Dealing with soft evidence in credal networks

José Carlos F. da Rocha[1], Alaine M. Guimarães[1], and Cassio P. de Campos[2]

[1] Universidade Estadual de Ponta Grossa, PR, Brazil
jrocha@uepg.br
[2] Rensselaer Polytechnic Institute Troy, NY, USA

Abstract. The formalism of credal networks can be used to represent imprecision in multivariate probabilistic models. Currently, the algorithms for inference with credal nets allow to compute posterior probability intervals given specific evidence. However, they do not deal with soft evidence. This paper presents an approach to integrate soft evidence in credal networks. The proposal is to convert the soft evidence into constraints that are appended to a multilinear program to perform inferences.

1 Introduction

Knowledge based systems draw their conclusions by making inferences from a knowledge base that represents a formal model of the knowledge of the domain [13]. Since the knowledge expressed in the model can be uncertain, it is necessary to deal with uncertainty during the inference process [16]. The study and the development of methods to obtain conclusions with uncertain knowledge is the aim of *reasoning with uncertainty*, a major subject in artificial intelligence.

The theory of probability has been a common approach to this matter. In this extent many AI systems encode probabilistic knowledge using the formalism of Bayesian networks [14]. This formalism allows to encode multivariate models in a graph-based framework that highlights local probabilistic dependencies; the strength of such dependencies is specified by local conditional probabilities. Given a Bayesian network it is possible to compute the posterior probability of an event of interest - this computational procedure is called belief updating. Algorithms for belief updating handle two kinds of evidence: the *specific evidence* that informs the state of a variable with certainty and the *virtual evidence* that reports a likelihood ratio that associates the observation and its respective variable.

Different of the belief updating, which aims to compute *posterior* probabilities, the inference with *soft evidence* aims to adjust the Bayesian network model in such a way that it agrees with probabilistic information provided by external agents [10]. After the adjustment, belief updating can be employed to compute the probabilities of interest in the adjusted model. This reasoning approach is useful when an intelligent agent wants to update her beliefs so as they become consistent with probabilistic information provided by external agents while the decision models of such agents are not disclosed.

The formalism of credal networks extends the Bayesian networks schema, also relying on directed acyclic graphs and probabilistic dependencies to represent multivariate models, but where numerical parameters are uncertain [5]. Credal networks use sets of probability distributions, also called credal sets [11], to represent the imprecision in the local conditional probabilities. The current algorithms for inference in credal networks do not deal with soft evidence. This paper presents an approach to perform inferences with soft evidence in credal networks by extending the multilinear programming procedure of de Campos and Cozman [6]. The basic idea is to use the Chan and Darwiche's transformation [1] to convert soft evidences in multilinear constraints, which are appended to the multilinear program generated for inferences.

The paper is organized as follows. Section 2 describes the formalism of Bayesian networks and some concepts in evidential reasoning. Section 3 shows a brief introduction about credal sets and credal networks. Section 4 explores an example that illustrates our proposal, while Section 4.1 generalizes our idea. The last section presents our final remarks.

2 Bayesian networks, uncertain reasoning and soft evidence

This section contains a brief review of Bayesian networks, which are the basis for the extended model of credal networks. A Bayesian network [14] $\mathcal{N}$ consists of a triple $(\mathbf{G}, \mathbf{X}, \mathbf{F})$ in which $\mathbf{X}$ is a set of random variables $X_1, ..., X_n$ and $\mathbf{G}$ is a directed acyclic graph (DAG) that expresses dependencies among variables. Each node[3] of $\mathbf{G}$ represents a random variable in $\mathbf{X}$ and each arc highlights a relationship of conditional dependence. Let $pa(X_i)$ be the parents of X_i in the DAG and r the number of joint instantiations of $pa(X_i)$, then $\mathbf{F}$ is a collection of local distribution functions $p(X_i|pa(X_i))$. So, if X_i is a root node it is associated with $p(X_i)$, otherwise it is associated with the conditional distributions $p(X_i|pa(X_i)_1) \ldots p(X_i|pa(X_i)_r)$.

