning Bayesian network classifiers

Finding an appropriate subset of predictor variables by removing redundant and irrelevant variables can be remarkably helpful for classification using BNs. EAs are one of the search techniques extensively used for this task, which is usually called *feature subset selection* (FSS). Liu et al. [89] used GAs to search for an optimal subset of predictor variables for their improved NB classifier, whereas Inza et al. [62] applied EBNA for FSS in a number of different datasets. They also compared the proposed method with two GA-based and two greedy search algorithms [63]. PBIL and a dependency tree-based EDA (COMIT) are used for FSS in the problem of predicting the survival of cirrhotic patients, and the results are compared with two versions of a GA [64]. Blanco et al. [7] used EDAs for gene selection in cancer classification problem using a NB classifier.

The reported results show that both GA and EDA versions perform better than simple deterministic hill climbing algorithms like forward selection and backward elimination. A comparison between EDAs also show that using more powerful probabilistic models allow selecting better feature subsets which for many of the tested data sets are also better than the feature subsets found by GA.

Robles et al. [116] used EDAs in their interval estimation NB classifier to search for proper class and conditional probabilities within their respective confidence intervals. An EDA with a continuous representation is used to search for the best combination of probability values in these intervals. They also used EDAs to improve the search for new hybrid variables in the SNB classifier [117]. A comparison with standard techniques like forward selection and joining of variables, or backward

Table 3
Learning Bayesian classifiers with evolutionary algorithms.

	Reference	Classifier	Algorithm
FSS	Inza et al. [62]	Naïve Bayes	EBNA
	Liu et al. [89]	Naïve Bayes	GA
	Inza et al. [63]	Naïve Bayes	EBNA, GA, greedy search
	Inza et al. [64]	Naïve Bayes	PBIL, COMIT, GA
	Blanco et al. [7]	Naïve Bayes	EDA
Classification	Sierra and Larrañaga [126]	Markov blanket	GA
	Sierra et al. [127]	Markov blanket	GA
	Robles et al. [117]	Naïve Bayes, semi-Naïve Bayes	EDA
	Kline et al. [73]	General Bayesian network	GA
	Flores et al. [41]	Naïve Bayes	$UMDA_C$
	Zhu et al. [140]	Markov blanket	GA
	Reiz et al. [113]	TAN	GA
	Dong et al. [36]	TAN	GA
	Peña et al. [110]	Bayesian network	UMDA

elimination and joining of variables [104] show that EDA-based search and joining of variables can find significantly better classifiers, though at a higher computational complexity.

Flores et al. [41] proposed the use of $UMDA_C$ to search for the optimal discretization of all predictor variables simultaneously for the NB model. Reiz et al. [113] employed Prüfer numbers to encode TAN Bayesian classifiers and search for the optimal structure using GAs. AIC, BIC and Hannan-Quinn information criteria were employed as fitness measures. Dong et al. [36] designed genetic operators to evolve TAN structures with the objective of maximizing the likelihood function.

Sierra and Larrañaga [126] used GAs to search for the optimal MB of the class variable for a real-world classification problem. They compared the resulting MB-based classifiers with NB classifier and a Bayesian classifier learned by likelihood maximization and show that the MB-based classifiers have higher classification accuracy. This method was employed in a multi-classifier schema to classify intensive care unit patient data [127]. Zhu et al. [140] proposed a MB-embedded GA for gene selection in microarray datasets and showed that using GA to search for the MB of the class variable results in higher classification accuracy. Kline et al. [73] showed the use of GAs to search for the most accurate BN structure to predict venous thromboembolism according to gathered data.

Also in the field of BN classification, Peña [110] applied UMDA to search for the optimal dependency structure between predictor variables in unsupervised learning using a specific representation of BNs. Table 3 shows these methods along with the classifiers and EAs they used.

5. Conclusions

Bayesian networks are an important class of probabilistic graphical models, which have proven to be very useful and effective for reasoning in uncertain domains. They have been successfully used in machine learning tasks like classification and clustering. They are studied at length over the last three decades and many methods have been proposed to automate their learning and inference. Nevertheless, many of these methods involve difficult combinatorial search problems that directly affects their widespread use, especially for large problem instances, and thus require advanced search techniques like meta-heuristics.

Evolutionary algorithms, are general-purpose stochastic search methods inspired from natural evolution and have been frequently applied to solve many complex real-world problems. Different types of solutions from bit strings to program trees can be evolved within this framework in search for better solutions. A relatively new type of these algorithms, estimation of distribution algorithms, uses probabilistic modeling (and possibly Bayesian networks) to capture problem regularities and use them for new solution generation. They have been shown to solve problems that are considered challenging for traditional evolutionary algorithms.

