A New Framework for Learning Generalized Credal Networks

Andrés R. Masegosa, Serafín Moral
Dpto. Ciencias de la Computación e Inteligencia Artificial
Universidad de Granada, 18071, Granada, Spain
{andrew,smc}@decsai.ugr.es

Abstract

In this paper we consider the problem of learning credal networks from observations when the prior information is given by a set of prior densities instead of a single one. We shall concentrate in the case in which the prior information is given by a family of Dirichlet distributions with uniform weights and different values of the equivalent sample size parameter. Also the use of imprecise probability models to specify the prior probability regarding the different graphs will be considered. The novelty is twice: first we assume an additional information on the set of possible values of the equivalent sample size parameter; and second we give a formalization of the problem which includes also as particular case different Bayesian approaches. Additionally, approximate and exact algorithms based on the A* search procedure are provided to compute the set of undominated decisions. Some preliminary experiments are reported.

Keywords. Credal networks, learning, imprecise sample size Dirichlet model, search algorithms.

1 Introduction

Learning Bayesian networks from a dataset of observations [15, 8, 12] is an important research challenge that despite the important effort made in past years, is far from having resulted in a satisfactory solution in all situations. In particular, in some cases it is assumed that providing a single network can be insufficient, especially if the sample size is small, as there can be several graphical structures with high posterior probability given the data. Model averaging approaches [7] provide a set of alternative networks each one with its own posterior probability. Inferences about any structural feature (for example the presence of a link) are carried out by averaging in the solution set. In this paper, we will show that even if our decision consists in selecting one graph, it makes sense to consider a set of alternative solutions if we use imprecise probability models to specify the prior probability regarding graphs and parameters [18]. In this paper we present a new framework which is based on imprecision in the prior probability about the graph associated to the network and in the value of the equivalent sample size of a BDeu score [2]. The result of the learning problem will be a set of undominated graphs (a *multigraph credal network* [5, 10]). The approach is based on our previous work [10] but assumes more information on the equivalent sample size (ϵ-contaminated model of a uniform distribution) and on the space of graphs. The main contributions of this paper are the following: we provide a new formulation of the learning problem depending on the prior information and the set of possible decisions; we transform the problems of computing undominated solutions in simpler problems involving the optimization of specific scores; and we propose different algorithms both approximate and exact for the associated optimization problems. Corani and Zaffalon [4] also consider multigraph credal networks in a classification problem. They are based on an imprecise prior information on the space of graphs but their approach is different of the one proposed in this paper and they concentrate in the classification problem and not in the computation of the graphs.

The paper is structured as follows: Section 2 reviews the problem of learning Bayesian networks; Section 3 is devoted to credal networks; Section 4 introduces our framework for learning multigraph credal networks [10]; Section 5 proposes approximate and exact algorithms to compute the set of undominated decisions; Section 6 reports some very preliminary experiments; while Section 7 is devoted to the conclusions.

2 Learning Bayesian Networks

Assume that we have a vector $\mathbf{X} = \{X_1, \ldots, X_m\}$ of variables. A *Bayesian network* about variables $\mathbf{X}$ is a directed acyclic graph G with a node for each vari-

able X_i and a list of conditional probability distributions $(p(X_1|\Pi_1), \ldots, p(X_m|\Pi_m))$, where Π_i is the set of parents of variable X_i in graph G [13] and $p(X_i|\Pi_i)$ is a conditional distribution of variable X_i given Π. Given the independence relationships represented by G, the list of conditional distributions determines a joint probability distribution about $\mathbf{X}$ as a product of the conditional distributions (p *factorizes* with respect to G):

$$p(X_1, \ldots, X_m) = p(X_1|\Pi_1) \cdot \ldots \cdot p(X_m|\Pi_m) \quad (1)$$

The number of different values of variables in Π_i is denoted by q_i, the set of possible values of Π_i by $\{\pi_1^i, \ldots, \pi_{q_i}^i\}$, the number of possible values of X_i is denoted by k_i, and the set of possible values of X_i by $\{w_1^i, \ldots, w_{k_i}^i\}$.

If we have a database of n observations D of variables $\{X_1, \ldots, X_m\}$, *learning* a Bayesian network consists first in estimating the graph G, and then estimating the conditional probability distributions associated to the graph [12]. In both cases, the most common approach is based on assuming a Dirichlet prior probability distribution for the values of each one of the conditional probability distributions $p(X_i|\Pi_i = \pi_j^i)$ with parameters $D(\alpha^i, \ldots, \alpha^i)$ (the weights are the same for the different values π_j^i of the parents, but they can be different for different variables). The value $s^i = k_i\alpha^i$ is called the *equivalent sample size*.

To estimate G, it is also assumed that there is a prior probability about the different graph structures (usually the uniform) and that the prior distributions for the different conditional distributions are independent. Under these conditions, it is possible to compute $p(G|D)$ which is a value proportional to (under the uniform prior for the graphs):

$$p(G|D) \propto P(D|G) = \prod_{i=1}^{m}\prod_{j=1}^{q_i}\frac{\Gamma(k_i\alpha^i)}{\Gamma(n_{ij}+k_i\alpha^i)}\prod_{l=1}^{k_i}\frac{\Gamma(\alpha^i+n_{ijl})}{\Gamma(\alpha^i)} \quad (2)$$

where n_{ij} is the number of cases in D in which $\Pi_i = \pi_j^i$ and n_{ijl} the number of cases in D in which $\Pi_i = \pi_j$, $X_i = w_l^i$, and $\Gamma(.)$ is the gamma function. This value is called the score of the graph given the data and will be denoted as $Score(G|D)$.

The problem is then to find the graph with maximum score, and usually a greedy search algorithm is employed: given an initial graph, the set of graphs obtained by adding, removing, and inverting a link is computed, and for each one of them the score is found. Then the current graph is changed to the graph of the set with highest score, and the process is repeated while a score greater than the score of the current graph can be found.

Once a graph with highest score has been found, G, the conditional probabilities can be estimated (*parameter learning*). If for each conditional probability $p(X_i|\Pi_i = \pi_j^i)$ we have a Dirichlet prior with parameters $D(\alpha^i, \ldots, \alpha^i)$ and these prior distributions are independent, then the estimation can be done independently for each conditional probability. The estimated values are [12]:

$$p(X_i = w_l^i|\Pi_i = \pi_j^i, D) = \frac{n_{ijl}+\alpha^i}{n_{ij}+k_i\alpha^i} \quad (3)$$

In both problems (learning the structure and the parameters), we have to specify how the weights, α^i, are computed. Usually the so called *Bayesian Dirichlet equivalent metric* or BDeu [2] is used in which a parameter s is fixed (the *global equivalent sample size*) and then the weights for each variable are computed as $\alpha^i = \frac{s}{q_i k_i}$. Though for the conditional probabilities it is also common to consider the Laplace correction which is equivalent to consider $\alpha^i = 1$. In this paper, we will always consider $\alpha^i = \frac{s}{q_i k_i}$.