Let $d(X_i)$ be the descendants of X_i in the DAG. The Bayesian network formalism assumes the following condition (Markov condition): each X_i is conditionally independent of $\mathbf{X} \setminus d(X_i) \cup \{X_i\}$ given the states of variables in $pa(X_i)$. Because of this condition, the structure of a Bayesian network encodes a joint probability distribution $p(\mathbf{X})$ that can be evaluated from $\mathcal{N}$ by the expression

$$p(\mathbf{X}) = \prod_{i=1}^{n} p(X_i|pa(X_i)) \,.$$

Given a Bayesian network and a query about a variable X_q, a belief updating procedure computes the posterior (given some evidence) or marginal distribution of X_q. We consider the variable elimination algorithm [18] to perform such computation. This algorithm starts by establishing an ordering on the non-query and

[3] In this text we denote a variable and its associated node with the same symbol.

non-observed variables. According to this ordering, each variable X_i is associated with a data structure called bucket. The bucket of X_i stores the probabilistic functions $p(X_j|pa(X_j))$, with $X_j \in \{X_i\} \cup ch(X_i)$ and where $ch(X_i)$ is the set of children of X_i. Note that each function is stored in the earliest bucket where one of its variables appears.

Now let X_{i_1} be the first variable in the ordering. The main step in X_{i_1}'s bucket is to simplify X_{i_1} by computing [3]:

$$p(ch(X_{i_1})|pa(X_{i_1}), sp(X_{i_1})) = \sum_{X_{i_1}} \prod_{X_j \in \{X_{i_1}\} \cup ch(X_{i_1})} p(X_j|pa(X_j))\,. \qquad (1)$$

The left side of Equation 1 is called the separator of X_{i_1}'s bucket and denotes an unnormalized function from where X_{i_1} was eliminated. Here, $sp(X_{i_1})$ means the spouses of X_{i_1} in the DAG. After the separator has been determined it is stored in the bucket of X_{i_w}, where X_{i_w} is the earliest non-processed variable which bucket contains at least one of the variables of the separator of X_{i_1}'s bucket. Variable elimination proceeds summing out variables until the last bucket in the ordering is processed.

When the last bucket is reached a new bucket is created and associated with X_q, where the last computed separator is stored. Then the desired probability is calculated. We note that variable elimination is applied to the subset of network variables that are relevant to the query (this is found through *d-separation*). Below, we summarize the variable elimination algorithm.

– **Algorithm 1** : Variable elimination
  - input: A Bayesian network $\mathcal{N}$ and a query variable X_q;
  - definitions: let $\mathbf{X^r}$ and $\mathbf{X^e}$ be the irrelevant and the observed variables, respectively;

1. eliminate all irrelevant variables using *d-separation*;
2. define an ordering for the variables in $\mathbf{X} \setminus \{X_q\} \cup \mathbf{X^e} \cup \mathbf{X^r}$;
3. for each variable X_i in the ordering create the bucket B_i;
4. for each bucket B_i in the ordering;
   (a) insert all distributions that contain X_i in B_i;
   (b) multiply the distributions of B_i;
   (c) calculate B_i's separator by sum out X_i from the result of the step 4.b;
   (d) if B_i is not the last bucket in the ordering, store the B_i's separator in the earliest bucket where any of its variables appears ;
5. store the last calculated separator in the bucket of X_q;
6. normalize the functions in X_q's bucket and calculate the query.

2.1 Dealing with evidence

Pearl [14] highlights two kinds of evidence for belief updating in Bayesian networks. The specific evidence describes with certainty the observed state of a variable X_k, while the virtual evidence expresses uncertainty about an observation. If X_k has m categories, an specific evidence informing an observation about the event $x_{k,l}$ is reported as a vector $\phi(X_k) = [\phi_1, ..., \phi_m]$ where $\phi_j = 1$ if $j = l$, otherwise $\phi_j = 0$. In a virtual evidence report, the vector $\phi(X_k)$ specifies a likelihood function such as $\phi_j = p(y_{k,l}|x_{k,l})$ for each $j = 1 \ldots m$. Y_k is an auxiliary variable that encodes the observation of the X_k's values. The specific or virtual evidence can be integrated to variable elimination by multiplying $\phi(X_k)$ in one of the buckets in which X_k appears.