Because of their advantages, different types of evolutionary algorithms have been used in Bayesian networks learning and inference tasks. A wide range of tasks like triangulation of the moral graph in Bayesian network inference, abductive inference, Bayesian network structure learning in difference search spaces, Bayesian classifier learning and learning dynamic Bayesian networks from stream data have employed evolutionary algorithms, which has led to significant improvements in the computational time and performance.

This topic is still an active field of research and with the intrinsic complexity of Bayesian network tasks, evolutionary algorithms are always a potential competitor. Especially, estimation of distribution algorithms with their ability to account for the interactions between variables seem to be a promising approach for further study. So far, several works have empirically compared the conventional approaches to Bayesian network tasks (see for example [132] for a comparison between several Bayesian network learning methods). An interesting future work that can complement this review is to perform similar empirical comparison of the evolutionary approaches presented here, on standard datasets.

Acknowledgements

This work has been partially supported by TIN2010-20900-C04-04, TIN-2010-14931, Consolider Ingenio 2010-CSD 2007-00018, and Cajal Blue Brain projects (Spanish Ministry of Science and Innovation), the Saiotek and Research Groups 2007–2012 (IT-242-07) programs (Basque Government), and COMBIOMED network in computational biomedicine (Carlos III Health Institute).

References

[1] H. Akaike, A new look at the statistical model identification, IEEE Transactions on Automatic Control 19 (1974) 716–723.
[2] J.T. Alander, An Indexed Bibliography of Genetic Algorithms in Medicine, Technical Report 94-1-MEDICINE, University of Vaasa, Finland, 2012.
[3] S. Baluja, Population-Based Incremental Learning: A Method for Integrating Genetic Search Based Function Optimization and Competitive Learning, Technical Report CMU-CS-94-163, Carnegie-Mellon University, Pittsburgh, PA, 1994.
[4] S. Baluja, S. Davies, Using optimal dependency-trees for combinational optimization, in: 14th International Conference on Machine Learning, Morgan Kaufmann Publishers Inc., 1997, pp. 30–38.
[5] O. Barrière, E. Lutton, P.H. Wuillemin, Bayesian network structure learning using cooperative coevolution, in: 11th Annual Conference on Genetic and Evolutionary Computation (GECCO '09), ACM, New York, NY, USA, 2009, pp. 755–762.
[6] R. Blanco, I. Inza, P. Larrañaga, Learning Bayesian networks in the space of structures by estimation of distribution algorithms, International Journal of Intelligent Systems 18 (2003) 205–220.
[7] R. Blanco, P. Larrañnaga, I. Inza, B. Sierra, Gene selection for cancer classification using wrapper approaches, International Journal of Pattern Recognition and Artificial Intelligence 18 (2004) 1373–1390.
[8] R.R. Bouckaert, Bayesian Belief Networks: From Construction to Inference, Ph.D. thesis, Universiteit Utrecht, Faculteit Wiskunde en Informatica, 1995.
[9] W. Buntine, Theory refinement on Bayesian networks, in: B. D'Ambrosio, P. Smets (Eds.), 7th Annual Conference on Uncertainty in Artificial Intelligence (UAI '91), Morgan Kaufmann, San Francisco, CA, USA, 1991, pp. 52–60.
[10] W. Buntine, A guide to the literature on learning probabilistic networks from data, IEEE Transactions on Knowledge and Data Engineering 8 (1996) 195–210.
[11] L.M. de Campos, J.A. Gámez, P. Larrañaga, S. Moral, T. Romero, Partial abductive inference in Bayesian networks: an empirical comparison between GAs and EDAs, in: [81], 2001, pp. 323–341.
[12] L.M. de Campos, J.A. Gámez, S. Moral, Partial abductive inference in Bayesian belief networks using a genetic algorithm, Pattern Recognition Letters 20 (1999) 1211–1217.
[13] A. Cano, S. Moral, A. Salmerón, Novel strategies to approximate probability trees in penniless propagation, International Journal of Intelligent Systems 18 (2003) 193–203.
[14] G. Casella, E.I. George, Explaining the gibbs sampler, The American Statistician 46 (1992) 167–174.
[15] R. Castelo, T. Kočka, On inclusion-driven learning of Bayesian networks, Journal of Machine Learning Research 4 (2003) 527–574.
[16] E. Castillo, J.M. Gutierrez, A.S. Hadi, Expert Systems and Probabilistic Network Models, Springer, 1997.
[17] L.N. de Castro, J. Timmis, Artificial Immune Systems: A New Computational Intelligence Approach, Springer, 2002.