3 Credal Networks

A *credal network* [5] is a generalized Bayesian network where the probabilities can be imprecise. More concretely, a *locally defined credal network* [5] is a directed acyclic graph G and a list $(\mathcal{P}_1, \ldots, \mathcal{P}_m)$ where each $\mathcal{P}_i$ is a set of conditional distributions for variable X_i given its parents in G. The joint credal set associated to a credal network, is the set of probability distributions p that can be obtained as a product $p(X_1, \ldots, X_m) = p_1(X_1|\Pi_1) \cdot \ldots \cdot p_m(X_m|\Pi_m)$, where $p_i \in \mathcal{P}_i$. A *credal network* is a directed acyclic graph G and a credal set of joint probability distributions $\mathcal{P}$ such that any extreme probability $p \in \mathcal{P}$ factorizes with respect to G, i.e. it can be expressed as $p(\mathbf{X}) = p_1(X_1|\Pi_1) \cdot \ldots \cdot p(X_m|\Pi_m)$, where $p(X_i|\Pi_i)$ is a conditional distribution of X_i given its parents in G. Not every credal network is locally defined.

A credal network can be obtained by determining a single graphical structure (either elicited from experts or learned with a Bayesian procedure) and a set of decomposable probability distributions learned with an imprecise probability procedure, for example using the Imprecise Dirichlet Model (IDM) [19]. The most simple application is the separable estimation of the conditional probabilities, being the result a locally defined credal network where $\mathcal{P}_i$ is the set of all the conditional probability distributions such that $p(X_i = w_l^i|\Pi_i = \pi_j^i, D) \in [\frac{n_{ijl}}{n_{ij}+s}, \frac{n_{ijl}+s}{n_{ij}+s}]$, where s is a global parameter (usually $s = 1$ or $s = 2$). This procedure has a tendency to produce overly impre-

cise intervals when computing conditional imprecise probabilities given observations in these credal networks [21]. There is another alternative application of the IDM to learn the parameters: the global application [21] which produces more precise results, but it is more difficult to compute with it. There is a solution procedure for the naive credal classifier [21], but there is no efficient algorithm available for computing at the general case.

Sometimes the most natural result of learning is a family of graphs instead of a single one [4, 10]. To encompass this case, we define a *multigraph credal network* as a finite set a credal networks, i.e. a set of graphs and with each graph having a set of probability distributions that factorize according to the graph. In our approach, usually each graph will have a single probability distribution associated to it.

A multigraph credal network will be said to be *locally defined* when the variables can be sorted in such a way that for each variable X_i we have a family of possible sets of parents for this variable $\{\Pi_i^1, \ldots, \Pi_i^{l_i}\}$ were Π_i^j is always included in the set of variables preceding X_i in the given order, and for each set of parents Π_i^j we have a set of conditional probability distributions for X_i given Π_i^j. A locally defined multigraph credal network will have an associated multigraph credal network: we only have to consider all the credal networks obtained by selecting a possible set of parents Π_i^j and its corresponding set of conditional probability distributions for each variable X_i. For a locally defined multigraph we have to give the family of possible parents and for each parent the set of possible conditional distributions. However, if l_i is the number of possible parents for X_i, the number of credal networks that can be obtained by selecting a possible set of parents for each variable is $\prod_{i=1}^m l_i$, which can be a very large number. Then it is clear that the local definition can be much more compact than the multigraph definition. Furthermore, it is possible to directly make inferences with the local definition [10].

To locally define a multigraph credal network, we need to specify an order of the variables in such way that the set of parents of a variable are selected from preceding variables. If we did not specify this order then, there could be two variables X_i and X_j, a possible set of parents of X_i, Π_i, containing X_j and a possible set of parents of X_j, Π_j, containing X_i. The two set of parents Π_i and Π_j are not compatible as they give rise to a cycle. More complex cycles could be created by circular relationships. So modularity would be lost, as we could not locally specify a family of parent sets for each variable without considering additional global restrictions for them.

4 Learning Multigraph Credal Networks

A directed acyclic graph G about variables $\mathbf{X}$ will be represented by a finite list $G = (\Pi_1, \ldots, \Pi_m)$ of the set of parents of the different variables. The set of all the acyclic directed graphs will be denoted as $\mathcal{G}$.

To learn a multigraph credal network, we will follow a variant of the Imprecise Sample Size Dirichlet Model (ISSDM) introduced in [10]. As in the case of learning precise Bayesian networks, our model is based on assuming prior distributions $D(\alpha^i, \ldots, \alpha^i)$ for the conditional probability distributions $p(X_i|\Pi_i = \pi_j^i)$, where $\alpha_i = \frac{s}{q_i k_i}$ and $s > 0$ is the global equivalent sample size. But now instead of an unique global equivalent sample size $s > 0$, it will be assumed that there is a finite set S of possible equivalent sample sizes[1].

In [10] two basic applications of the Imprecise Sample Size Dirichlet Model (ISSDM) have been considered:

- *The global approach.-* Given a graph G, the set of prior distributions for the conditional probabilities $p(X_i|\Pi_i = \pi_j^i)$ is the set of Dirichlet distributions $D(\alpha^i, \ldots, \alpha^i)$ that are obtained by considering a value $s \in S$ and computing $\alpha^i = \frac{s}{q_i k_i}$.
- *The local approach.-* Given a graph G, the set of prior distributions for the conditional probabilities $p(X_i|\Pi_i = \pi_j^i)$ is the set of Dirichlet distributions $D(\alpha^i, \ldots, \alpha^i)$ that are obtained by considering a value $s_i \in S$ for each variable X_i and computing $\alpha^i = \frac{s_i}{q_i k_i}$.

The difference between the local and the global approach is that in the global we have to select the same $s \in S$ to compute the weights of the prior distribution for each variable X_i, and in the local approach, we can select a different value $s_i \in S$ for each variable. The number of different prior distributions for the parameters is higher in the local approach than in the global approach. In the global approach it is $|S|$ (the cardinal of S) but in the local approach is $|S|^m$.

In this paper we will follow the local approach. So, given a graph G, we will consider that there is a prior distribution about the values of the conditional probabilities for each $\mathbf{s} = (s_1, \ldots, s_m) \in S^m$, where the prior distribution for the conditional probabilities of variable X_i is $D(\alpha^i, \ldots, \alpha^i)$ where $\alpha^i = \frac{s_i}{q_i k_i}$. There are reasons for assuming the possibility of a different $s_i \in S$ for each variable instead of the same $s \in S$ for all the variables. The value $s_i \in S$ determines

[1] In [10] it was assumed that S was an interval, but finally for computational reasons it was approximated by a finite set.

the prior probabilities for the conditional probability distribution of X_i: with small values of s_i the prior Dirichlet distribution is concentrated in the extremes (close to 0 and 1) and with high values of s_i the prior distribution is concentrated around the uniform distribution (all the probabilities close to $\frac{1}{k_i}$). If all the values are the same for all the variables, we are assuming that if prior density of X_i is concentrated in the extreme values (small s) so is the prior probability about the conditional probabilities for X_j. Assuming a different values of s_i for different variables, we are expressing that the probabilities for one variable can be extreme, while for other variable they can be close to the uniform distribution. In [3], we elaborate on these arguments when using precise Bayesian methods.