Soft evidence differs from virtual and specific evidence because its information is not related to the observation but to the final beliefs of the decision agent. Take a situation in which we have a joint distribution $p(\mathbf{X})$ that encodes the beliefs of an agent. Now, let X_i be a variable with marginal distribution $p(X_i)$ obtained from $p(\mathbf{X})$. Suppose the agent receives a report from an external agent who is an expert in make predictions about X_i. The report says that the correct distribution for X_i is $p_s(X_i)$. In such situation the first agent would desire to adjust its internal model in such a way that her beliefs about X_i agree with $p_s(X_i)$. This kind of uncertain information is called soft evidence and the adjustment of $p(\mathbf{X})$ is the aim of the Jeffrey's rule. Following this rule, $p(\mathbf{X})$ should be updated to:

$$p_s(\mathbf{X}) = p(\mathbf{X} - \{X_i\}|X_i)\, p_s(X_i) \tag{2}$$

given we suppose that $p(\mathbf{X} - \{X_i\}|X_i)$ is not affected by the evidence.

We note that it is possible to encode a soft evidence $p_s(X_i)$ report as virtual evidence [1]. We compute each ϕ_j as $k\ p_s(x_{i,j})\,/P(x_{i,j})$, where $P(x_{i,j})$ is the original probability of $x_{i,j}$ in $p(\mathbf{X})$ and k is a constant. However, it must be pointed out that, even if we can converse soft evidence to virtual evidence, these kinds of evidence have not the same meaning. A virtual evidence has probabilistic information about an observation while a soft evidence refers to the final belief. Soft and virtual evidence also differ themselves about commutativity [7]. Virtual evidence is commutative and therefore the belief updating with virtual evidence is not sensible to the order in which the evidence is entered. The same does not happen with soft evidence.

3 Credal Networks and Credal Sets

A credal set $K(X_i)$ on a discrete random variable X_i is a set of probability distribution defined on X_i [11]. In this paper we assume that a credal set is a polytope and it is represented by its vertices (extreme distributions) or by a collection of linear constraints on probability assignments. Given $\mathbf{X}$, a joint credal set $K(\mathbf{X})$ is obtained from joint distributions $p(\mathbf{X})$. If $X_i \in \mathbf{X}$, then the marginal credal set $K(X_i)$ can be computed from $K(\mathbf{X})$ by marginalizing each extreme distribution in $K(\mathbf{X})$ and by taking the resulting convex hull [9]. Similarly, given an event y defined on $\mathbf{Y} \subset \mathbf{X}$, it is possible to obtain a conditional credal set $K(\mathbf{X} \setminus \mathbf{Y}|y)$ composed by conditional distributions as $p(\mathbf{X} \setminus \mathbf{Y}|y)$. That can be done by conditioning[4] every extreme point of $K(\mathbf{X})$ on y and by taking the convex hull of such conditionals.

[4] We suppose all conditioning events have positive probability.

Let $\mathbf{Y}$ and $\mathbf{Z}$ be two proper and disjoint subsets of $\mathbf{X}$. The conditional information of $\mathbf{Y}$ given $\mathbf{Z}$ can be organized in many ways [12]. In this paper we assume that such information is given as a collection of separately specified credal sets $\mathbf{Q}(\mathbf{Y}|\mathbf{Z}) = \{K(\mathbf{Y}|z_0), \ldots, K(\mathbf{Y}|z_t)\}$ where z_k is a joint event on $\mathbf{Z}$. That is, $\mathbf{Q}(\mathbf{Y}|\mathbf{Z})$ is a collection that contains one conditional credal set defined on $\mathbf{Y}$, for every conjunction of the variables in $\mathbf{Z}$.

Both time and space complexities of inferences in joint credal sets make difficult to utilize the credal set theory in the development of real applications [15]. To mitigate such difficulties, the graph-based formalism of the credal networks [5] provides an implicit representation for joint credal sets. Basically, a credal network is a DAG where each node X_i is associated with a variable and every variable X_i is associated with a list of *local* credal sets relating it to its parents. Here we consider credal networks with separately specified credal sets, so a node X_i is associated to $\mathbf{Q}(X_i|pa(X_i))$. The Figure 1 shows the topology of a multiconnected credal network. The collections of credal sets in this network are $\mathbf{Q}(X_1)$, $\mathbf{Q}(X_2|X_1)$, $\mathbf{Q}(X_3|X_1)$, $\mathbf{Q}(X_4|X_2, X_3)$ and $\mathbf{Q}(X_5|X_4)$.