[18] S. Chib, E. Greenberg, Understanding the Metropolis-Hastings algorithm, The American Statistician 49 (1995) 327–335.
[19] D. Chickering, Learning Bayesian networks is NP-complete, in: D. Fisher, H.J. Lenz (Eds.), Learning from Data: Artificial Intelligence and Statistics V, Lecture Notes in Statistics, vol. 112, Springer, 1996, pp. 121–130.
[20] D. Chickering, D. Geiger, D. Heckerman, Learning Bayesian Networks is NP-hard, Technical Report MSR-TR-94-17, Microsoft Research, Redmond, WA, USA, 1994.
[21] D. Chickering, D. Heckerman, C. Meek, Large-sample learning of Bayesian networks is NP-hard, Journal of Machine Learning Research 5 (2004) 1287–1330.
[22] D.M. Chickering, Learning equivalence classes of Bayesian-network structures, Journal of Machine Learning Research 2 (2002) 445–498.
[23] G. Cooper, E. Herskovits, A Bayesian method for the induction of probabilistic networks from data, Machine Learning 9 (1992) 309–347.
[24] G.F. Cooper, The computational complexity of probabilistic inference using Bayesian belief networks, Artificial Intelligence 42 (1990) 393–405.
[25] C. Cotta, J. Muruzábal, Towards a more efficient evolutionary induction of Bayesian networks, in: 7th International Conference on Parallel Problem Solving from Nature (PPSN VII), Springer-Verlag, London, UK, 2002, pp. 730–739.
[26] C. Cotta, J. Muruzábal, On the learning of Bayesian network graph structures via evolutionary programming, in: Second European Workshop on Probabilistic Graphical Models, 2004, pp. 65–72.
[27] N.L. Cramer, A representation for the adaptive generation of simple sequential programs, in: First International Conference on Genetic Algorithms, L. Erlbaum Associates Inc., Hillsdale, NJ, USA, 1985, pp. 183–187.
[28] R. Daly, Q. Shen, J.S. Aitken, Learning Bayesian networks: approaches and issues, Knowledge Engineering Review 26 (2011) 99–157.
[29] A. Darwiche, Modeling and Reasoning with Bayesian Networks, Cambridge University Press, 2009.
[30] A.P. Dawid, Conditional independence in statistical theory, Journal of the Royal Statistical, Society Series B (Methodological) 41 (1979) 1–31.
[31] J. De Bonet, C. Isbell, P. Viola, MIMIC: Finding optima by estimating probability densities, in: Advances in Neural Information Processing Systems, volume 9, 1997, pp. 424–430.
[32] K. Deb, Multi-Objective Optimization using Evolutionary Algorithms, John Wiley & Sons, Inc., New York, NY, USA, 2001.
[33] A.P. Dempster, N.M. Laird, D.B. Rubin, Maximum likelihood from incomplete data via the EM algorithm, Journal of the Royal Statistical Society, Series B (Methodological) 39 (1977) 1–38.
[34] S. van Dijk, D. Thierens, On the use of a non-redundant encoding for learning Bayesian networks from data with a GA, in: X. Yao, E. Burke, J.A. Lozano, J. Smith, J.J. Merelo-Guervós, J.A. Bullinaria, J. Rowe, P. Tino, A. Kabán, H.P. Schwefel (Eds.), 8th International Conference on Parallel Problem Solving from Nature (PPSN VIII), Lecture Notes in Computer Science, vol. 3242, Springer, 2004, pp. 141–150.
[35] S. van Dijk, D. Thierens, L. van der Gaag, Building a GA from design principles for learning Bayesian networks, in: E. Cantú-Paz, J. Foster, K. Deb, L. Davis, R. Roy, U.M. O'Reilly, H.G. Beyer, R. Standish, G. Kendall, S. Wilson, M. Harman, J. Wegener, D. Dasgupta, M. Potter, A. Schultz, K. Dowsland, N. Jonoska, J. Miller (Eds.), Fifth Annual Conference on Genetic and Evolutionary Computation (GECCO '03): Part I, Lecture Notes in Computer Science, vol. 2723, Springer, 2003, pp. 886–897.
[36] L. Dong, H. Zhang, X. Ren, Y. Li, Classifier learning algorithm based on genetic algorithms, International Journal of Innovative Computing, Information and Control 6 (2010) 1973–1981.
[37] X. Dong, D. Ouyang, Y. Ye, S. Feng, H. Yu, A stable stochastic optimization algorithm for triangulation of Bayesian networks, in: Third International Conference on Knowledge Discovery and Data Mining (WKDD '10), 2010, pp. 466–469.
[38] R. van Engelen, Approximating Bayesian belief networks by arc removal, IEEE Transactions on Pattern Analysis and Machine Intelligence 19 (1997) 916–920.
[39] R. Etxeberria, P. Larrañaga, Global optimization using Bayesian networks, in: Second Symposium on Artificial Intelligence (CIMAF-99), 1999, pp. 332–339.
[40] R. Etxeberria, P. Larrañaga, J.M. Picaza, Analysis of the behaviour of genetic algorithms when learning Bayesian network structure from data, Pattern Recognition Letters 18 (1997) 1269–1273.