Given $\mathbf{s} \in S^m$, we can compute the score of a graph using (2) and to estimate the conditional probabilities associated to a graph with expression (3), with $\alpha^i = \frac{s_i}{q_i k_i}$. To emphasize the dependence upon $\mathbf{s}$, the score will be denoted by $Score(G|D, \mathbf{s})$ and the estimated conditional probability by $p_{\mathbf{s}}(X_i|\Pi_i, D)$.

Given an arbitrary set H, an ϵ-contaminated imprecise probability model of the uniform distribution in H where $\epsilon > 0$ is given by the set of all the probability distributions p defined on H and satisfying $p(h) \geq \frac{1-\epsilon}{k}$, where k is the number of elements in H. Equivalently this model can be characterized by the inequalities $\frac{p(h)}{p(h')} \geq \beta_\epsilon$, where $\beta_\epsilon = \frac{1-\epsilon}{1-\epsilon+k\epsilon}$. This is a convex set of probabilities and there is an extreme probability, p^ϵ_h, for each $h \in H$, given by $p^\epsilon_h(h) = \frac{1-\epsilon}{k} + \epsilon$ and $p_h(h') = \frac{1-\epsilon}{k}$ if $h' \neq h$. This probability can be expressed as a convex combination: $p^\epsilon_h = \epsilon p_h + (1-\epsilon)p_u$, where p_h is the probability degenerated on h (assigning probability 1 to h) and p_u is the uniform distribution. p^ϵ_h will be called the probability that *concentrates mass* ϵ on h and it is the probability for which the probability of h is maximized.

In [10] it was considered that the information on S^m was vacuous, but in this paper additional assumptions will be made.

- There will be an imprecise prior probability for the graph and the equivalent sample size vector, i.e. in $\mathcal{G} \times S^m$. The following options will be considered:
  - An ϵ-contaminated model of the uniform distribution in $\mathcal{G} \times S^m$, where $\epsilon > 0$. The associated set of probability distributions will be denoted as $\mathcal{P}_1$.
  - An ϵ-contaminated model of the uniform distribution in the space of directed acyclic graphs $\mathcal{G}$ and for each graph $G \in \mathcal{G}$ we have an uniform distribution in S^m conditioned to G. It will be denoted as $\mathcal{P}_2$.
- The set of possible decisions, $\mathcal{D}$, can be one of the following options:
  - The set of graphs $\mathcal{D} = \mathcal{G}$.
  - The set of graphs and equivalent sample sizes: $\mathcal{D} = \mathcal{G} \times S^m$.
- The utility function when $\mathcal{D} = \mathcal{G}$ is $U : \mathcal{G} \times \mathcal{D} \to [0, 1]$, given by

$$U(G, G') = \begin{cases} 1 & \text{if } G = G' \\ 0 & \text{otherwise} \end{cases},$$

  where G is the true graph and G' our decision.

  If $\mathcal{D} = \mathcal{G} \times S^m$, the utility function is $U : \mathcal{G} \times S^m \times \mathcal{D} \to [0, 1]$ given by:

$$U(G, \mathbf{s}, G', \mathbf{s}') = \begin{cases} 1 & \text{if } G = G', \mathbf{s} = \mathbf{s}' \\ 0 & \text{otherwise} \end{cases}$$

  where G is the true graph and $\mathbf{s}$ the true equivalent sample size and $(G', \mathbf{s}')$ our decision.

A decision $d \in \mathcal{D}$ is said to be *dominated* by another decision $d' \in \mathcal{D}$ if and only if for any possible probability distribution, p, associated to the problem we have that $p(U(., d)|D) < p(U(., d')|D)$, where $U(., d)$ is the function that assigns to each graph G (or pair $(G, \mathbf{s})$) the value $U(G, d)$ (or $U(G, \mathbf{s}, d)$) and $p(U(., d)|D)$ stands for the prevision or mathematical expectation of this function with respect to p conditioned to the dataset D.

In these conditions, *learning* is defined as the computation of all the undominated decisions. Assume that a probability p has been fixed in $\mathcal{G} \times S^m$. If $\mathcal{D} = \mathcal{G}$, then if $d = G$, we have that $p(U(., d)|D) = p(G|D) = p(d|D)$, as $U(., d)$ is a function in $\mathcal{G}$ that is equal to 1 in G and 0 otherwise. Analogously, in the case of $\mathcal{D} = \mathcal{G} \times S^m$ and $d = G \times \mathbf{s}$, we also obtain $p(U(., d)|D) = p(G \times \mathbf{s}|D) = p(d|D)$.

Depending on the different options, we have the following situations.

4.1 $\mathcal{D} = \mathcal{G}$ and $\mathcal{P}_2$ as prior probability

In this case, we have an ϵ-contaminated model in $\mathcal{G}$ and for each graph an uniform distribution in S^m.

Given that $p(U(., G)|D) = p(G|D)$, we have to compute all the graphs G such that for any graph G' there is a probability $p \in \mathcal{P}_2$ such that $p(G|D) \geq p(G'|D)$. Given that we have the ϵ-contaminated model in $\mathcal{G}$,

then this probability exists, if and only if this inequality is satisfied for the probability p^{ϵ}_G in $\mathcal{G}$ that maximizes the probability of G (concentrates the ϵ mass on G).

So we have to compute all the graphs G such that $p^{\epsilon}_G(G|D) \geq p^{\epsilon}_G(G'|D), \forall G' \in \mathcal{G}$, for the probability $p^{\epsilon}_G \in \mathcal{P}_2$.

For any probability $p \in \mathcal{P}_2$, we have that $p(G|D) \propto p(G)p(D|G)$. In this expression, $p(D|G) = \sum_{\mathbf{s}\in S^m} p(D|G,\mathbf{s}) = \frac{1}{|S|^m}\sum_{\mathbf{s}\in S^m} Score(G|D,\mathbf{s})$, as given a graph, we have the uniform distribution in S^m. This probability is fixed and does not depend on the probability $p \in \mathcal{P}_2$ and will be denoted as $AScore(G|D) = \frac{1}{|S|^m}\sum_{\mathbf{s}\in S^m} Score(G|D,\mathbf{s})$.