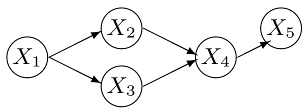

Fig. 1. A simple credal network example.

It is important to note that there is no unique interpretation for the conditional independence concept in the credal set theory. There are several different definitions, which makes necessary to select the one that is more suitable for the desired application [2]. In this work we consider the integration of soft evidence when the concept of *strong independence* is applicable [4]. Two variables X and Y are strongly independent when every vertex $p(XY)$ obeys stochastic independence of X and Y. That is, given $K(XY)$, each vertex satisfies $p(X|Y) = p(X)$ and $p(Y|X) = p(Y)$. Thus, we consider credal networks where every variable is *strongly independent* of its nondescendants nonparents given its parents. In this case the joint credal set encoded in a credal network is called a *strong extension*, which is the largest credal set that agrees with the strong independence assumptions posed in the network.

An *inference* in a credal network calculates bounds for the probability of some event of interest, using the associated extension. If X_q is a query variable, $x_{q,a}$ is the event of interest and $\mathbf{E}$ represents the evidence, an inference must determine the interval $[\underline{P}(x_{q,a}|\mathbf{E}), \overline{P}(x_{q,a}|\mathbf{E})]$ where $\underline{P}(x_{q,a}|\mathbf{E})$ and $\overline{P}(x_{q,a}|\mathbf{E})$ are called lower and upper probabilities, respectively. de Campos and Cozman [6] have presented an algorithm for exact and approximate inference in credal networks. Their method achieves the solution by generating a multilinear program [8] which constraints are obtained from: (a) a symbolic manipulation of the variable elimination procedure, and (b) relationships originated from the credal network and probability theory. In order to deal with the multilinear program, they apply a branch-and-bound solver [17] to determine the maximum and minimum for the probability of the event of interest. The translation procedure that generates the multilinear program is summarized in the Algorithm 2.

- **Algorithm 2**: translate a credal network inference into a multilinear program.
  - input: a credal network $\mathcal{C}$ and the query event $x_{q,t}$
  - definitions : a variable $w_{a^i_j}$ is associated to $P\left(a^i_j\right)$, where a^i_j is a joint event defined on a set of variables $\mathbf{A}_i$; take $w_{a^j_u|b^j_v}$ as the probability of $A_j = a^j_u$ given $B_j = b^j_v$; let $\mathbf{R}$ be a collection of constraints;

1. initialize $\mathbf{R}$ as an empty collection and apply the *d-separation* procedure;
2. let $\mathbf{X^r}$ be the irrelevant variables and $\mathbf{X^e}$ the observed variables; generate an ordering for the variables in $\mathbf{X} \setminus \{X_q\} \cup \mathbf{X^e} \cup \mathbf{X^r}$;
3. for each variable X_i in the ordering generate the bucket B_i;
4. for each bucket B_i in the ordering, generate the expressions related to X_i;
   (a) take every probabilistic function where X_i appears and name these functions by $f_1(\mathbf{A}_1|\mathbf{B}_1)\ldots f_d(\mathbf{A}_d|B_d)$; note that $X_i \in \mathbf{A}_j \cup \mathbf{B}_j$ and $\mathbf{A}_j \cap \mathbf{B}_j \equiv \emptyset$ for $j = 1..d$. Observe also that these functions come from the network constraints or bucket elimination;
   (b) for each $f_j(A_j|B_j)$:
      - for each b^j_v, add the constraint $\sum_u w_{a^j_u|b^j_v} = 1$ to $\mathbf{R}$;
   (c) let $\mathbf{A}_s \cup \mathbf{B}_s$ be the variables in the separator and $w_{a^s_u|b^s_v}$ means $P(a^s_u|b^s_v)$; for each a^s_u and b^s_v append $\sum_{X_i=x_{i,1}}^{m_i} \prod_{j=1}^{d} w_{a^j_u|b^j_v} = w_{a^s_u|b^s_v}$ into $\mathbf{R}$;
5. take the credal sets in $\mathcal{C}$ and add their linear constraints to $\mathbf{R}$;
6. repeat the process above for the bucket of X_q;
7. select the variable $w_{q,t}$ in X_q's bucket as the objective function to be minimized/maximized.