[41] J.L. Flores, I. Inza, P. Larrañaga, Wrapper discretization by means of estimation of distribution algorithms, Intelligent Data Analysis 11 (2007) 525–545.
[42] L.J. Fogel, Artificial Intelligence Through Simulated Evolution, Wiley, New York, 1966.
[43] N. Friedman, D. Geiger, M. Goldszmidt, Bayesian network classifiers, Machine Learning 29 (1997) 131–163.
[44] D. Geiger, D. Heckerman, Learning Gaussian networks, in: 10th Conference on Uncertainty in Artificial Intelligence (UAI'94), 1994, pp. 235–243.
[45] E.S. Gelsema, Abductive reasoning in Bayesian belief networks using a genetic algorithm, Pattern Recognition Letters 16 (1995) 865–871.
[46] M. Gen, R. Cheng, Genetic Algorithms and Engineering Optimization, Springer, 2000.
[47] C.K. Goh, K.C. Tan, Evolutionary Multi-objective Optimization in Uncertain Environments: Issues and Algorithms, Studies in Computational Intelligence, vol. 186, Springer, 2009.
[48] D.E. Goldberg, Genetic Algorithms in Search, first ed., Optimization and Machine Learning, Addison-Wesley Longman Publishing Co. Inc., Boston, MA, USA, 1989.
[49] D.E. Goldberg, The Design of Innovation: Lessons from and for Competent Genetic Algorithms, Kluwer Academic Publishers, Norwell, MA, USA, 2002.
[50] P. Grünwald, The Minimum Description Length Principle and Reasoning Under Uncertainty, Ph.D. thesis, University of Amsterdam, 1998.
[51] J. Habrant, Structure learning of Bayesian networks from databases by genetic algorithms: application to time series prediction in finance, in: First International Conference on Enterprise Information Systems (ICEIS), vol. 1, 1999, pp. 225–231.
[52] D. Hanzelka, The use of hybrid genetic algorithms in Bayesian network structure learning from data, Journal of Applied Mathematics 1 (2008) 387–396.
[53] G. Harik, E. Cantú-Paz, D. Goldberg, B. Miller, The gambler's ruin problem, genetic algorithms, and the sizing of populations, Evolutionary Computation 7 (1999) 231–253.
[54] G. Harik, F. Lobo, D. Goldberg, The compact genetic algorithm, IEEE Transactions on Evolutionary Computation 3 (1999) 287–297.
[55] W. Hart, N. Krasnogor, J. Smith, Memetic evolutionary algorithms, Recent Advances in Memetic Algorithms 166 (2005) 3–27.
[56] D. Heckerman, A tutorial on learning with Bayesian networks, in: NATO Advanced Study Institute on Learning in Graphical Models, Kluwer Academic Publishers, 1998, pp. 301–354.
[57] D. Heckerman, D. Geiger, D. Chickering, Learning Bayesian networks: the combination of knowledge and statistical data, Machine Learning 20 (1995) 197–243.
[58] M. Henrion, Propagating uncertainty in Bayesian networks by probabilistic logic sampling, in: J.F. Lemmer, L.N. Kanal (Eds.), Second Annual Conference on Uncertainty in Artificial Intelligence (UAI '86), vol. 2, Elsevier, 1986, pp. 149–163.
[59] J. Holland, Adaptation in Natural and Artificial Systems, University of Michigan Press, 1975.
[60] J.H. Holland, Adaptation in Natural and Artificial Systems, MIT Press, Cambridge, MA, USA, 1992.
[61] W.H. Hsu, H. Guo, B.B. Perry, J.A. Stilson, A permutation genetic algorithm for variable ordering in learning Bayesian networks from data, in: Conference on Genetic and Evolutionary Computation (GECCO '02), Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2002, pp. 383–390.
[62] I. Inza, P. Larrañaga, R. Etxeberria, B. Sierra, Feature subset selection by Bayesian network-based optimization, Artificial Intelligence 123 (2000) 157–184.
[63] I. Inza, P. Larrañaga, B. Sierra, Feature subset selection by Bayesian networks: a comparison with genetic and sequential algorithms, International Journal of Approximate Reasoning 27 (2001) 143–164.
[64] I. Inza, M. Merino, P. Larrañaga, J. Quiroga, B. Sierra, M. Girala, Feature subset selection by genetic algorithms and estimation of distribution algorithms: a case study in the survival of cirrhotic patients treated with TIPS, Artificial Intelligence in Medicine 23 (2001) 187–205.
[65] F. Jensen, S. Anderson, Approximations in Bayesian belief universe for knowledge based systems, in: Sixth Annual Conference on Uncertainty in Artificial Intelligence (UAI'90), Elsevier Science, New York, NY, 1990, pp. 162–169.
[66] F.V. Jensen, T.D. Nielsen, Bayesian Networks and Decision Graphs, 2nd ed., Information Science and Statistics, Springer, 2007.
