Credal Sentential Decision Diagrams

Alessandro Antonucci ALESSANDRO@IDSIA.CH
Alessandro Facchini ALESSANDRO.FACCHINI@IDSIA.CH
Lilith Mattei LILITH@IDSIA.CH
Istituto Dalle Molle di Studi sull'Intelligenza Artificiale (IDSIA), Lugano (Switzerland)

Abstract

Probabilistic sentential decision diagrams are logical circuits annotated by probability mass functions on the disjunctive gates. This allows for a compact representation of joint mass functions consistent with logical constraints. We propose a *credal* generalisation of the probabilistic quantification of these models, that allows to replace the local probabilities with (credal) sets of mass functions specified by linear constraints. This induces a joint credal set, that sharply assigns probability zero to states inconsistent with the constraints. These models can support cautious estimates of the local parameters when only small amounts of training data are available. Algorithmic strategies to compute lower and upper bounds of marginal and conditional queries are provided. The task can be achieved in linear time with respect to the diagram size for marginal queries. The same can be done for conditional queries if the topology of the circuit is singly connected.

Keywords: probabilistic graphical models, credal sets, logical constraints, arithmetic circuits, sentential decision diagrams, sum-product networks

1. Introduction

Probabilistic graphical models [8] are widely used in artificial intelligence for machine learning and knowledge-based decision-support systems. Such models provide a compact description of joint probability mass functions based on a factorization induced by conditional independence relations among the model variables (and depicted as a graph). This does not necessarily implies that inferences in the model can be computed efficiently [9].

To prevent computational issues, some authors proposed models based on *logical compilation* allowing for fast inferences at the price of a less transparent structure and reduced expressive power. *Sum-product networks* (SPNs) [15] are the most popular example in this area. Remarkably, SPNs can be also intended as a probabilistic counterpart of deep neural networks and, when applied to machine learning, they offer competitive performances [14].

Shortly after the first proposals for graphical models, some authors considered the possibility of relaxing the specification of the probabilistic parameters. In the area of *imprecise probabilities* [16], this is often done by means of *credal sets*, i.e., sets of probability mass functions induced by a (typically finite) number of linear constraints. If the first example of this approach is the credal set extension of the notion of Bayesian network, called *credal network* [3], one of the most recent ones is the credal set extension of SPNs proposed in [12, 13]. These papers show that a credal extension of SPNs can be achieved without compromising the computational eases provided by the circuital approach.

In this paper we present a very similar approach intended to achieve such a credal set extension in the case of *probabilistic sentential decision diagrams* (PSDDs) [7]. The formal analogies between PSDDs and SPNs allow for an adaptation of the algorithms originally proposed in [12] for credal SPNs to the case of *credal PSDDs* (to be called CSDDs in the rest of the paper). The contribution is significant as PSDDs can natively embed logical constraints, a feature not offered by SPNs.

Overall, the goal of the paper is to propose CSDDs as a new class of imprecise probabilistic graphical models, which allows for a compact specification of joint credal sets in the presence of logical constraints whilst keeping *tractable* the inferences. On another perspective, in the same spirit of recent articles such as [4], this work can be regarded as another contribution in the aim of unifying logic and imprecise probabilities.

The paper is organized as follows. In the next section we open the discussion by means of a toy example from [7] to be used along the paper to illustrate our approach. Section 3 contains background material about credal sets and PSDDs. The technical results are in Section 4 where we define CSDDs and in Section 5 where we derive two inference algorithms. Conclusions and outlooks are in Section 6.