4 Computing probability intervals with soft evidence

In this section we deal with a situation where an agent wants to compute the probability interval for some interest event $x_{q,a}$ given soft evidence. We start with a credal network $\mathcal{C}$ with collections $\mathbf{Q}(X_1|pa(X_1))\ldots,\mathbf{Q}(X_n|pa(X_n))$ and a soft evidence defined as a marginal $p_s(X_o)$. The goal is to compute $[\underline{P}(x_{q,a}), \overline{P}(x_{q,a})]$ in such a way that the constraints imposed by the soft evidence on the agent's final beliefs are taken in consideration.

Consider for example the computation of the lower bound of the probability of $x_{5,0}$ in the network $\mathcal{C}_1$ (Figure 1) given $p_s(X_2)$. We start by applying the Algorithm 2 to generate a preliminary multilinear program. Initially we proceed as if we wanted to compute $\underline{P}(x_{5,0})$ without considering the soft evidence. We minimize the following objective function:

$$\underline{P}(x_{5,0}) = \min P(x_{5,0}) \;:\; s.t.\ \mathbf{W}, \tag{3}$$

where $\mathbf{W}$ is the set of constraints.

Some constraints of $\mathbf{W}$ generated by this preliminary step are listed below:

$$P(x_{5,0}) = \sum_{i=0;}^{1} P(x_{5,0}|x_{4,i}) \cdot P(x_{4,i}) \,; \quad P(x_{4,j}) = \sum_{i=0;}^{1} P(x_{4,l}, x_{3,i}) \,;$$

$$P(x_{2,j}, x_{3,k}) = \sum_{i=0;}^{1} P(x_{2,j}|x_{1,i}) \, P(x_{3,j}|x_{1,i}) \cdot P(x_{1,i}) \,;$$

$$P(x_{4,j}, x_{3,k}) = \sum_{i=0;}^{1} P(x_{4,j}|x_{2,i}, x_{3,k}) \cdot P(x_{2,i}, x_{3,k}) \,.$$

The next step is to take the constraints associated with the soft evidence in consideration. That is, we must append the restrictions induced by $p_s(X_2)$ to $\mathbf{W}$. This is similar to the transformation of Chan and Darwiche [1]: define a virtual evidence report $\varphi(X_2)$ where $\varphi_{x_{2,j}} \cdot P(x_{2,j}) = P_s(x_{2,j})$ and insert it in the appropriated bucket in $\mathbf{W}$. We proceed this last operation by updating the expression related to X_2's as:

$$P_u(x_{4,j}, x_{3,k}) = \sum_{i=0;}^{1} P(x_{4,j}|x_{2,i}, x_{3,k}) \cdot P(x_{2,i}, x_{3,k}) \cdot \varphi_{x_{2,i}} .$$

The new value, $P_u(x_{4,j}, x_{3,k})$, substitutes $P(x_{4,j}, x_{3,k})$ in the constraint set.

In order to exploit the probabilistic knowledge related to $P(x_{2,0})$ and $P(x_{2,1})$ it is necessary to generate the constraints associated with them. That can be achieved by using a modified version of the Algorithm 2, where instead of maximizing $P(x_{2,0})$, we just generate the constraints that define $P(x_{2,0})$. We call this modified algorithm, which has basically the same structure of the Algorithm 2, as Algorithm 2b. The difference appears in the last step, which is changed to:

– select the variable $w_{q,t}$ in X_q's bucket and add the constraint $w_{q,t} = P_s(x_{q,t})$ for each category of X_q.

Finally, we add the normalization equation $P(x_{2,0}) + P(x_{2,1}) = 1$. Note that some probabilities can appear many times during the generation of the multilinear constraints. In the example, expression $P(x_{2,0})$ appears twice. So, it is necessary to keep track of variables names so as each occurrence is denoted by the same optimization variable in the multilinear program. After all, the desired probabilities may be obtained by applying the same branch-and-bound solver of de Campos and Cozman [6].

4.1 The general approach

This section generalizes the previous discussed procedure by outlining the steps of the proposed method. The procedure, described here as Algorithm 3, starts with a credal network $\mathcal{C}$ and a set of soft evidences $p_s(X_{o_1}), \ldots p_s(X_{o_s})$ about some variables in $\mathbf{X}$. Two rules are assumed:

- the evidences are independent of each other;
- the evidences are integrated simultaneously.

