

# A Deterministic Annealing Approach to Learning Bayesian Networks

**Ahmed M. Hassan**

AHASSAN@IEEE.ORG

*Computer Engineering Dept.  
Cairo University  
Giza, 1234, Egypt*

**Amir F. Atiya**

AMIR@ALUMNI.CALTECH.EDU

*Computer Engineering Dept.  
Cairo University  
Giza, 1234, Egypt*

**Ihab E. Talkhan**

ITALKHAN@AUCEGYPT.EDU

*Computer Engineering Dept.  
Cairo University  
Giza, 1234, Egypt*

**Editor:**

## Abstract

Graphical Models bring together two different mathematical areas: graph theory and probability theory. Recent years have witnessed an increase in the significance of the role played by Graphical Models in solving several machine learning problems. Graphical Models can be either directed or undirected. Undirected Graphical Models are also called Bayesian networks. The manual construction of Bayesian Networks is usually time consuming and error prone. Therefore, there has been a significant interest in algorithms for the automatic induction of Bayesian Networks structures from data.

This paper presents a new method for the induction of Bayesian Networks structures. The proposed method uses the concept of deterministic annealing to propose an iterative search-score learning algorithm that utilizes a global optimization technique. Deterministic annealing is a global optimization technique that was originally used for clustering, regression,...etc and similar optimization problems. The experimental results show that the proposed approach achieves very promising results compared to other structure learning approaches.

**Keywords:** Graphical Models, Bayesian Networks, Structure Learning, Deterministic Annealing

## 1. Introduction

Graphical models are a marriage between probability theory and graph theory. They provide a natural tool for dealing with two problems that occur throughout applied mathematics and engineering - uncertainty and complexity - and in particular they are playing an increasingly important role in the design and analysis of machine learning algorithms (Jordan,1999). There are two kinds of graphical models: undirected graphical models, also known as Markov Random Fields (MRFs), and directed graphical models, also known as Bayesian Networks

(BNs). Bayesian networks are a graphical representation of a multivariate joint probability distribution that exploits the dependency structure of distributions to describe them in a compact and natural manner (Pearl, 1988).

A BN is a directed acyclic graph, in which nodes correspond to domain variables, and edges correspond to direct probabilistic dependencies between them. The network structure represents a set of conditional independence assertions about the distribution. Informally, the existence of an edge between a variable A, and another variable B can lead to the indication that A causes B. The conditional independence assertions encoded in the network structure are the key to the ability of Bayesian networks to provide a general-purpose representation for complex probability distributions. Automatic learning of Bayesian networks is very crucial in the application of BN to several machine learning problems. Learning of Bayesian networks from data has two main constituents: learning the network structure, and learning the network parameters given its structure.

Recent years have witnessed an ever increasing interest in the automatic induction of Bayesian network structures from data. There are two main approaches for learning the structure of Bayesian Networks. The first poses learning as a constraint satisfaction problem. In this approach, the properties of conditional independence among attributes are estimated using several statistical tests. The second approach poses learning as an optimization problem where several standard heuristic search techniques, such as greedy hill-climbing and simulated annealing, are utilized to find high-scoring structures according to some structure fitness measure. Such local search procedures may sometimes work well, yet they often get stuck in local maximum rather than finding a global one.

In this work, we propose a novel approach for the automatic induction of Bayesian networks structures from data. The approach poses the problem in a probabilistic framework. This means the existence of an edge is not considered as a hard 0/1 issue, but rather we assign a probability  $p$  representing the existence of the edge. We then utilize deterministic annealing (Rose, 1998), a global optimization technique derived within a probabilistic framework from basic information theoretic principles, to assign probabilities to edges. The approach depends on maximizing a network scoring function subject to a constraint on the randomness (Shannon entropy) of the solution. It proceeds iteratively while gradually lowering the level of randomness till it converges to a global solution making it immune to getting stuck at a local maximum. The approach presents a hybrid two-tier solution that restricts the network space by using a node order or by employing some statistical dependence measures. It proceeds iteratively finding a better solution at different levels of entropy till it converges to a global solution.

The rest of the paper is organized as follows. In Section 2, we present a brief survey of previous work. In Section 3 we review the necessary back-ground on deterministic annealing and on learning Bayesian network structure. We present an outline of our approach in section 4. In section 5, we evaluate the performance of our approach on different datasets. We then conclude and present a discussion of future directions in section 6.