2. A Motivating Example

In this section we introduce a simple example advocating the need of coherently combining logic and probability by a circuital approach. In this toy example, originally presented in [7], we start from a registration of the results of 100 students in four subjects, namely *logic* (L), *knowledge representation* (K), *probability* (P) and *AI* (A). From those data (Table 1), we want to learn a probabilistic model such as a Bayesian network (BN) [8]. The white nodes in

Figure 1 show a BN model that assumes conditional independence between K and P given A and L. A Laplace prior of strength one is used to learn the BN parameters. This assigns probability $\Pr(X=x) = [n(X=x)+|X|^{-1}]/[n(\cdot)+1]$ to the state x of variable X, where $|X|$ is the cardinality of X and n is a counting function for the data set (e.g., $\Pr(L=0) = 70.5/101$). Out of sixteen possible combinations, Table 1 reports no observations for eight cases (grey rows). Yet, seven of these eight cases (dark grey) are not observed because of logical constraints: $L \vee P$ (it is compulsory for a student to either take logic or probability), $A \to P$ (probability is prerequisite for AI), and $K \to A \vee L$ (knowledge representation is a prerequisite to either AI or logic). The light grey row corresponds instead to a possible configuration for which we just do not have observations. The BN cannot distinguish between those two cases and non-zero probabilities can be therefore assigned to impossible events (e.g., $\Pr(A=1, P=0) \simeq 0.005$).

Auxiliary nodes (grey nodes in Figure 1) should augment the BN in order to embed the logical constraints. Each constraint induces a Boolean variable which is a child of all the variables involved in the logical constraint and whose conditional probability table has zero/one values modelling the fact that the variable is true if and only if the logical constraint is satisfied. The model is therefore queried given the fact that the auxiliary variables implementing the constraints are in their true state. Such a BN approach to the learning of probability mass functions that are subject to domain constraints might significantly increase the *treewidth* of the original graphical model, thus potentially affecting the speed of the inferences (exact inference is exponential in this parameter).

L	K	P	A	#
0	0	0	0	0
0	0	0	1	0
0	0	1	0	6
0	0	1	1	54
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	10
1	0	0	0	5
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	13
1	1	0	1	0
1	1	1	0	8
1	1	1	1	3

Table 1: Results of 100 students in four subjects

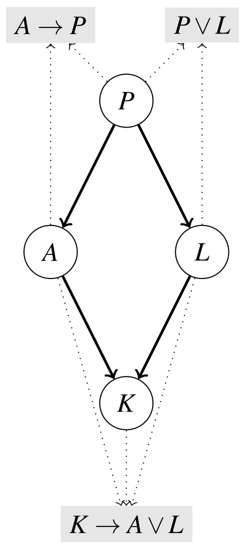

Figure 1: A Bayesian net over four variables (white nodes) with logical constraints (grey nodes)

3. Background

3.1. Credal Sets

Consider a variable X taking its values in a finite set $\mathscr{X}$ and whose generic value is x. A *probability mass function* (PMF) over X, denoted as $\mathbb{P}(X)$, is a real-valued non-negative function $\mathbb{P} : \mathscr{X} \to \mathbb{R}$ such that $\sum_{x\in\mathscr{X}} \mathbb{P}(x) = 1$. Given a function f of X, the *expectation* of f with respect to a PMF $\mathbb{P}$ is $\mathbb{P}[f] := \sum_{x\in\mathscr{X}} f(x) \cdot \mathbb{P}(x)$. A set of PMFs over X is called *credal set* (CS) and denoted as $\mathbb{K}(X)$. Here we consider CSs induced by a finite number of linear constraints. Given CS $\mathbb{K}(X)$, the bounds of the expectation with respect to $\mathbb{K}(X)$ can be computed. E.g., for the lower bound, $\underline{\mathbb{P}}[f] := \min_{\mathbb{P}(X)\in\mathbb{K}(X)} \sum_{x\in\mathscr{X}} f(x) \cdot \mathbb{P}(x)$. This is a linear programming task, whose optimum remains the same after replacing $\mathbb{K}(X)$ with its convex hull. Such optimum is attained on an extreme point of the convex closure. Moreover, if f is an indicator function, the lower expectation is called *lower* probability. Notation $\overline{\mathbb{P}}$ is used instead for the upper bounds and duality $\overline{\mathbb{P}}(f) = -\underline{\mathbb{P}}(-f)$ holds.

In the special case of Boolean variables, it is easy to see the number of extreme points of the convex closure of a CS cannot be more than two, and the specification of a single interval constraint, say $0 \leq l \leq \mathbb{P}(x) \leq u \leq 1$ for one of the two states is a fully general CS specification.