[67] H. Jia, D. Liu, J. Chen, J. Guan, Learning Markov equivalence classes of Bayesian network with immune genetic algorithm, in: Third IEEE Conference on Industrial Electronics and Applications (ICIEA '08), 2008, pp. 197–202.
[68] H.Y. Jia, D.Y. Liu, P. Yu, Learning dynamic Bayesian network with immune evolutionary algorithm, in: International Conference on Machine Learning and Cybernetics, vol. 5, 2005, pp. 2934–2938.
[69] R. Kabli, F. Herrmann, J. McCall, A chain-model genetic algorithm for Bayesian network structure learning, in: 9th Annual Conference on Genetic and Evolutionary Computation (GECCO '07), ACM, New York, NY, USA, 2007, pp. 1264–1271.
[70] R. Kabli, J. McCall, F. Herrmann, E. Ong, Evolved Bayesian networks as a versatile alternative to Partin tables for prostate cancer management, in: 10th Annual Conference on Genetic and Evolutionary Computation (GECCO '08), ACM, New York, NY, USA, 2008, pp. 1547–1554.
[71] K.J. Kim, J.O. Yoo, S.B. Cho, Robust inference of Bayesian networks using speciated evolution and ensemble, in: M.S. Hacid, N.V. Murray, Z.W. Raś, S. Tsumoto (Eds.), Foundations of Intelligent Systems, Lecture Notes in Computer Science, vol. 3488, Springer, 2005, pp. 185–232.
[72] U. Kjærulff, Optimal decomposition of probabilistic networks by simulated annealing, Statistics and Computing 2 (1992) 7–17.
[73] J.A. Kline, A.J. Novobilski, C. Kabrhel, P.B. Richman, D.M. Courtney, Derivation and validation of a Bayesian network to predict pretest probability of venous thromboembolism, Annals of Emergency Medicine 45 (2005) 282–290.
[74] D. Koller, N. Friedman, Probabilistic Graphical Models: Principles and Techniques, Adaptive Computation and Machine Learning, The MIT Press, 2009.
[75] J.R. Koza, Genetic Programming: On the Programming of Computers by Means of Natural Selection, Complex Adaptive Systems, MIT Press, 1992.
[76] J.R. Koza, Genetic Programming II: Automatic Discovery of Reusable Programs, Complex Adaptive Systems, Springer, 1994.
[77] J.R. Koza, D. Andre, F.H. Bennett, M.A. Keane, Genetic Programming III: Darwinian Invention and Problem Solving, first ed., Morgan Kaufmann Publishers, San Francisco, CA, USA, 1999.
[78] J.R. Koza, M.A. Keane, M.J. Streeter, W. Mydlowec, J. Yu, G. Lanza, Genetic Programming IV: Routine Human–Competitive Machine Intelligence, Kluwer Academic Publishers, 2003.
[79] P. Larrañaga, C. Kuijpers, R. Murga, Y. Yurramendi, Learning Bayesian network structures by searching for the best ordering with genetic algorithms, IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans 26 (1996) 487–493.
[80] P. Larrañaga, C.M.H. Kuijpers, M. Poza, R.H. Murga, Decomposing Bayesian networks: triangulation of the moral graph with genetic algorithms, Statistics and Computing 7 (1997) 19–34.
[81] P. Larrañaga, J. Lozano (Eds.), Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation, Kluwer Academic Publishers, Norwell, MA, USA, 2001.
[82] P. Larrañaga, J.A. Lozano, J.M. Peña, I. Inza, Editorial of the special issue on probabilistic graphical models in classification, Machine Learning 59 (2005) 211–212.
[83] P. Larrañaga, S. Moral, Probabilistic graphical models in artificial intelligence, Applied Soft Computing 11 (2011) 1511–1528.
[84] P. Larrañaga, R. Murga, M. Poza, C. Kuijpers, Structure learning of Bayesian networks by hybrid genetic algorithms, in: AI and Statistics V, Lecture Notes in Statistics, vol. 112, Springer-Verlag, 1996, pp. 165–174.
[85] P. Larrañaga, M. Poza, Y. Yurramendi, R. Murga, C. Kuijpers, Structure learning of Bayesian networks by genetic algorithms: a performance analysis of control parameters, IEEE Transactions on Pattern Analysis and Machine Intelligence 18 (1996) 912–926.
[86] S.L. Lauritzen, D.J. Spiegelhalter, Local computations with probabilities on graphical structures and their application to expert systems, Journal of the Royal Statistical Society. Series B: Methodological 50 (1988) 157–224.
[87] S.L. Lauritzen, N. Wermuth, Graphical models for associations between variables, some of which are qualitative and some quantitative, Annals of Statistics 17 (1989) 31–57.
[88] J. Lee, W. Chung, E. Kim, Structure learning of Bayesian networks using dual genetic algorithm, IEICE Transactions on Information and Systems E91-D (2008) 32–43.