## 2. Previous Work

Structure learning algorithms usually fall into two main categories. In the first category, learning is posed as a *constraint satisfaction* problem. Constraint satisfaction problems are

those problems where one must find states that satisfy a number of constraints or criteria. Those algorithms try to discover conditional dependence/independence relationships between different attributes in the data. Later on, they attempt to construct the network that represents most of the independences discovered from the data. Discovering dependences/independences usually utilizes statistical based test like the  $\chi^2$  test, or information theory based test like the mutual information metric. Examples of this approach include (Geiger et al., 1993; Fung and Crawford, 1990; Pearl and Verma, 1991; Cheng et al., 1997; De Campos and Huete, 1997a, 2000). Constraint Satisfaction methods have the disadvantage that repeated independence tests are sensitive to failures and lose statistical power.

In the second category, structure learning is posed as optimization problem. In this approach, a statistically motivated score that describes the quality of the structure, or its fitness to the training data, is defined. Exhaustive searching for the best network structure is NP-Hard (Chickering, 1995). Hence a stochastic optimization method has to be employed to search for the best network structure according to the scoring function. There are two scoring metrics that have been widely used in the literature. The Bayesian score (Cooper and Herskovits, 1992; Heckerman et al., 1995) is equivalent to the marginal likelihood of the model given the data. The BIC (Bayesian Information Criterion) score which is also equivalent to the Minimum Description Length (MDL) of a model (Lam and Bacchus, 1994; Suzuki, 1993) tries to balance between the model's likelihood given the data and its complexity by penalizing complex graphs. Most of the algorithms used for learning structures are stochastic optimization algorithms. Some examples include the K2 algorithm (Cooper and Herskovits, 1992), the Structure EM algorithm (Friedman, 1998), the Hill-Climbing algorithm, the Simulated Annealing algorithm (Wang et al., 2004), the Sparse Candidate algorithm (Friedman et al., 1999), The Ant Colonies algorithm (De Campos et al., 2002) and so on. Other algorithms uses evolutionary algorithms (Li et al., 2004), and Genetic Algorithms (naga et al., 1995; Larrafiag et al., 1996).

As the space of learning Bayesian networks structure is exponential, some preprocessing steps may be applied to restrict the search space and hence make the learning process easier. There are several types of such space restriction steps: (Cheng et al., 1997; Cooper and Herskovits, 1992; Acid and De Campos, 1996; Herskovits and Cooper, 1996; Srinivas et al., 1990) use an ordering among the variables in the model. (Srinivas et al., 1990; Lam and Bacchus, 1993) make use of the fact that some variable are already causally connected. (De Campos, 1998; De Campos and Huete, 1997b) use information about the structure of the model to be recovered. (Acid and De Campos, 1996; Singh and Valtorta, 1995; Spirtes et al., 1995) combine conditional independence and scoring metrics to find the best structure.

### 3. Background

#### 3.1 Deterministic Annealing

Deterministic annealing (DA) is a global optimization technique originally proposed as a clustering algorithm. It has the advantages of being able to escape poor local optima, and being applicable to several problem architectures (Rose et al., 1990, 1993a,b; Rose, 1998; Miller et al., 1996). DA is derived within a probabilistic framework from basic information theoretic principles (e.g., maximum entropy and random coding). The application-specific cost is minimized subject to a constraint on the randomness (Shannon entropy) of the

solution, which is gradually lowered. It starts out with a high degree of exploration of the space and gradually gives way to honing in on the minimum. Unlike the concept of simulated annealing, DA is a purely deterministic method.

The deterministic annealing (DA) approach was originally presented as a solution for clustering optimization problems and its extensions. We will start by introducing DA with the simplest nontrivial problem instance in order to obtain a clear understanding of the essentials. We therefore start with a simple clustering problem that seeks the optimal partition of several data points into a prescribed number of subsets, which minimizes the average cluster variance or the mean squared error (MSE). Let us denote the cost function by  $D$ , where  $D$  is defined as:

$$D = \sum_x \sum_y p(x, y) d(x, y) \tag{1}$$

where  $x$  is a source vector,  $y$  is its best reproducing cluster,  $d(x, y)$  is the probability that the source vector  $x$  is assigned to the cluster  $y$ , and  $d(x, y)$  is the Euclidean distance between  $x$  and the center of the cluster  $y$ . Instead of directly minimizing  $D$ , we recast the optimization problem as minimizing  $D$  subject to a specified level of randomness. We measure the level of randomness by Shannon entropy

$$H = \sum_x \sum_y p(x, y) \log p(x, y) \tag{2}$$

The problem can now be redefined as an optimization problem, where we want to maximize:

$$F = H - \beta D \tag{3}$$

where  $\beta = 1/T$ , is the Lagrange multiplier,  $D$  is the cost function given by (1), and  $H$  is the Shannon entropy given by (2). We start out with large temperature and gradually lower it during the course of iterations. When  $T$  is large, we mainly attempt to maximize the entropy. As  $T$  gets lower, we trade entropy for reduction in the cost function, and as  $T$  approaches zero, we minimize the cost function directly to obtain a hard solution.