Learning CSs from multinomial data can be done by the *imprecise Dirichlet model* (IDM) [16]. This is a generalised Bayesian approach in which a single Dirichlet prior with equivalent sample size s is replaced by the set of all the Dirichlet priors with this size. The corresponding bounds for $\mathbb{P}(x)$ are $\left[\frac{n(x)}{n(\cdot)+s}, \frac{n(x)+s}{n(\cdot)+s}\right]$ where the notation is analogous to that in the previous section (e.g., with the counts in Table 1 and $s=1$, $P(K=\top,L=\bot,A=\top,P=\top)\in\frac{10}{101},\frac{11}{101}]$).

Given PMF $\mathbb{P}(X_1,X_2)$, X_1 and X_2 are *stochastically independent* if and only if $P(x_1,x_2)=P(x_1)\cdot P(x_2)$ for each $x_1\in\mathscr{X}_1$ and $x_2\in\mathscr{X}_2$. Similarly, given CS $K(X_1,X_2)$, we say that X_1 and X_2 are *strongly independent* if and only if stochastic independence is satisfied for each extreme point of the convex closure of the CS.

3.2. Probabilistic Sentential Decision Diagrams

PSDDs offer an elegant approach to the learning of probabilistic models from data subject to logical constraints [7]. A PSDD is a circuit representation of a joint PMF assigning zero probability to the impossible states of the logical constraints. This requires a parametrised directed acyclic graph such that each non-root node is either a logical AND gate with two inputs, or a logical OR gate with an arbitrary number of inputs and whose incoming wires are annotated with the probabilities of a conditional mass function (that can be learned from the data). These types of nodes alternate. Each leaf node X is intended as a univariate mass function assigning probability one to X, when X is always true, to $\neg X$ when X is always false, or assigning θ to X (written as $(X:\theta)$) when X is true with probability θ (conditional to a certain *context* encoded by the circuit). As an example the circuit in Figure 3 with sharp numbers instead of intervals associated to the wires defines a PSDD consistent with the logical constraints of the toy example in Section 2.

PSDD inference, intended here as the computation of the probability of marginal or conditional queries, takes time linear in the size of the diagram [7]. Dedicated algorithms have been developed to identify the smallest circuit modelling a given formula [2]. This makes inferences based on a PSDD typically faster than those based on a BN with auxiliary nodes implementing the same constraints as in Figure 1. Yet, PSDDs embed context-specific independence relations and the problem of trading off the search for a small circuit with the learning of proper dependence relations from the data is still relatively unexplored [10]. This is not the case of BNs, for which many structural learning techniques have been proposed [8].

In the rest of this section we review the basic notions and results about PSDDs. We start by defining a generalisation of orders on variables based on the following definition.

Definition 1 (Vtree) *Consider a finite set $\mathbf{X}$ of Boolean variables. A* vtree *for $\mathbf{X}$ is a full binary tree v whose leaves are in one-to-one correspondence with elements of $\mathbf{X}$. We denote by v^l (v^r) the left (right) subtree of v, i.e., the vtree rooted at the left (right) child of the root of v.*

A vtree for the example in Section 2 is in Figure 2. Note that the in-order tree traversal of a vtree induces a total order on the variables, but two distinct vtrees can induce in this way the same order.

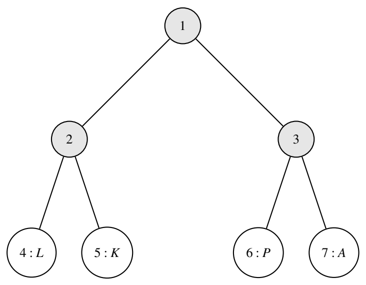

Figure 2: A vtree over four variables

Based on vtree, we can recursively define SDDs.