As for any graph we have that $\frac{p^{\epsilon}_G(G')}{p^{\epsilon}_G(G)} = \beta_\epsilon$, then a graph G is undominated if and only if $AScore(G|D) \geq \beta_\epsilon AScore(G'|D)$ for any graph G'.

If G is the graph maximizing $AScore(G|D)$, then this graph is undominated (β_ϵ is always lower than 1), and another graph G' is undominated if and only if $AScore(G'|D) \geq \beta_\epsilon AScore(G|D)$.

$AScore(G|D)$ is the locally averaged score of a Bayesian network as defined by Cano et al. [3]. It is immediately clear that it is not necessary to average an exponential number of scores as,

$$AScore(G|D) = \frac{1}{|S|^m}\sum_{\mathbf{s}\in S^m} Score(G|D,\mathbf{s}) =$$

$$\prod_{i=1}^{m}\left(\frac{1}{|S|}\sum_{s_i\in S}\prod_{j=1}^{q_i}\frac{\Gamma(k_i.\alpha^i)}{\Gamma(n_{ij}+k_i.\alpha^i)}\prod_{l=1}^{k_i}\frac{\Gamma(\alpha^i+n_{ijl})}{\Gamma(\alpha^i)}\right).$$

In short, in this case the problem is to compute the set of graphs with an averaged score greater or equal to $\beta_\epsilon.MAXAVG$, where $MAXAVG$ is the maximum averaged score of a graph. If $\epsilon = 0$, we have the problem of computing the graph G optimizing the locally average score as considered in [3].

4.2 $\mathcal{D} = \mathcal{G} \times S^m$ and $\mathcal{P}_2$ as prior probability

In this case, we have to select a graph G and a vector of equivalent sample sizes $\mathbf{s} = (s_1, \ldots, s_m)$ such that there is not another pair $(G', \mathbf{s}')$ such that for any probability p we have that $p(G,\mathbf{s}|D) < p(G',\mathbf{s}'|D)$. Given that for any $p \in \mathcal{P}_2$ the prior probability in S^m given G is uniform, we have that $p(G,\mathbf{s}|D) \propto p(G)Score(G|D,\mathbf{s})$. So, a pair $(G,\mathbf{s})$ is undominated if and only if for any pair $(G',\mathbf{s}')$ there is a probability in $p \in \mathcal{P}_2$ for which $p(G)Score(G|D,\mathbf{s}) \geq p(G')Score(G'|D,\mathbf{s}')$. As $Score(G|D,\mathbf{s})$ does not depend on $p \in \mathcal{P}_2$, if this inequality is true for a probability $p \in \mathcal{P}_2$, it will also hold for the probability maximizing the probability of G, p^{ϵ}_G. So this is equivalent that for any $(G',\mathbf{s}')$ we have that $p^{\epsilon}_G(G)Score(G|D,\mathbf{s}) \geq p^{\epsilon}_G(G')Score(G'|D,\mathbf{s}')$. Taking into account that $\frac{p^{\epsilon}_G(G')}{p^{\epsilon}_G(G)} = \beta_\epsilon$ (if $G \neq G'$), this inequality is equivalent to $Score(G|D,\mathbf{s}) \geq \beta_\epsilon Score(G'|D,\mathbf{s}')$ if $G \neq G'$ and to $Score(G|D,\mathbf{s}) \geq Score(G'|D,\mathbf{s}')$ if $G = G'$.

If G is fixed, then $(G,\mathbf{s}')$ dominates $(G,\mathbf{s})$ if and only if $Score(G|D,\mathbf{s}') > Score(G|D,\mathbf{s})$. Then for a pair $(G,\mathbf{s})$ to be undominated, it is necessary that $\mathbf{s} = \arg_{\mathbf{s}'}\max Score(G|D,\mathbf{s}')$.

Let us define $MScore(G|D) = \max_{\mathbf{s}\in S^m} Score(G|D,\mathbf{s})$ and assume that $MAXMAX$ is the maximum of this score in the space of all the graphs and G^* the graph for which this score is obtained. We can prove the following result.

Proposition 1 *A pair $(G,\mathbf{s})$ is undominated if and only if $\mathbf{s} = \arg_{\mathbf{s}'}\max Score(G|D,\mathbf{s}')$ and $MScore(G|D) \geq \beta_\epsilon MAXMAX$.*

Proof: If the pair $(G,\mathbf{s})$ is undominated we know that $\mathbf{s} = \arg_{\mathbf{s}'}\max Score(G|D,\mathbf{s}')$. Also this pair can not be dominated by $(G^*,\mathbf{s}^*)$ where $s^* = arg_{\mathbf{s}'}\max Score(G^*|D,\mathbf{s}')$ and therefore $MScore(G|D) \geq \beta_\epsilon Score(G^*|D,\mathbf{s}^*) = \beta_\epsilon MAXMAX$.

On the other hand, assume $\mathbf{s} = \arg_{\mathbf{s}'}\max Score(G|D,\mathbf{s}')$ and $MScore(G|D) \geq \beta_\epsilon MAXMAX$. If $\mathbf{s} = \arg_{\mathbf{s}'}\max Score(G|D,\mathbf{s}')$ then the pair $(G,\mathbf{s})$ is not dominated by any pair $(G,\mathbf{s}')$ (a pair with the same graph and different vector of equivalent sample sizes). Also $MAXMAX \geq Score(G'|D,\mathbf{s}')$ and therefore for any pair $(G',\mathbf{s}')$ $Score(G|D,\mathbf{s}) \geq \beta_\epsilon MAXMAX \geq \beta_\epsilon Score(G'|D,\mathbf{s}')$, and the pair $(G,\mathbf{s})$ is not dominated by any pair $(G',\mathbf{s}')$ with $G' \neq G$ either. So, the pair $(G,\mathbf{s})$ is undominated. □

It is important to remark that $MScore(G|D)$ can be locally computed as

$$MScore(G|D) = \max_{\mathbf{s}\in S^m} Score(G|D,\mathbf{s}) =$$

$$\max_{\mathbf{s}\in S^m}\prod_{i=1}^{m}\left(\prod_{j=1}^{q_i}\frac{\Gamma(k_i.\alpha^i)}{\Gamma(n_{ij}+k_i.\alpha^i)}\prod_{l=1}^{k_i}\frac{\Gamma(\alpha^i+n_{ijl})}{\Gamma(\alpha^i)}\right) =$$

$$\prod_{i=1}^{m}\left(\max_{s_i\in S}\prod_{j=1}^{q_i}\frac{\Gamma(k_i.\alpha^i)}{\Gamma(n_{ij}+k_i.\alpha^i)}\prod_{l=1}^{k_i}\frac{\Gamma(\alpha^i+n_{ijl})}{\Gamma(\alpha^i)}\right).$$

In short, in this case the problem is to compute the set of graphs with a maximum score greater or equal

than $\beta_\epsilon.MAXMAX$, where $MAXMAX$ is the optimal value of the maximum score of a graph and for each one of these graphs we have to determine the vector $\mathbf{s}$ maximizing the score. If $\epsilon = 0$, we have the problem of computing the graph $(G, \mathbf{s})$ optimizing the score $MScore(G|D, \mathbf{s}')$. This approach has been considered by Steck [16] to minimize the effect in the learned structure of a Bayesian network of the equivalent sample size parameter. However, in that paper a continuous set of possible parameters S is considered and a global approach is applied: the same parameter s must be applied to the conditional probability of each variable X_i.