### 3.2 Structure Learning of Bayesian Networks

A Bayesian Network  $B$  over a set of random variables  $X_i \mid i = 1 : n$  is a directed acyclic graph that represents the joint probability distribution over all  $X_i$ 's

$$P(X_1, \dots, X_n) = \prod_{i=1}^n p(X_i | Pa(X_i)) \tag{4}$$

where  $Pa(X_i)$  are the parents of node  $X_i$ .

The Bayesian Network  $B$  is a pair  $\langle G, \theta \rangle$ .  $G$  is the directed acyclic graph  $G \langle V, E \rangle$ , where the set of nodes  $V = \{X_1, X_2, X_3, \dots, X_n\}$  represents the random variables, and  $E$  is the set of edges encoding the dependence relations among the variables.  $\theta$  represents a set of parameters for each variable in  $V$ , which defines its conditional probability distribution. For each variable  $X_i \in V$ , we have a family of conditional distributions  $P(X_i | Pa(X_i))$ , where  $Pa(X_i)$  is the set of  $X_i$ 's parents. Those conditional distributions allow us to recover the

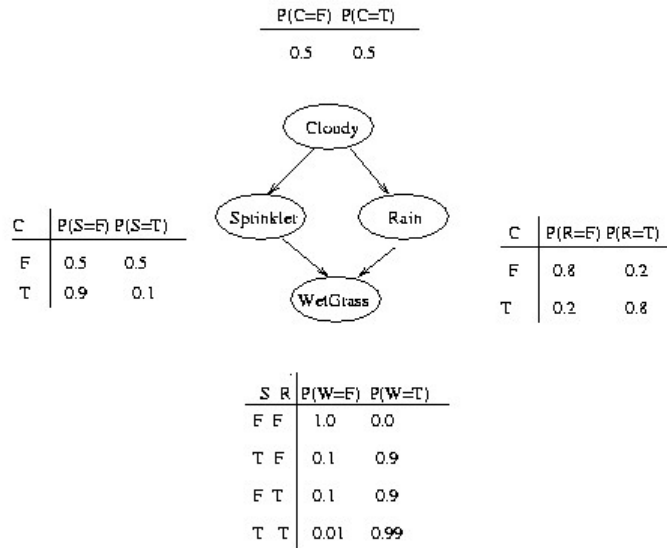


Figure 1: The Sprinkler Network

joint distribution over  $V$  described in (4). An example of a Bayesian network is illustrated in.

The problem of learning a Bayesian Network Structure can be stated as follows: Given a training set  $T$  of instances of  $(X_1, \dots, X_n)$ , find a DAG  $G$  that best matches  $T$ . A common approach is to introduce a scoring function that measures how the DAG fits the data, and use it to search for the best network. The most commonly used scoring functions are the *Bayesian* scoring metric and the *Minimal Description Length* (MDL) metric. Both metrics are described in details in (Cooper and Herskovits, 1992) and (Lam and Bacchus, 1994) respectively.

The Bayesian metric, also called the K2 metric is the result of a Bayesian approach to learning Bayesian networks from data. It assumes that a complete database  $D$  of sample cases over a set of attributes that accurately models the network is given. Given this dataset, we can compute from a given network structure  $B_S$  and a set of conditional probabilities  $B_P$  associated with it the probability of the database, i.e., we can compute  $P(D|B_S, B_P)$ . If we integrated over all possible sets of conditional probabilities  $B_P$  for the given structure  $B_S$ , we get  $P(B_S, D)$

$$P(B_S, D) = \int_{B_P} P(D|B_S, B_P)P(B_S)d_{B_P} \tag{5}$$

This expression can be rewritten as:

$$P(B_S, D) = P(G) \prod_{i=1}^n \prod_{j=1}^{q_i} \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \prod_{k=1}^{r_i} N_{ijk}! \tag{6}$$

where  $r_i$  is the number of possible values of the variable  $x_i$ ,  $q_i$  is the number of possible configurations (instantiations) for the variables in  $Pa(x_i)$ ,  $N_{ijk}$  is the number of cases in

$D$  in which variable  $x_i$  has its  $k$ th value and  $Pa(x_i)$  is instantiated to its  $j$ th value, and  $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$ . Assuming a uniform prior for  $P(G)$  and using  $\log(P(G, D))$  instead of  $P(G, D)$ , we get the K2 metric:

$$F_B(B_s : D) = \sum_{i=1}^n F_B(x_i, Pa(x_i) : N_{x_i, Pa(x_i)}) \quad (7)$$

where  $N_{x_i, Pa(x_i)}$  are the statistics of the variable  $x_i$  and  $Pa(x_i)$  in  $D$ .

$$F_B(x_i, Pa(x_i) : N_{x_i, Pa(x_i)}) = \sum_{j=1}^{q_i} \left( \log \left( \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \right) + \sum_{k=1}^{r_i} \log(N_{ijk}!) \right) \quad (8)$$

The MDL metric is based on the MDL principle which is based on the idea that the best model that represents a set of data items is the model that minimizes:

- the length of the encoding of the model, and
- the length of the encoding of the data given the model.