[89] J. Liu, B. Li, T. Dillon, An improved naive Bayesian classifier technique coupled with a novel input solution method, IEEE Transactions on Systems, Man and Cybernetics, Part C: Applications and Reviews 31 (2001) 249–256.
[90] J. Lozano, P. Larrañaga, I. Inza, E. Bengoetxea (Eds.), Towards a New Evolutionary Computation: Advances on Estimation of Distribution Algorithms, Studies in Fuzziness and Soft Computing, vol. 192, Springer, Secaucus, NJ, USA, 2006.
[91] M. Mascherini, F. Stefanini, M-GA: A genetic algorithm to search for the best conditional Gaussian Bayesian network, in: International Conference on Computational Intelligence for Modelling, Control and Automation, and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, vol. 2, 2005, pp. 61–67.
[92] I. Matzkevich, B. Abramson, The topological fusion of Bayes nets, in: 8th Annual conference on Uncertainty in Artificial Intelligence (UAI '92), Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1992, pp. 191–198.
[93] O.J. Mengshoel, Efficient Bayesian Network Inference: Genetic Algorithms, Stochastic Local Search, and Abstraction, Ph.D. thesis, University of Illinois at Urbana-Champaign, Champaign, IL, USA, 1999.
[94] M. Minsky, Steps toward artificial intelligence, Proceedings of the Institute of Radio Engineers 49 (1961) 8–30.
[95] H. Mühlenbein, T. Mahnig, The factorized distribution algorithm for additively decomposed functions, in: IEEE Congress on Evolutionary Computation (CEC '99), vol. 1, 1999, pp. 759–766.
[96] H. Mühlenbein, G. Paaß, From recombination of genes to the estimation of distributions I. Binary parameters, in: H.M. Voigt, W. Ebeling, I. Rechenberger, H.P. Schwefel (Eds.), Fourth International Conference on Parallel Problem Solving from Nature (PPSN IV), Lecture Notes in Computer Science, vol. 1141, Springer, 1996, pp. 178–187.
[97] K. Murphy, Dynamic Bayesian Networks: Representation, Inference and Learning, Ph.D. thesis, UC Berkeley, Computer Science Division, 2002.
[98] J. Muruzábal, C. Cotta, A primer on the evolution of equivalence classes of Bayesian-network structures, in: X. Yao, E. Burke, J.A. Lozano, J. Smith, J.J. Merelo-Guervós, J.A. Bullinaria, J. Rowe, P. Tino, A. Kabán, H.P. Schwefel (Eds.), 8th International Conference on Parallel Problem Solving from Nature (PPSN VIII), Lecture Notes in Computer Science, vol. 3242, Springer, 2004, pp. 612–621.
[99] J.W. Myers, K.B. Laskey, K.A. DeJong, Learning Bayesian networks from incomplete data using evolutionary algorithms, in: 15th Conference on Uncertainty in Artificial Intelligence (UAI '99), Morgan Kaufmann Publishers, 1999, pp. 476–485.
[100] R.E. Neapolitan, Learning Bayesian Networks, Pearson Prentice Hall, Upper Saddle River, NJ, 2004.
[101] D. Nilsson, An efficient algorithm for finding the M most probable configurations in probabilistic expert systems, Statistics and Computing 8 (1998) 159–173.
[102] G. Ochoa, E. Lutton, E. Burke, The cooperative royal road: avoiding hitchhiking, in: N. Monmarché, E.G. Talbi, P. Collet, M. Schoenauer, E. Lutton (Eds.), 8th International Conference on Artificial Evolution, Lecture Notes in Computer Science, vol. 4926, Springer, Berlin,Heidelberg, 2008, pp. 184–195.
[103] J. Park, A. Darwiche, Complexity results and approximation strategies for MAP explanations, Journal of Artificial Intelligence Research 21 (2004) 101–133.
[104] M.J. Pazzani, Searching for dependencies in Bayesian classifiers, in: D. Fisher, H. Lenz (Eds.), Learning from data: Artificial intelligence and statistics V, Lecture Notes in Statistics, Springer-Verlag, 1996, pp. 239–248.
[105] J. Pearl, Bayesian networks: a model of self-activated memory for evidential reasoning, in: 7th Conference of the Cognitive Science Society, 1985, pp. 329–334.
[106] J. Pearl, Distributed revision of composite beliefs, Artificial Intelligence 33 (1987) 173–215.
[107] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988.
[108] M. Pelikan, Hierarchical Bayesian Optimization Algorithm: Toward a New Generation of Evolutionary Algorithms, Studies in Fuzziness and Soft Computing, first ed., vol. 170, Springer, 2005.
[109] M. Pelikan, D. Goldberg, F. Lobo, A survey of optimization by building and using probabilistic models, Computational Optimization and Applications 21 (2002) 5–20.