4.3 $\mathcal{D} = \mathcal{G} \times S^m$ and $\mathcal{P}_1$ as prior probability

This case is similar to the one considered in Subsection 4.2. Now, we have that $p(G, \mathbf{s}|D) \propto p(G, \mathbf{s})Score(G|D, \mathbf{s})$ and we have an ϵ-contaminated model in $\mathcal{G} \times S^m$. If a pair $(G, \mathbf{s})$ is dominated by another pair $(G', \mathbf{s}')$, then this is equivalent to $p(G, \mathbf{s})Score(G|D, \mathbf{s}) < p(G', \mathbf{s}')Score(G'|D, \mathbf{s}')$ for any probability $p \in \mathcal{P}$, which given the structure of $\mathcal{P}_1$ is equivalent to $p^\epsilon_{G,\mathbf{s}}(G, \mathbf{s})Score(G|D, \mathbf{s}) < p^\epsilon_{G,\mathbf{s}}(G', \mathbf{s}')Score(G'|D, \mathbf{s}')$ where $p^\epsilon_{G,\mathbf{s}}$ is the probability in $\mathcal{P}_1$ maximizing the probability of $(G, \mathbf{s})$. And this is equivalent to $Score(G|D, \mathbf{s}) < \beta_\epsilon Score(G'|D, \mathbf{s}')$.

It is immediately clear that for a graph G there is a pair $(G, \mathbf{s})$ that is undominated if and only if $MScore(G|D) \geq \beta_\epsilon MAXMAX$. The difference with the computations in Subsection 4.2, is that for a given undominated graph G, now there can be several vectors of parameters $\mathbf{s} \in S^m$ such that $(G, \mathbf{s})$ is undominated: all the pairs for which $Score(G|D, \mathbf{s}) \geq \beta_\epsilon.MAXMAX$.

We can proceed as follows: we can compute the set of graphs with a $MScore(G|D)$ greater or equal than $\beta_\epsilon.MAXMAX$ as in the previous case. Then for each graph G we compute the set S' of parameters $\mathbf{s}$ such that $Score(G|D, \mathbf{s}) \geq \beta_\epsilon MAXMAX$. This computation can be difficult as the number of elements in $\mathbf{s} \in S^m$ is exponential and the problem can not be decomposed by computing a set of components, S_i, for each variable X_i and then computing $S' = S_1 \times \cdots \times S_m$. Whether a component s_i belongs to an undominated vector $\mathbf{s}$ depends on the other components in the vector and can not be separately computed for each variable.

4.4 $\mathcal{D} = \mathcal{G}$ and $\mathcal{P}_1$ as prior probability

This case poses an additional difficulty compared to above situations. A graph G is undominated if and only if for each graph G' there is a probability p such that $p(G|D) \geq p(G'|D)$. The difference is that now the probability p can depend on the graph G'. In previous cases, the problem could be simplified as it could be shown that if this happened we could select the same probability for any graph: the probability p^ϵ_G maximizing the probability of G. But this simplification is not possible in this case. A possible solution is to concentrate in the set *e-admissible* decisions [9]: a graph G is e-admissible if it maximizes $p(G|D)$ for a possible probability p. All the e-admissible decisions are undominated but not the reverse.

In the following we shall concentrate in computing e-admissible solutions. If G' maximizes the conditional probability $p(.|D)$ with p in a convex set $\mathcal{P}_1$, then G' will also be optimal for one extreme probability $p_{(G,\mathbf{s})} \in \mathcal{P}_1$. So we shall concentrate in finding the graphs that optimize the posterior probability for extreme probabilities.

Let us consider $p_{(G,\mathbf{s})}(G'|D)$ the posterior probability of graph G' when the prior probability in $\mathcal{G} \times S^m$ is $p_{(G,\mathbf{s})}$. We have the following situations:

- If $G \neq G'$, then

  $$p_{(G,\mathbf{s})}(G'|D) \propto \sum_{\mathbf{s}' \in S^m} p_{(G,\mathbf{s})}(G', s')p_{(G,\mathbf{s})}(D|G', s') = \sum_{\mathbf{s}' \in S^m} \frac{1-\epsilon}{k} Score(G'|D, \mathbf{s}') = \frac{1-\epsilon}{k'} AScore(G'|D),$$

  where $k' = \frac{k}{|S|^m}$. In the above equalities, we have that $p_{(G,\mathbf{s})}(D|G', s')$ is equal to $Score(G'|D, \mathbf{s}')$ as this conditional probability does not depend of the prior probability in $\mathcal{G} \times S^m$.

- If $G = G'$, then

  $$p_{(G,\mathbf{s})}(G|D) \propto \sum_{\mathbf{s}' \in S^m} p_{(G,\mathbf{s})}(G', s')p_{(G,\mathbf{s})}(D|G, s') = \sum_{\mathbf{s}' \in S^m} \frac{1-\epsilon}{k} Score(G|D, \mathbf{s}') + \epsilon Score(G|D, \mathbf{s}) = \frac{1-\epsilon}{k'} AScore(G|D) + \epsilon Score(G|D, \mathbf{s})$$

  where $k' = \frac{k}{|S|^m}$

Given above expressions, it can immediately be seen that if G' maximizes $p_{(G,\mathbf{s})}(G'|D)$ for one extreme probability, then $G' = G$. In that case, we have that $p_{(G',\mathbf{s})}(G'|D) = \frac{1-\epsilon}{k'} AScore(G'|D) + \epsilon Score(G'|D, \mathbf{s})$ and $p_{(G',\mathbf{s})}(G|D) = \frac{1-\epsilon}{k'} AScore(G|D)$, for $G \neq G'$. Then, if G' maximizes $p_{(G,\mathbf{s})}(G'|D)$, it will also do it when $\mathbf{s} = \arg_{\mathbf{s}'} \max Score(G'|D, \mathbf{s}')$, and in this case, $p_{(G',\mathbf{s})}(G'|D) = \frac{1-\epsilon}{k'} AScore(G'|D) +$

$\epsilon MScore(G'|D)$. So, we can express the condition for G' e-admissible: $\frac{1-\epsilon}{k'} AScore(G'|D) + \epsilon MScore(G'|D) \geq \frac{1-\epsilon}{k'} AScore(G|D),\ \forall G \in \mathcal{G}$.