To apply this principle to Bayesian networks, we need to specify how the encoding of the network itself and the raw data given the network can be performed. A simple approximation of the MDL metric which is also equivalent to the The Bayesian information criterion (BIC) is:

$$L(B_s, D) = \log(P(B_s)) + \sum_{i=1}^n \sum_{j=1}^{q_i} \sum_{k=1}^{r_i} N_{ijk} \log \frac{N_{ijk}}{N_{ij}} - \frac{1}{2} \sum_{i=1}^n q_i (r_i - 1) \log N \quad (9)$$

where  $r_i$  is the number of possible values of the variable  $x_i$ ,  $q_i$  is the number of possible configurations (instantiations) for the variables in  $Pa(x_i)$ ,  $N_{ijk}$  is the number of cases in  $D$  in which variable  $x_i$  has its  $k$ th value and  $Pa(x_i)$  is instantiated to its  $j$ th value, and  $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$ . Assuming a uniform prior for  $P(B_s)$ , we get:

$$L(B_s, D) = \sum_{i=1}^n \sum_{j=1}^{q_i} \sum_{k=1}^{r_i} N_{ijk} \log \frac{N_{ijk}}{N_{ij}} - \frac{1}{2} \sum_{i=1}^n q_i (r_i - 1) \log N \quad (10)$$

#### 4. A Deterministic Annealing Approach to Learning Bayesian Networks

In the proposed method, we propose a new structure learning algorithm for Bayesian networks based on the concept of DA. To model the problem in a way that is suitable to the application of DA, we assume that the existence of an edge between two nodes is no longer a hard decision. Rather, an edge does always exist with some probability. The problem of learning Bayesian networks structure is now transformed to assigning a probability to each possible edge. The algorithms advances in this soft manner without producing any intermediate hard solutions. Taking hard decisions about the existence of edges will only take place when producing the final solution.

The proposed approach is a two tier approach. In the first tier, we try to restrict the search space. Several techniques may be used for this purpose and will be outlined hereunder. In the second tier we apply the DA method to the structure learning problem. Below, an outline of the algorithm will be presented followed by a detailed discussion of several challenges that needed to be addressed.

**Input**

- *A dataset  $D$*
- *A decomposable scoring function  $Score()$*
- *A function  $H()$  that calculates solution entropy*
- *A threshold on entropy  $T_H$*

**Output**

- *A network  $B$*

**Algorithm**

- *Restrict the search space*
- *Create an initial network  $B_{init}$*
- *for each possible edge  $e_{ij}$  let  $P(e_{ij}) = 0.5$*
- *Loop until  $H(B) < T_H$*
- *Begin*
  - $B_{score} = Score(B)$
  - $F = H - \beta D$
  - *Search for  $B$  that maximizes  $F$*
  - $B_{init} = B$
  - *Increase  $\beta$*
- *End*
- *foreach edge  $e_{ij}$* 
  - *if  $P(e_{ij}) > 0.5$*
  - \* *add  $e_{ij}$  to the final network*

#### 4.1 Restricting the Search Space

Restricting or reducing the search space is a crucial step in the structure learning problem due to the exponential nature of the search space. Several approaches may be employed to accomplish this purpose. Some of them will be described in the few coming paragraphs.

The first approach suggests using an independence measure to filter out edges linking independent nodes. Filtering out edges based on an independence test is rather risky, because any correct edges excluded here will never appear in the final solution. Hence, we suggest using two or more independence tests and only filter out edges that fail to pass a rather high threshold in both tests. The most frequently used tests for this purpose are the  $\chi^2$  test and the mutual information test. Some variants of the mutual information test, described in (Friedman et al., 1999), that may also be employed are the discrepancy mutual information test and the shielding mutual information test. Another approach is to allow the search algorithm to select a non candidate edge for altering with some low probability.

The second approach assumes that an ordering of nodes is present. The parent of any node must be preceding it in the ordering. This allows us to reduce the number of candidate edges by half.

The last approach allows the algorithm to make use of any domain knowledge suggesting that some nodes are independent. This domain knowledge may be respected by the algorithm by removing edges between independent nodes from the candidate set.