[110] J.M. Peña, J.A. Lozano, P. Larrañaga, Unsupervised learning of Bayesian networks via estimation of distribution algorithms: an application to gene expression data clustering, International Journal of Uncertainty, Fuzziness and Knowledge-based Systems 12 (2004) 63–82.
[111] M.A. Potter, K.A.D. Jong, J.J. Grefenstette, A coevolutionary approach to learning sequential decision rules, in: Sixth International Conference on Genetic Algorithms (ICGA95), Morgan Kaufmann Publishers, San Francisco, CA, USA, 1995, pp. 366–372.
[112] I. Rechenberg, Evolutionsstrategie-Optimierung Technischer Systeme nach Prinzipien der Biologischen Evolution, Ph.D. thesis, Reprinted by Fromman-Holzboog, 1973.
[113] B. Reiz, L. Csato, D. Dumitrescu, Prüfer number encoding for genetic Bayesian network structure learning algorithm, in: 10th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC '08), IEEE Computer Society, 2008, pp. 239–242.
[114] J. Rissanen, Modeling by shortest data description, Automatica 14 (1978) 465–471.
[115] R. Robinson, Counting unlabeled acyclic digraphs, in: C. Little (Ed.), Combinatorial Mathematics V, Lecture Notes in Mathematics, vol. 622, Springer, Berlin, Heidelberg, 1977, pp. 28–43.
[116] V. Robles, P. Larrañaga, J. Peña, M. Pérez, E. Menasalvas, V. Herves, Learning semi–naïve Bayes structures by estimation of distribution algorithms, in: F. Pires, S. Abreu (Eds.), Progress in Artificial Intelligence, Lecture Notes in Computer Science, vol. 2902, Springer, Berlin, Heidelberg, 2003, pp. 244–258.
[117] V. Robles, P. Larrañaga, J.M. Peña, E. Menasalvas, M.S. Pérez, V. Herves, A. Wasilewska, Bayesian network multi-classifiers for protein secondary structure prediction, Artificial Intelligence in Medicine 31 (2004) 117–136. Data Mining in Genomics and Proteomics.
[118] C. Rojas-Guzmán, M.A. Kramer, An evolutionary computing approach to probabilistic reasoning on Bayesian networks, Evolutionary Computation 4 (1996) 57–85.
[119] T. Romero, P. Larrañaga, Triangulation of Bayesian networks with recursive estimation of distribution algorithms, International Journal of Approximate Reasoning 50 (2009) 472–484. Special Section on Bayesian Modelling.
[120] T. Romero, P. Larrañaga, B. Sierra, Learning Bayesian networks in the space of orderings with estimation of distribution algorithms, International Journal of Pattern Recognition and Artificial Intelligence 18 (2004) 607–625.
[121] B. Ross, E. Zuviria, Evolving dynamic Bayesian networks with multi-objective genetic algorithms, Applied Intelligence 26 (2007) 13–23.
[122] M. Sahami, Learning limited dependence Bayesian classifiers, in: Second International Conference on Knowledge Discovery and Data Mining, 1996, pp. 335–338.
[123] G. Schwarz, Estimating the dimension of a model, Annals of Statistics 6 (1978) 461–464.
[124] R. Shachter, M. Peot, Simulation approaches to general probabilistic inference on belief networks, in: Fifth Annual Conference on Uncertainty in Artificial Intelligence (UAI'89), Elsevier Science, New York, NY, 1989, pp. 311–318.
[125] S.E. Shimony, Finding MAPs for belief networks is NP-hard, Artificial Intelligence 68 (1994) 399–410.
[126] B. Sierra, P. Larrañaga, Predicting survival in malignant skin melanoma using Bayesian networks automatically induced by genetic algorithms: an empirical comparison between different approaches, Artificial Intelligence in Medicine 14 (1998) 215–230. Selected Papers from AIME '97.
[127] B. Sierra, N. Serrano, P. Larrañaga, E.J. Plasencia, I. Inza, J.J. Jiménez, P. Revuelta, M.L. Mora, Using Bayesian networks in the construction of a bi-level multi-classifier: a case study using intensive care unit patients data, Artificial Intelligence in Medicine 22 (2001) 233–248.
[128] P. Spirtes, C. Glymour, An algorithm for fast recovery of sparse causal graphs, Social Science Computer Review 9 (1991) 62–72.
[129] P. Spirtes, C. Glymour, R. Scheines, Causation, Prediction, and Search, 2nd ed., Adaptive Computation and Machine Learning, The MIT Press, 2001.
[130] N. Sriwachirawat, S. Auwatanamongkol, On approximating K-MPE of Bayesian networks using genetic algorithm, in: IEEE Conference on Cybernetics and Intelligent Systems, 2006, pp. 1–6.