To compute the set of e-admissible graphs, we can start by computing the graph G^* maximizing $AScore(G|D)$. It is clear that this graph is e-amissible and that a graph G' is non dominated if and only if $\frac{1-\epsilon}{k'} AScore(G'|D) + \epsilon MScore(G'|D) \geq \frac{1-\epsilon}{k'} AScore(G^*|D)$, which is equivalent to $AScore(G'|D) + \frac{\epsilon k'}{1-\epsilon} MScore(G'|D) \geq AScore(G^*|D)$. If we call $MAMScore_\epsilon(G'|D)$ to $AScore(G'|D) + \frac{\epsilon k'}{1-\epsilon} MScore(G'|D)$. The computational problem is similar to the previous ones: to compute a family of graphs with a score above a threshold.

5 Algorithms

In this section we discuss some procedures to compute the set of undominated graphs or undominated graphs and equivalent sample sizes. In all the situations the procedure involves the computation of one graph maximizing a specific score, $Score1$ (in some cases the averaged and in others the maximum score). Then we have to compute all the graphs with a score ($Score2$) above B, where B is a value depending of the previous computed optimum. If A is the optimum of $Score1$, then this value will be denoted as $B = f(A)$. The scores of the first and second stages are not necessarily the same as in the case of $\mathcal{D} = \mathcal{G}$ and $\mathcal{P}_1$ as prior probability.

To carry out this task, we shall propose approximate and exact algorithms. Approximate algorithms can be based on algorithms that try to visit a significant set of networks with a high score as in Bayesian model averaging procedures [7]. In this paper we have considered a modification of the algorithms presented in Masegosa, Moral [11]. The procedure has two stages:

- First, it computes a graph G^* with a high value $Score1$ using the Max-Min hill climbing algorithms by Tsamardinos et al. [17], a state of the art algorithm to learn Bayesian networks. Compute $A = Score1(G^*|D)$ and $B = f(A)$.
- In a second step, a Markov Chain Monte-Carlo procedure is employed to compute the family of undominated graphs. It starts with a family $\mathcal{H}$ of graphs containing only G^*, the current graph G is initially equal to G^*. Then it randomly generates a graph G' from the neighboring graphs of the current graph G (the graphs obtained by adding, deleting, or reversing a link of G) and computes $Score2(G'|D)$. If $Score2(G'|D) \geq B$, then the current graph is set to G' and is added to $\mathcal{H}$.

  It also computes $Score1(G'|D)$ (there is no additional computation if $Score1 = Score2$) and if $Score1(G'|D) > A$, then we make $B = f(Score1(G'|D))$ and remove from $\mathcal{H}$ all the graphs G with $Score2(G|D) < B$. This step is done because the first stage is an approximate algorithm, and in this step we are visiting graphs with a high $Score2$. As the different scores are strongly correlated, there is a possibility that the computed optimum is improved by one of the graphs visited at this stage. This is specially convenient when $Score1 = Score2$ and there is not necessity in doing additional computations.

Recently, exact algorithms for computing the Bayesian networks maximizing a decomposable score have been presented [14, 6]. In this paper, we will concentrate on the A* algorithm proposed by Yuan et al. [20] and we will indicate how it can be generalized to compute the full family of undominated graphs in the case of a decomposable score such that $Score1 = Score2$. $Score$ is decomposable if and only if we can express $Score(G|D) = \prod_{i=1}^{m} Score_i(\Pi_i|D)$, i.e. it can be expressed as a product of scores associated to each variable and its sets of parents in the graph. This covers all the situations except the last one ($\mathcal{D} = \mathcal{G}$ and $\mathcal{P}_1$ as prior probability), as the score $MAMScore_\epsilon(G|D) = AScore(G|D) + \frac{\epsilon k'}{1-\epsilon} MScore(G|D)$ can not be expressed as a product of scores for each variable[2].

In the following $Score(G|D)$ is any decomposable score function and $LScore(G|D)$ is the logarithm of this score. We have that $LScore(G|D) = \sum_{i=1}^{m} LScore_i(\Pi_i|D)$, where $LScore_i(\Pi_i|D)$ is the logarithm of $Score_i(\Pi_i|D)$. First, we describe the A* algorithm for precise Bayesian networks (to compute the graph maximizing the score). It is assumed that we have a function $BestScore(X_i, R_i)$ that computes the optimal value of $LScore(\Pi_i|D)$ for $\Pi_i \subseteq R_i$ (see [20] for efficient procedures for this task) where R_i is a subset of $\mathbf{X}$. The learning problem is formulated as a search of the best path between two states. The set of states is the family of all the possible subsets $\mathbf{Y} \subseteq \mathbf{X}$. The initial state is the empty set and the final state is the full set $\mathbf{X}$. The set of children of a state $\mathbf{Y}$ is the set of states $\mathbf{Y} \cup \{X_i\}$, where $X_i \notin \mathbf{Y}$. The utility of going from one state $\mathbf{Y}$ to one of its children $\mathbf{Y} \cup \{X_i\}$ is $BestScore(X_i, \mathbf{Y})$. The utility of a path is the sum of the utilities of each one of its arcs, and the problem is to compute the path maximizing the utility of going from the initial state to the final

[2] As it is a linear combination of decomposable scores this does not pose any problem for the local computation under local changes (we locally compute $AScore(G|D)$ and $MScore(G|D)$ which are decomposable).

one. For that the A* algorithm is used with heuristic function $h(\mathbf{Y}) = \sum_{X_i \notin \mathbf{Y}} BestScore(X_i, \mathbf{X} \setminus \{X_i\})$. It can be proved that this heuristic is admissible [20] as it is an optimistic evaluation of the utility of the rest of the path. In these conditions a search procedure that expands the node with maximum value of $g(\mathbf{Y}) = u(\mathbf{Y}) + h(\mathbf{Y})$, where $u(\mathbf{Y})$ is the utility of the best path arriving to $\mathbf{Y}$, is guaranteed to find the optimal path the first time it chooses the final state $\mathbf{X}$ to be expanded.

A path from the initial state to the goal represents an ordering of the variables (if we go from $\mathbf{Y}$ to $\mathbf{Y} \cup \{X_i\}$, then all the variables from $\mathbf{Y}$ are predecessors of $\{X_i\}$). The utility of this path is the logarithm of the score of the best network that can be obtained restricted to this order (a variable can not be a parent of one of its predecessors). Knowing this path, we obtain the order with best score. The optimal graph can be found by considering for each variable X_i the first time this variable appears in the path from node $\mathbf{Y}$ to node $\mathbf{Y} \cup \{X_i\}$. The set of parents of X_i is the subset of $\mathbf{Y}$ for which the optimal value $BestScore(X_i, \mathbf{Y})$ is obtained.