## 4.2 Representation

Before moving on to describe the details of the algorithm, we will elaborate on the representation of the network. The most convenient way of representing a graph is the matrix representation. The Bayesian network will be presented as an  $n \times n$  matrix, where  $n$  is the number of random variables. Each position  $G(i, j)$  will hold a real number between 0 and 1 that corresponds to the probability of existence for the edge between node  $i$  and node  $j$ . Initially all candidate edges will have a probability of 0.5. Entries corresponding to edges removed from the candidate set, by the search space restriction phase, will be set to 0. This matrix will be called a Probabilistic Directed Acyclic Graph (PDAG).

Along with the matrix, there will be a vector with the indexes of the candidate edges. Initially this vector will contain all possible edges. Later on, edges will be removed from the candidate edges vector in the search space restriction phase.

Take as an example the sprinkler network of figure 1. If we decided to restrict the search space based of the node order (C,R,S,W), the initial PDAG will be as follows:

$$\begin{pmatrix} 0 & 0.5 & 0.5 & 0.5 \\ 0 & 0 & 0.5 & 0.5 \\ 0 & 0 & 0 & 0.5 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

## 4.3 Local Search Algorithm

The local search algorithm used is a simple greedy hill-climbing algorithm augmented with a simple Tabu (Glover, 1993; Glover et al., 1990) list. In its simplest form, a Tabu list contains recently visited solutions (less than  $n$  moves ago). Instead of applying the best local change, the algorithm select the best local change resulting in a solution not existing in the Tabu list. The local search procedure terminates when a certain number of local changes fail to present an improvement over the current solution.

The local moves done by the search algorithm select one of the candidate edges at random and alter its probability. The probability altering is accomplished by sampling a new value from the edge probability distribution. Edge distribution follows a softmax distribution parameterized by the temperature  $T$ , where  $T = 1/\beta$ . Parameterizing the distribution by the temperature makes it more discriminating as the temperature is lowered. This parameterization assures that the lower the temperature , The nearer the probabilities are to 0 or 1.

If  $u$  is random number sampled from a zero-mean Gaussian distribution, the edge probability can be calculated as follows:

$$P(e) = \frac{e^{\frac{1}{T}u}}{e^{\frac{1}{T}u} + e^{\frac{1}{T}(1-u)}} \tag{11}$$

The local search procedure is desribed in the following listing:

**Input**

- *An initial probabilistic DAG  $G_{init}$*
- *A probabilistic DAG scoring function  $Score_{pdag}()$*

**Output**

- *A new probabilistic DAG  $G_{out}$*

**Algorithm**

- *Define a uniform distribution over all possible edges*
- *Select an edge at random*
- *Sample a new value for the edge probability*
- *Check whether the new pdag exists in the Tabu list*
- *If it already exists, ignore it*
- *$new\_G\_score = Score_{pdag}(new\_G)$*
- *If the pdag has a better score, make it the current pdag*

**4.4 Evaluating PDAG's**

Previous sections discussed several scoring metrics to evaluate DAGs against data. To evaluate our probabilistic DAG, we use Monte Carlo sampling to generate a number of DAGs from the PDAG, a description of Monte Carlo Method can be found in (Robert and Casella, 1999). Standard scoring method are used to evaluate those dags. The PDAG score is then reported as the average of the scores of the generated DAGs. The algorithm is described in the following listing:

**Input**

- *A probabilistic DAG  $G$*
- *A dataset  $D$*
- *A number of samples  $N$*
- *A decomposable scoring function  $Score()$*

**Output**

- *A score for the probabilistic DAG*

**Algorithm**

- *Define an array of DAGs  $G_{arr}$*
- *For  $i = 1 : N$*
- *Begin*
  - *Foreach element in  $G$* 
    - \* *Generate a random number  $r$*
    - \* *Create a temp graph  $G_{tmp}$*
    - \* *if( $P(e_{ij}) > r$ ) add an edge  $e_{ij}$  to the graph*
  - *Add  $G_{tmp}$  to  $G_{arr}$*
  - *Calculate the score of all dags in  $G_{arr}$*
  - *Set  $G$ 's score to the average of  $G_{arr}$  scores*

## 4.5 Caching

Several methods exist for the evaluation of DAG's that may represent a Bayesian networks. The most commonly used scoring functions are the *Bayesian* scoring metric (Cooper and Herskovits, 1992) and the *Minimal Description Length* (MDL) metric (Lam and Bacchus, 1994). Those two metrics have a very important characteristic called *decomposability*. This characteristic is very useful because it allows reusing computations made for some DAGs in the evaluation of new DAGs. The *Bayesian* and *MDL* scoring functions can be decomposed in the following way:

$$Score(G) = \sum_i Score(X_i|Pa(X_i)) \quad (12)$$

This decomposability allows implementing a caching mechanism to reuse computations among several graphs. In this approach, two levels of caching are employed. The first level cache is a very small cache that caches complete DAG's. The second level cache is a larger cache that caches families. Families are the subgraphs connecting each node to its parents. Usually the DAGs have a lot of families in common. Caching scores of families make the scoring process much more efficient than scoring the DAGs one at a time.