Definition 2 (SDD) *A sentential decision diagram (SDD) α normalised for vtree v is either a terminal or a decision. The interpretation of an SDD α normalised for v, denoted $\langle\alpha\rangle$, is a propositional sentence over the variables of v.*

- ***Terminal:*** *if v is a leaf with variable X, α might be a constant, $\alpha\in\{\bot,\top\}$, or a literal, $\alpha\in\{X,\neg X\}$. The interpretations are as follows $\langle\alpha\rangle=\bot$, $\langle\alpha\rangle=\top$ and $\langle\alpha\rangle=X$, $\langle\alpha\rangle=\neg X$.*
- ***Decision:*** *if v not a leaf, $\alpha=\{(p_i,s_i)\}_{i=1}^k$, where the p_i's and s_i's, called* primes *and* subs*, are SDDs normalised for v^l and v^r respectively. The interpretations $\langle p_i\rangle$'s form a partition and $\langle\alpha\rangle=\bigvee_{i=1}^k\langle p_i\rangle\wedge\langle s_i\rangle$. The pairs (p_i,s_i)'s are called the* elements *of the decision node and k is its* size*.*

Given a SDD, and two of its nodes n and n', if n is a decision node and n' appears as a prime or sub in n, we call n a *parent* of n'. Whenever every node of a SDD has a unique parent, we call the SDD *singly connected.*

As each decision represents an OR gate, and each of its elements an AND gate, we can intend SDD α as a logical circuit providing a representation of the formula $\langle\alpha\rangle$.

Example 1 *Let $\phi=A\to P$. Given the vtree $v=(A,P)$, the interpretation of the SDD α normalised for v given by $\langle\alpha\rangle=(A\wedge P)\vee(\neg A\wedge\top)$ is logically equivalent to ϕ. Hence $\alpha=\{(A,P),(\neg A,\top)\}$ is the SDD (normalised for v) corresponding to ϕ.*

The following definition characterises paths in SDDs. This is needed to provide a semantics to the parameters when annotating SDDs with probabilities.

Definition 3 (Context) *Let n be a node (either terminal or decision) of an SDD. Denote as $(p_1,s_1)\dots(p_l,s_l)$ the path from the root to node n. Then the conjunction of the interpretations of the primes encountered in this path, i.e., $\langle p_1\rangle\wedge\cdots\wedge\langle p_l\rangle$, is called the* context *of n and denoted as γ_n. The context γ_n is* feasible *if and only if $s_i\neq\bot$ for each $i=1,\dots,l$.*

Example 2 *Consider the underlying SDD α in Figure 3, and let n be the terminal node associated to the first occurrence of A (from the left). Then $\gamma_n = ((\neg L \wedge K) \wedge P)$. Indeed, the path from the root to node n is given by $(p_1, s_1)(p_2, s_2)$ with $\langle p_1 \rangle = ((\neg L \wedge K) \vee (L \wedge \bot)) = (\neg L \wedge K)$ and $\langle p_2 \rangle = P$.*

The interpretation of a node is implied by its context and by the interpretation of the SDD to which it belongs, i.e., for each node n of an SDD r, $\langle r \rangle \wedge \gamma_n \models \langle n \rangle$. Moreover, nodes normalized for the same vtree node have mutually exclusive and exhaustive contexts. In other words the OR gates associated to decision nodes act as XORs.

We are now in the condition of providing a formal definition of PSDDs as parametrised SDDs inducing a PMF over the variables of the vtree. For each terminal node $\top$, a positive parameter θ is provided such that $0 \leq \theta \leq 1$. The notation for such a terminal node is $X : \theta$, where X is the variable of the leaf vtree node that $\top$ is normalised for. Terminal nodes other than $\top$ appear as they are; for each decision node $\{(p_i, s_i)\}_{i=1}^k$, each prime p_i is provided with a $\theta_i > 0$, such that $\sum_{i=1}^k \theta_i = 1$, while $\theta_i = 0$ if and only if $s_i = \bot$. Notation $\{(p_i, s_i, \theta_i)\}_{i=1}^k$ is used. In other words, PSDDs are SDDs with PMFs associated to the terminal and decision nodes.