In order to adapt this algorithm to our problem we have to compute all the graphs with an score greater than or equal to $\log(f(A))$, where A is the graph with best score. A first approximation can be to continue after the final node $\mathbf{X}$ has been expanded, and expands the nodes while $g(\mathbf{Y}) \geq \log(f(A))$ (the value of A is known after the first time the full node is expanded). In this way we obtain a set of paths, and for each path an undominated graph with the same procedure used in the optimal path. With this modification, it is necessary that if the same state is obtained with two different paths we keep the two copies of the state one for each path as they can lead to different solutions[3]. However, in this procedure we do not obtain all the undominated graphs, but the set of orders of the variables such that there is an undominated graph compatible with this order. However, only one undominated graph is obtained for each one these orders. Computing all the undominated graphs given an order can be difficult and perhaps a solution could be to decompose the problem and once an order is considered, to compute a set of different parents for each variable, for example for variable X_i we compute all the set of parents Π_i selected from the predecessors variables $\mathbf{Y}$ (the parent state), such that changing $BestScore(X_i, \mathbf{Y})$ as utility of the arc arriving to X_i by the value $LScore_i(\Pi_i|D)$ the cost of the path is greater than or equal to the threshold ($\log(f(A))$). This procedure has the advantage of decomposing the problem in local problems for each variable, and that we obtain a locally defined credal network. But not all the compatible networks are undominated: it is possible that changing the best parent for X_i or changing the best parent for X_j we obtain undominated graphs, but changing both of them the obtained graph is dominated.

A modification of the A* algorithm can be done in order to compute all the undominated graphs. For that, we change the set of states to the set of pairs $(\mathbf{Y}, \mathbf{T})$, where $\mathbf{T} \subseteq \mathbf{Y}$. $\mathbf{Y}$ will have the same interpretation as above, and $\mathbf{T}$ will be the set of parents of the last variable introduced in $\mathbf{Y}$. The problem starts with $(\emptyset, \emptyset)$ and the final states are $(\mathbf{X}, \mathbf{T})$ where $\mathbf{X}$ is the full set of variables. The children of an state $(\mathbf{Y}, \mathbf{T})$ are all the states $(\mathbf{Y} \cup \{X_i\}, \mathbf{T}')$, where $X_i \notin \mathbf{Y}$ and $\mathbf{T}' \subseteq \mathbf{Y}$. The utility of the arc going from $(\mathbf{Y}, \mathbf{T})$ to $(\mathbf{Y} \cup \{X_i\}, \mathbf{T}')$ is the logarithm of $LScore_i(\mathbf{T}'|D)$. The heuristic function on one state is computed as above $h(\mathbf{Y}, \mathbf{T}) = h(\mathbf{Y}) = \sum_{X_i \notin \mathbf{Y}} BestScore(X_i, \mathbf{Y} \setminus \{X_i\})$ The algorithm first computes the optimum value A of the score. For that, it works as the A^*, but not expanding all the nodes. $(\mathbf{Y}, \mathbf{T})$ is only expanded to nodes $(\mathbf{Y} \cup \{X_i\}, \mathbf{T})$, where $X_i \notin \mathbf{Y}$ and $\mathbf{T}$ is the subset of $\mathbf{Y} \setminus \{X_i\}$ maximizing $LScore_i(\mathbf{T}|D)$, i.e. the set of parents for which $BestScore(X_i, \mathbf{Y})$ is obtained. In this way, the behavior is very similar to the simple A* algorithm, with the only difference that here we make explicit the best set of parents for each variable. Afterwards, we compute all the undominated graphs, i.e. those with a score greater than or equal to the threshold, $\log(f(A))$. For that, for a node $(\mathbf{Y}, \mathbf{T})$ we expand all the children $(\mathbf{Y} \cup \{X_i\}, \mathbf{T}')$, such that $u(\mathbf{Y}, \mathbf{T}) + LScore_i(\mathbf{T}'|D) + h(\mathbf{Y} \cup \{X_i\}) \geq \log(f(A))$ (the cost of the path to arrive to the state plus the cost of the heuristic is above the threshold). This implies to compute all the set of parents $\mathbf{T}' \subseteq \mathbf{Y}$ with a score greater that or equal to $\log(f(A)) - u(\mathbf{Y}, \mathbf{T}) - h(\mathbf{Y} \cup \{X_i\})$ for a given variable X_i. Existing algorithms to compute $BestScore(X_i, \mathbf{Y})$ can be adapted to this task, as they make almost an exhaustive search in the set of all possible parents of X_i in $\mathbf{Y}$.

The algorithm expands a state with a new variable and a set of parents if the associated partial network could obtain a score above the threshold, by assuming an optimistic evaluation for the score of the rest of the variables $(\mathbf{X} \setminus (\mathbf{Y} \cup \{X_i\}))$. At the end, all the paths arriving to final states $(\mathbf{X}, \mathbf{T})$ represent undominated Bayesian networks (their utility is the log of the score, being the heuristic function equal to 0, so we have an exact value of the utility). In each path the graph is obtained by assigning to variable X_i the set of parents

[3] alternatively, we could maintain a unique state with a set of utilities, one of each path arriving to it. A utility value is active while the value plus the heuristic value is greater than or equal to the threshold. But for simplicity in the exposition, we shall assume that we repeat the states.

T_i in the node $(\mathbf{Y}, T_i)$ for which $X_i \in \mathbf{Y}$ for the first time (starting in the root node).

In the case of $\mathcal{D} = \mathcal{G} \times S^m$ and $\mathcal{P}_2$, with this algorithm we can compute all the graphs for which there is a vector $\mathbf{s}$ of equivalent sample sizes for which $(G, \mathbf{s})$ is undominated. If we want to compute all the undominated pairs $(G, \mathbf{s})$, we can proceed with a further modification of the A* algorithm. In this case, the set of states is the set of all triples $(\mathbf{Y}, \mathbf{T}, s)$ where now s represents the equivalent sample size with which the score of the set of parents $\mathbf{T}$ has been computed. The procedure is completely analogous to the above one (first only expanding the states with s maximizing the score) and then all the states with a cost plus heuristic above the threshold. In that case, in a path to a final state, we record the equivalent sample size of each variable and we have the pair $(G, \mathbf{s})$.

6 Experiments

We have done some experiments to illustrate the behavior of our approach. The experiments are done with the approximate algorithm for the case of $\mathcal{P}_2$ as prior probability. In both cases, we have to compute the set of graphs in which the score is greater or equal than a given threshold: $Score(G|D) \geq \beta_\epsilon$, where $Score(G|D)$ is the average score if $\mathcal{D} = \mathcal{G}$ and the maximum score if $\mathcal{D} = \mathcal{G} \times S^m$. In the last case, the decision involves the vector $\mathbf{s}$ for which the maximum score is obtained, but these data are not reported. We have considered a very well known network, the *Alarm* network [1]. This network has 37 nodes and 46 arcs. We have considered different data samples (50, 100, 500, 1000, 5000) and two different values of β_ϵ (0.8 and 0.9). For each one of them, we have computed the following values: NM number of different learned models (graphs); PI percentage of imprecise links (links appearing in some models and not in others without taking into account the direction) in relation with the total number of possible links (37*36/2); PE percentage of sure extra links (links appearing in all the learned models but not present in the original graph) with respect to the missing links in the original graph (37*36/2 - 46); PM percentage of sure missing links (links appearing in the original graph but missing in all learned networks) with respect to the number of links of the graph (46). Results are reported in Table 1.