## 4.6 Calculating Entropy

Calculating the entropy is strait forward given the Probabilistic DAG:

$$H(G) = - \sum_i \sum_j H_{ij} F(e_{ij}) \quad (13)$$

where  $H_{ij}$  is defined by:

$$H_{ij} = -[P(e_{ij})\log P(e_{ij}) + (1 - P(e_{ij}))\log(1 - P(e_{ij}))] \quad (14)$$

and  $F$  is defined by:

$$\begin{aligned} F(e_{ij}) &= 1 && \text{If } e_{ij} \in \text{Candidate Edge List} \\ &= 0 && \text{Otherwise} \end{aligned} \quad (15)$$

## 5. Experimental Evaluation

### 5.1 Datasets

In this section we carry out several experiments to illustrate the effectiveness of the proposed algorithm. Several networks from several problem domains are considered. All of data sets are generated from the well-known benchmarks of Bayesian networks including ASIA (et al, 1999), INSURANCE (Binder et al., 1997), and ALARM (Beinlich et al., 1989).

The ASIA network is small network that studies the effect of several parameters on having lung cancer. The network has 8 nodes and 8 edges. The INSURANCE network was originally used for evaluating car insurance risks. The network contains 27 variables and 52 arcs. ALARM network is used in the medical domain for potential anesthesia diagnosis in the operating room. The network has 37 nodes, of which 13 have 2 values, 22 have 3 values, and 2 have 4 values, and 46 directed edges. The ALARM network has been considered to be a benchmark for evaluating learning algorithms.

## 5.2 Comparisons

Empirical comparisons were carried out for the proposed learning algorithm with the K2 algorithm (Cooper and Herskovits, 1992). The K2 algorithm is one of the best and most frequently used algorithms for learning Bayesian networks. K2 uses a Bayesian scoring metric, which measures the joint probability of a BN  $G$  and a database  $D$ . The metric has adopted the name of the algorithm and is referred to as the K2 metric. K2 searches for the parent configuration for each node that maximizes the K2 metric. K2 needs an initial node ordering to restrict the search space.

## 5.3 Performance Measures

Several measuring metrics were employed for the purpose of evaluating the quality of the output of the proposed algorithm. These output quality measures were also used to compare the algorithms output to the output of other algorithms. The used measures are:

- The K2 metric assigned to the network.
- The BIC metric assigned to the network.
- The Kullback-Leibler divergence (KL) defined as follows:

$$KL(G, D) = \sum_{i=1}^n Dep(x_i Pa(x_i)) \quad (16)$$

where  $Dep(A, B)$  is the measure of mutual information between  $A$  and  $B$ . Note that  $KL(G, D)$  is a decreasing monotonic transformation of the Kullback distance (Kullback, 1968) between the probability distribution associated with the database and the probability distribution associated with the network. We use this transformation because it can be calculated very efficiently, whereas the computation of the Kullback distance has an exponential complexity.

- Structural differences between the learned and the original network: the number of arcs added (A), and deleted (D) compared with the original network.

## 5.4 Results

We carried out several experiments on the three networks with varying the size of the datasets. Before displaying the results, we elaborate on some of the parameter settings used:

- The local search algorithm is greedy hill-climbing algorithm augmented with a simple Tabu lists to save recently explored solutions. The size of the Tabu list was set to 10.
- The number of samples used by the Monte Carlo method to score probabilistic DAGs ranges from 100 to 500.
- Two level caching scheme was employed. The first level cache is a small cache with size 10 that caches complete DAG's. The second level cache is a larger cache that caches up to 500 families.

- The DA iterations stop when the entropy drops between a low threshold. This results in all probabilities converging either to a value near 1 or near 0.
- A predetermined node ordering is used to reduce the search space.

The results for the ASIA, INSURANCE, and ALARM networks are shown in Table 2, Table 3, and Table 4 respectively. The performance measure values calculated for the original networks are illustrated in Table 1 We can see from the results that the proposed algorithm reaches the same results reached by K2 even when trained on a dataset with smaller size.