The aim is to compute $[\overline{P}(x_{q,a})\,, \underline{P}(x_{q,a})]$ subject to the soft evidence data. Without loss of generality, Algorithm 3 is presented for the case where we want to compute $\underline{P}(x_{q,t})$.

- **Algorithm 3** : Multilinear inference with soft evidence
  - input: a credal network $\mathcal{C}$ related to $\mathcal{N}$, a query event $x_{q,a}$ and soft evidences $p_s(X_{o_1})\,, \ldots p_s(X_{o_s})$;
  - output: $\underline{P}(x_{q,a})$

1. use the Algorithm 2 to generate the multilinear program for $\underline{P}(x_{q,a})$; consider every X_{o_i} as if it was observed;
2. for each soft evidence $p_s(X_{o_i})$
   (a) update a bucket in which X_o appears by appending terms from the report $\varphi_{X_{o_i}}$; rename the ocorrences of the separator entries to $P_u(\cdot)$;
   (b) for each $x_{o_i,j}$ generate the constraint $\varphi_{x_{o_i,j}} \cdot P(x_{o_i,j}) = P_s(x_{o_i,j})$;
   (c) for each X_{o_i} generate the normalization constraints $\sum_j P(x_{o_i,j}) = 1$;
   (d) for each event $x_{o_i,j}$, generate the constraints for $P(x_{o_i,j})$ using the Algorithm 2b;
3. solve the resulting multilinear problem by minimizing $w_{q,t}$.

To illustrate every step of this procedure we consider the computation of $\underline{P}(x_{1,0})$ in the very simple network of the Figure 2. The variable X_1 means the proposition *the clothes inside box are colored.* Its values, $x_{1,0}$ and $x_{1,1}$, indicate whether the proposition is true or false. The variable X_2 represents the proposition *the clothes will be sold.* The category $x_{2,0}$ denotes *true* and $x_{2,1}$ denotes *false.* The network has the following credal sets[5]:

$$K(X_1) = CH(\{(0.7, 0.3); (0.6, 0.4)\})$$
$$K(X_2|x_{1,0}) = CH(\{(0.9, 0.1); (0.5, 0.5)\})$$
$$K(X_2|x_{1,1}) = CH(\{(0.3, 0.7); (0.6, 0.4)\})$$

The soft evidence is provided by a marketing expert who has inspected the clothes and asserted $p_s(X_2) = (0.8, 0.2)$.

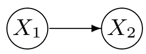

Fig. 2. Credal network for the clothes' example.

[5] $CH(\cdot)$ denotes the convex hull operator.

The objective is to compute $\underline{P}(x_{1,0}) = \min P(x_{1,0})$ - in fact we are going to minimize $P_u(x_{1,0})$ which is the updated version of $P(x_{1,0})$ after the soft evidence. The first two steps of the Algorithm 3 generate the constraints for credal sets and the following expression (from the variable elimination procedure):

$$P(x_{1,0}) = \sum_{j=0}^{1} P(x_{2,j}|x_{1,0}) \cdot P(x_{1,0}) \,. \tag{4}$$

Then step 2(a) updates the Expression 4 to:

$$P_u(x_{1,0}) = \sum_{j=0}^{1} P(x_{2,j}|x_{1,0}) \cdot P(x_{1,0}) \cdot \varphi_{x_{2,j}} \,.$$

Steps 2(b) and 2(c) append $\varphi_{x_{2,0}} \cdot P(x_{2,0}) = P_s(x_{2,0})$, $\varphi_{x_{2,1}} \cdot P(x_{2,1}) = P_s(x_{2,1})$ and $P(x_{2,0}) + P(x_{2,1}) = 1$ to the constraints. The step 2(d) adds the constraints for $P(x_{2,0})$ and $P(x_{2,0})$ and compares their results with the soft evidence. After all, the set of constraints contains all the expressions below.

$$\begin{aligned}
P_u(x_{1,0}) &= \sum_{j=0}^{1} P(x_{2,j}|x_{1,0}) \cdot P(x_{1,0}) \cdot \varphi_{x_{2,j}} ; \qquad P_s(x_{2,0}) = 0.8; \\
& P_s(x_{2,0}) = \varphi_{x_{2,0}} \cdot P(x_{2,0}) \,; P_s(x_{2,1}) = \varphi_{x_{2,1}} \cdot P(x_{2,1}) \,; \\
P(x_{2,0}) &= \sum_{j=0}^{1} P(x_{2,0}|x_{1,j}) \cdot P(x_{1,j}) \,; \qquad P_s(x_{2,1}) = 0.2; \\
P(x_{2,1}) &= \sum_{j=0}^{1} P(x_{2,1}|x_{1,j}) \cdot P(x_{1,j}) \,; \quad P(x_{2,0}) + P(x_{2,1}) = 1;
\end{aligned}$$