We can observe that the imprecision decreases with the sample size (with $N = 5000$ we have almost no imprecision); it is greater if β_ϵ decreases (ϵ increases and introduce more imprecision in the ϵ-contaminated prior); and it is higher when using the $MScore$ (we decide about the graph and about the equivalent sample sizes) than when using the $AScore$ (we only decide about the graph).

Table 1: Results of the experimental evaluation.

Samples	50	100	500	1000	5000
AScore-0.8					
NM	67.00	22.50	10.40	4.40	3.40
PI	3.95	1.44	0.45	0.21	0.05
PE	8.50	5.82	2.18	1.58	0.76
PM	43.04	31.96	12.61	8.26	6.09
AScore-0.9					
NM	34.20	9.20	2.00	2.20	1.80
PI	1.98	0.66	0.06	0.09	0.00
PE	9.35	6.29	2.34	1.53	0.79
PM	51.22	45.78	31.65	27.26	19.30
MScore-0.8					
NM	126.20	36.70	7.10	3.30	2.10
PI	4.95	1.59	0.39	0.26	0.12
PE	6.97	5.98	2.92	2.24	0.98
PM	38.91	30.87	10.65	8.48	5.22
MScore-0.9					
NM	58.60	22.40	2.30	2.30	1.30
PI	2.37	0.87	0.11	0.15	0.05
PE	7.98	6.18	3.10	2.29	0.95
PM	42.61	32.17	11.09	7.83	5.22

On the other hand, it can be seen that the structural errors, PE and PM, strongly improve with the sample size. When β_ϵ decreases there are less missing links but more extra links (we have more models and, in consequence, more links are considered); and, at least for this BN, $MScore$ seems to obtain less structural errors than $AScore$ specially for $\beta_\epsilon = 0.9$ where PM is much higher (although more extensive experiments are needed to evaluate if this trend persists).

7 Conclusions

We have presented a general methodology for learning multigraphs credal networks. Our approach justifies the use of different scores that can be found in the literature and the fact that several networks are the result of the learning task. Even if our set of decisions consists in determining a single graph, it makes sense that the output of the learning task is a set of undominated decisions (graphs) if we have imprecise prior information. Though the problem is computationally more difficult than learning a single network, algorithms have been proposed, both for exact and approximate computation. In the future, we plan to make a more extensive experimentation including the exact algorithms and to compare with the results of learning a single network.

Acknowledgments

This research was jointly supported by the Spanish Ministry of Education and Science under projects TIN2010-20900-C04-01,02, the European Regional Development Fund (FEDER), and the Andalusian Research Program under project P08-TIC-03717. We are very grateful to the Isipta reviewers for their useful comments and suggestions.

References

[1] I.A. Beinlich, H.J. Suermondt, R.M. Chavez, and G.F. Cooper. The Alarm monitoring system: A case study with two probabilistic inference techniques for belief networks. In *Proceedings of the Second European Conference on Artificial Intelligence in Medicine*, pages 247–256. Springer-Verlag, 1989.

[2] W. Buntine. Theory refinement in Bayesian networks. In *Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence*, pages 52–60. Morgan Kaufmann, San Francisco, CA, 1991.

[3] A. Cano, M. Gómez-Olmedo, A. Masegosa, and S. Moral. Locally averaged Bayesian Dirichlet metrics for learning the structure and the parameters of Bayesian networks. *International Journal of Approximate Reasoning*, pages 526–540, 2013.

[4] G. Corani and M. Zaffalon. Credal model averaging: an extension of Bayesian model averaging to imprecise probabilities. In *ECML PKDD 2008: European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, pages 257–271. Springer, 2008.

[5] F.G. Cozman. Credal networks. *Artificial Intelligence*, 120:199–233, 2000.

[6] C.P. de Campos and Q. Ji. Efficient structure learning of Bayesian networks using constraints. *Journal of Machine Learning Research*, 12:663–689, 2011.

[7] N. Friedman and D. Koller. Being Bayesian about network structure. A Bayesian approach to structure discovery in Bayesian networks. *Machine Learning*, 50:95–125, 2003.

[8] D. Heckerman, D. Geiger, and D.M. Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20:197–243, 1995.

[9] I. Levi. *The Enterprise of Knowledge*. MIT Press, London, 1980.

[10] A. Masegosa and S. Moral. Imprecise probability models for learning multinomial chances from data. Applications to learning credal networks. *International Journal of Approximate Reasoning*, Submitted, 2013.

[11] A.R. Masegosa and S. Moral. New skeleton-based approaches for Bayesian structure learning of Bayesian networks. *Applied Soft Computing*, 13:1110–1120, 2013.

[12] R.E. Neapolitan. *Learnig Bayesian Networks*. Prentice Hall, Upper Saddle River, 2004.

[13] J. Pearl. *Probabilistic Reasoning with Intelligent Systems*. Morgan & Kaufman, San Mateo, 1988.

[14] T. Silander and P. Myllymaki. A simple approach for finding the globally optimal Bayesian network structure. In *Proceedings of the 22nd Annual Conference on Uncertainty in Artificial Intelligence (UAI-06)*, pages 445–452. AUAI Press, 2006.

[15] P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction and Search*. Springer Verlag, Berlin, 1993.

[16] H. Steck. Learning the Bayesian network structure: Dirichlet prior versus data. In *Proceedings of the Twenty-Fourth Conference on Uncertainty in Artificial Intelligence (UAI2008)*, pages 511–518. AUAI Press, 2008.

[17] I. Tsamardinos, L. E. Brown, and C. F. Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, 65:31–78, 2006.

[18] P. Walley. *Statistical Reasoning with Imprecise Probabilities*. Chapman and Hall, London, 1991.

[19] P. Walley. Inferences from multinomial data: learning about a bag of marbles (with discussion). *Journal of the Royal Statistical Society, Series B*, 58:3–57, 1996.

[20] C. Yuan, B. Malone, and X. Wu. Learning optimal Bayesian networks using A* search. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI-11)*, pages 2186–2191, 2011.

[21] M. Zaffalon. Statistical inference of the naive credal classifier. In *Proceedings of the Second International Symposium on Imprecise Probabilities and Their Applications*, pages 384–393. Shaker Publishing, 2001.