	ASIA	INSURANCE	ALARM
<b>K2</b>	-2.3456e+03	-1.4200e+05	-9.4492e+04
<b>BIC</b>	-2.3581e+03	-1.4389e+05	-9.5294e+04
<b>KL</b>	0.6976	7.1826	11.6028

Table 1: Results for the original graphs of ASIA, INSURANCE, and ALARM networks

Dataset Size	250		500	
	K2	BN-DA	K2	BN-DA
<b>K2</b>	-2.3480e+03	-2.3466e+03	-2.3463e+03	-2.3447e+03
<b>BIC</b>	-2.3605e+03	-2.3609e+03	-2.3572e+03	-2.3545e+03
<b>KL</b>	0.6984	0.8268	0.6975	0.6975
<b>A</b>	2	1	1	0
<b>D</b>	1	1	1	1
Dataset Size	750		1000	
	K2	BN-DA	K2	BN-DA
<b>K2</b>	-2.3447e+03	-2.3447e+03	-2.3447e+03	-2.3447e+03
<b>BIC</b>	-2.3545e+03	-2.3545e+03	-2.3545e+03	-2.3545e+03
<b>KL</b>	0.6975	0.6975	0.6975	0.6975
<b>A</b>	0	0	0	0
<b>D</b>	1	1	1	1

Table 2: Results for the ASIA network

## 6. Conclusion

In this paper a new search-score algorithm for learning Bayesian networks from data was proposed. The novelty of this algorithm lies in the use of the deterministic annealing approach to optimization for guiding the search process. The algorithms allows the integration of search-score methods and the constraint satisfaction methods through restricting the search space by domain knowledge, and/or conditional independence tests. The proposed algorithm does not require a node ordering, although it may benefit from one if it already exists. The experimental results are encouraging and the algorithm achieves good results compared to other algorithms based on different search methods.

Dataset Size	2500		5000	
	K2	BN-DA	K2	BN-DA
K2	-1.4694e+05	-1.4647e+05	-1.4695e+05	-1.4695e+05
BIC	-1.4910e+05	-1.4879e+05	-1.4883e+05	-1.4883e+05
KL	7.2181	7.2134	7.2170	7.2170
A	10	8	8	8
D	13	12	14	14
Dataset Size	7500		10000	
	K2	BN-DA	K2	BN-DA
K2	-1.4693e+05	-1.4688e+05	-1.4661e+05	-1.4674e+05
BIC	-1.4910e+05	-1.4904e+05	-1.4893e+05	-1.4883e+05
KL	7.2181	7.2085	7.2227	7.2267
A	10	8	10	10
D	13	13	12	12

Table 3: Results for the INSURANCE network

Dataset Size	2500		5000	
	K2	BN-DA	K2	BN-DA
K2	-9.4706e+04	-9.4725e+04	-9.4738e+04	-9.4738e+04
BIC	-9.5724e+04	-9.5733e+04	-9.5790e+04	-9.5790e+04
KL	11.5667	11.5699	11.5820	11.5820
A	13	12	14	14
D	2	2	2	2
Dataset Size	7500		10000	
	K2	BN-DA	K2	BN-DA
K2	-9.4749e+04	-9.4739e+04	-9.4775e+04	-9.4791e+04
BIC	-9.5847e+04	-9.5792e+04	-9.5899e+04	-9.5915e+04
KL	11.5716	11.5819	11.5874	11.5874
A	13	13	12	12
D	2	2	2	2

Table 4: Results for the ALARM network

## References

- S. Acid and L. De Campos. An algorithm for learning probabilistic belief networks. In *Proceedings of the Sixth Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU)*, pages 979–984, 1996.
- I. Beinlich, H. Suermondt, R. Chavez, and G. Cooper. The case study with two probabilistic inference techniques for Bayesian networks. In *Proceedings of the Second European Conference on Artificial Intelligence in Medicine*, pages 247–256, 1989.
- J. Binder, D. Koller, S. Russell, and K. Kanazawa. Adaptive probabilistic networks with hidden variables. *Machine Learning*, 29:213244, 1997.