Now, the minimization function is given by $\underline{P}(x_{1,0}) = \min P_u(x_{1,0})$ and the answer is found by applying the multlinear solver, achieving $\underline{P}(x_{1,0}) = 0.666$, which differs from the original lower probability.

5 Conclusion

In this work we propose a method to integrate soft evidence in credal networks. The proposed approach uses the fact that soft evidences can be expressed by virtual evidences to generate multilinear constrains on the strong extension of the credal network. Then, a multilinear programming solver is used to compute the desired probability intervals.

The main contribution of the paper is to explore the use of soft evidence in credal networks, a matter of clear importance that, to the best of our knowledge, has not been fully discussed in the literature. The algorithm we have presented for dealing with soft evidence makes use of well known techniques for inference in credal networks, with their benefits and disadvantages. The main drawback of this approach comes from the fact that its computational complexity is very high (because credal network inferences have high complexity), so approximate ideas shall be investigated. Approximate methods for soft evidence in credal networks are an important topic that we intend to pursue in future work.

References

1. H. Chan and A. Darwiche. On the revision of probabilistic beliefs using uncertain evidence. *Artificial Intelligence*, 163(1):67–90, 2005.
2. I. Couso, S. Moral, and P. Walley. A survey of concepts of independence for imprecise probabilities. *Risk, Decision and Policy*, (5):165–185, 2000.
3. F. Cozman. Generalizing variable elimination. In *Workshop on Probabilistic Reasoning in Artificial Intelligence*, pages 1–12, Atibaia, Brazil, 2000.
4. F. G. Cozman. Robustness analysis of Bayesian networks with local convex sets of distributions. In D. Geiger, P. Shenoy, and H. Horvitz, editors, *13th Annual Conference on Uncertainty in Artificial Intelligence Conference*, pages 393–405, San Francisco, July 13-15 1997.
5. F. G. Cozman. Credal networks. *Artificial Intelligence*, 120(2):199–233, 2000.
6. C. P. de Campos and F. G. Cozman. Inference in credal networs using multilinear programming. In *Second Starting AI Researcher Symposium (STAIRS)*, pages 50–61, Valencia, Spain, 2004.
7. P. Diaconis and S. L. Zabell. Updating subjective probability. *J. American Stat. Assoc*, 77(380):822–830, 1982.
8. R. F. Drenick. Multilinear programming: Duality theories. *Journal of Optimization Theory and Applications*, 72(3):459–486, 1992.
9. H. Edelsbrunner. *Algorithms in Computational Geometry*. Springer-Verlag, Berlin, 1987.
10. Y.-G. Kim, M. Valtorta, and J. Vomlel. A prototypical system for soft evidential update. *Applied Intelligence*, 2004.
11. I. Levi. *The Enterprise of Knowledge*. MIT Press, Cambridge, 1980.
12. S. Moral. Algorithms for imprecise probabilities. Technical report, Universidade de Granada, DCCIA, Universidade de Granada, 1999.
13. K. Parsaye and M. Chignell. *Expert systems for experts*. John Wiley and Sons, New York, 1988.
14. J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Francisco, 1988.
15. J. C. F. Rocha. *Algoritmos para Inferência em Redes Credais*. Doutorado, Universidade de São Paulo, São Paulo, Brasil, 2004. Em português.
16. S. Russell and P. Norvig. *Artificial Intelligence: A modern approach*. Prentice Hall, Upper Saddle River, 1995.
17. H. Sherali and C. Tuncbilek. A global optimization algorithm for polinomial programming problems using a reformulation-linearization technique. *Journal of Global Optimization*, 2(1):101–112, 1992.
18. N. L. Zhang and D. Poole. A simple approach to bayesian network computations. In *Workshop on Real-Time Decision Support and Diagnosis System*, pages 171–178, Banff, Canada, 2002.