[131] K.O. Stanley, R. Miikkulainen, Competitive coevolution through evolutionary complexification, Journal of Artificial Intelligence Research 21 (2004) 63–100.
[132] I. Tsamardinos, L. Brown, C. Aliferis, The max-min hill-climbing Bayesian network structure learning algorithm, Machine Learning 65 (2006) 31–78.
[133] A. Tucker, X. Liu, D. Garway-Heath, Spatial operators for evolving dynamic Bayesian networks from spatio-temporal data, in: E. Cantú-Paz, J. Foster, K. Deb, L. Davis, R. Roy, U.M. O'Reilly, H.G. Beyer, R. Standish, G. Kendall, S. Wilson, M. Harman, J. Wegener, D. Dasgupta, M. Potter, A. Schultz, K. Dowsland, N. Jonoska, J. Miller (Eds.), International Conference on Genetic and Evolutionary Computation (GECCO '03): Part II, Lecture Notes in Computer Science, vol. 2724, Springer, Berlin,Heidelberg, 2003, pp. 2360–2371.
[134] A. Tucker, X. Liu, A. Ogden-Swift, Evolutionary learning of dynamic probabilistic models with large time lags, International Journal of Intelligent Systems 16 (2001) 621–645.
[135] H. Wang, K. Yu, X. Wu, H. Yao, Triangulation of Bayesian networks using an adaptive genetic algorithm, in: F. Esposito, Z. Ras, D. Malerba, G. Semeraro (Eds.), Foundations of Intelligent Systems, Lecture Notes in Computer Science, vol. 4203, Springer, Berlin, Heidelberg, 2006, pp. 127–136.
[136] W.X. Wen, Optimal decomposition of belief networks, in: P.P. Bonissone, M. Henrion, L.N. Kanal, J.F. Lemmer (Eds.), Sixth Annual Conference on Uncertainty in Artificial Intelligence (UAI '90), Elsevier Science Inc., New York, NY, USA, 1991, pp. 209–224.
[137] M.L. Wong, W. Lam, K.S. Leung, Using evolutionary programming and minimum description length principle for data mining of Bayesian networks, IEEE Transactions on Pattern Analysis and Machine Intelligence 21 (1999) 174–178.
[138] M.L. Wong, S.Y. Lee, K.S. Leung, Data mining of Bayesian networks using cooperative coevolution, Decision Support Systems 38 (2004) 451–472.
[139] M.L. Wong, K.S. Leung, An efficient data mining method for learning Bayesian networks using an evolutionary algorithm-based hybrid approach, IEEE Transactions on Evolutionary Computation 8 (2004) 378–404.
[140] Z. Zhu, Y.S. Ong, M. Dash, Markov blanket-embedded genetic algorithm for gene selection, Pattern Recognition 40 (2007) 3236–3248.