- J. Cheng, D. Bell, and W. Liu. An algorithm for Bayesian belief network construction from data. In *Proceedings of the Fourth International Workshop on Artificial Intelligence and Statistics*, pages 83–90, 1997.
- D. Chickering. Learning Bayesian networks is NP-Complete. Springer, 1995.
- G. Cooper and E. Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9:309347, 1992.
- L. De Campos and J. Huete. On the use of independence relationships for learning simplified belief networks. *International Journal of Intelligent Systems*, 12:495–522, 1997a.
- L. De Campos and J. Huete. On the use of independence relationships for learning simplified belief networks. *Internat. J. Intell. Sys.* 12, 7:495–522, 1997b.
- L. De Campos, J. Fernandez-Luna, J. Gamez, and J. Puerta. Ant colony optimization for learning Bayesian networks. *International Journal of Approximate Reasoning*, 31: 291–311, 2002.
- L.M. De Campos. Independency relationships and learning algorithms for singly connected networks. *J. Exp. Theoret. Artificial Intelligence*, 10:511–549, 1998.
- Luis M. De Campos and Juan F. Huete. A new approach for learning belief networks using independence criteria. *International Journal of Approximate Reasoning*, 24:11–37, 2000.
- Cowel et al. Probabilistic networks and expert systems. Springer Verlag, 1999.
- N. Friedman. The Bayesian structural em algorithm. In *Proceedings of the 14th Conference In Uncertainty in Artificial Intelligence*, page 129138. Morgan Kaufmann, 1998.
- N. Friedman, I. Nachman, , and D. Peer. Learning Bayesian network structure from massive datasets: The sparse candidate algorithm. In *Proceedings of the 15th Conference In Uncertainty in Artificial Intelligence*, page 206215. Morgan Kaufmann, 1999.
- R. Fung and S. Crawford. A system for the induction of probabilistic models. In *Proceedings of the Eighth National conference on Artificial Intelligence*, pages 762–769. MIT Press, 1990.
- D. Geiger, A. Paz, and J. Pearl. Learning simple causal structures. *International Journal of Intelligent Systems*, 8:231–247, 1993.
- F. Glover. A user’s guide to Tabu search. *Annals of Operations Research*, 41:3–28, 1993.
- F. Glover, E. Taillard, and D. De Werr. Tabu search. *ORSA Journal on Computing*, 2: 4–32, 1990.
- D. Heckerman, D. Geiger, , and D. Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20:197243, 1995.
- E. Herskovits and G. Cooper. An entropy-driven system for the construction of probabilistic expert systems from databases. In *Proceedings of the Sixth Conference on Uncertainty in Artificial Intelligence*, pages 54–62, 1996.

- S. Kullback. Information theory and statistics. Dover, New York, 1968.
- W. Lam and F. Bacchus. Using causal information and local measures to learn Bayesian belief networks. In *Proceedings of the Ninth Conference on Uncertainty in Artificial Intelligence*, pages 243–250, 1993.
- W. Lam and F. Bacchus. Learning Bayesian belief networks: An approach based on the MDL principle. *Computational Intelligence*, 10:269–293, 1994.
- P. Larrafiag, M. Poza, Y. Yurramendi, R. Murga, and C. Kuijpers. Structure learning of bayesian networks by genetic algorithms:performance analysis of control parameters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18:912–926, 1996.
- X. Li, S. Yuan, and X. He. Learning Bayesian networks structure based on extending evolutionary programming. In *Proceedings of the 3rd International Conference on Machine Learning and Cybernetics*, pages 1594–1598, 2004.
- D. Miller, A. Rao, K. Rose, and A. Gersho. A global optimization technique for statistical classifier design. *IEEE Transactions on Signal Processing*, 44(12):3108–3122, 1996.
- P. naga, R. Murga, M. Poza, and C. Kuijpers. Structure learning of bayesian networks by hybrid genetic algorithms. In *Preliminary Papers of the Fifth International Workshop on Artificial Intelligence and Statistics*, pages 310–316, 1995.
- J. Pearl and T. Verma. A theory of inferred causation. In *Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning*, pages 441–452. Morgan and Kaufmann, 1991.
- C. Robert and G. Casella. Monte Carlo statistical methods. Springer, 1999.
- K. Rose. Deterministic annealing for clustering, compression,classification, regression, and related optimization problems. *Proceedings of the IEEE*, 86:2210 – 2239, 1998.
- K. Rose, E. Gurewitz, and G. Fox. A deterministic annealing approach to clustering. *Pattern Recognition Letters*, 11:589–594, 1990.
- K. Rose, E. Gurewitz, and G. Fox. Constrained clustering as an optimization method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15:785–794, 1993a.
- K. Rose, E. Gurewitz, and G. Fox. Vector quantization by deterministic annealing. *IEEE Transactions on Information Theory*, 38:1249–1257, 1993b.
- M. Singh and M. Valtorta. Construction of Bayesian networks structures from data: a survey and an efficient algorithm. *International Journal of Approximate Reasoning*, 12: 111–123, 1995.
- P. Spirtes, T. Richarson, and C. Meek. Learning Bayesian networks with discrete variables from data. In *Proceedings of the First International Conference on Knowledge Discovery and Data Mining*, pages 294–299, 1995.

- S. Srinivas, S. Russell, and A. Agogino. Automated construction of sparse Bayesian networks from unstructured probabilistic models and domain information. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 295–308, 1990.
- J. Suzuki. A construction of Bayesian networks from databases based on an MDL principle. In *Proceedings of the 9th Annual Conference on Uncertainty in Artificial Intelligence*, 1993.
- Tie Wang, J.W. Touchman, and Guoliang Xue. Applying two-level simulated annealing on bayesian structure learning to infer genetic networks. In *Proceedings of the IEEE Computational Systems Bioinformatics Conference*, pages 647– 648, 2004.