In a PSDD α, each node n induces a PMF $\mathbb{P}_n$ over the variables of the vtree node it is normalised for. According to the *Base Theorem* for PSDDs [7, Theorem 1], such PMF $\mathbb{P}_n$ assigns zero probability to events which do not respect the propositional sentence associated to the SDD n, more precisely, for any instantiation $\boldsymbol{x}$ of variables $\boldsymbol{X}$ of the vtree n is normalised for, $\mathbb{P}_n(\boldsymbol{x}) > 0$ iff $\boldsymbol{x} \models \langle n \rangle$. Moreover, the probabilities $\mathbb{P}_n(\langle p_i \rangle)$ are the parameters θ_i's of $n = \{(p_i, s_i, \theta_i)\}_{i=1}^k$.

We simply denote as $\mathbb{P}$ the (joint) PMF induced by the root r. PMF $\mathbb{P}_n$ induced by an internal node can be obtained by conditioning $\mathbb{P}$ on a feasible context of the considered node [7, Theorem 4]: for each feasible context γ_n of n, $\mathbb{P}_n(\cdot) = \mathbb{P}(\cdot|\gamma_n)$. Finally, we have the following result about independence [7, Theorem 5]: according to $\mathbb{P}$, the variables inside v are independent of those outside v given context γ_n. This is the PSDD analogous of the *Markov condition* for Bayesian networks.

PSDD inferences are computed with respect to the joint PMF $\mathbb{P}$. The probability of a joint state $\mathbf{e}$ of a set of PSDD variables can be obtained in linear time with respect to the diagram size by the bottom-up scheme in Algorithm 1 Given a vtree node v, notation $\mathbf{e}_v$ is used for the subset of $\mathbf{e}$ including only the variables of v. Note also that the node index n in the loop runs in inverse topological order and this guarantees that the *message* $\pi(n)$, to be computed after the *else* statement, is always a combination of messages already computed.

Algorithm 1 Probability of evidence [7]

```
input: PSDD, evidence $\mathbf{e}$
for $n \leftarrow N, \dots, 1$ do
  $\pi(n) \leftarrow 0$
  if node $n$ is terminal then
    $v \leftarrow$ leaf vtree node that $n$ is normalized for
    $\pi(n) \leftarrow \mathbb{P}_n(\mathbf{e}_v)$
  else
    $(p_i, s_i, \theta_i)_{i=1}^k \leftarrow n$ (decision node)
    $\pi(n) \leftarrow \sum_{i=1}^l \pi(p_i) \cdot \pi(s_i) \cdot \theta_i$
  end if
end for
output: $\mathbb{P}(\mathbf{e}) \leftarrow \pi(1)$
```

4. Credal Sentential Decision Diagrams

In this section we present a generalization of PSDDs (see Section 3.2) based on the notion of credal set provided in Section 3.1. This can be achieved as follows.

Definition 4 *A* credal sentential decision diagram *(CSDD) is an SDD augmented as follows.*

- *For each terminal node $\top$, an interval $[l, u]$ is provided such that $0 < l \leq u < 1$. The notation for such a terminal node is $X : [l, u]$, where X is the variable of the leaf vtree node that $\top$ is normalised for. Terminal nodes other than $\top$ appear as they are.*
- *For each decision node $n = \{(p_i, s_i)\}_{i=1}^k$ a CS $\mathbb{K}_n(P)$ over the variable P is provided. The states of the variable are the interpretations (formulas) $\langle p_i \rangle$ associated to the primes p_i occurring in n. We require $\mathbb{P}(\langle p_i \rangle) = 0$ for each $\mathbb{P}(P) \in \mathbb{K}_n(P)$ if and only if $\langle s_i \rangle = \bot$.*

A CSDD is *singly connected* is its underlying SDD is singly connected, i.e., the undirected graph underlying the circuit is a tree.

Besides the CSs associated to the primes of each decision node, the intervals $[l, u]$ assigned to terminal nodes $\top$ are also CS specifications (see Section 3.1). A CSDD can be therefore regarded as a PSDD whose local PMFs have been replaced by CSs. This also gives a semantics for these CSs, which are regarded as conditional CSs for the variables/events in the associated nodes given the context.

Exactly as a PSDD defines a joint PMF, a CSDD defines a joint CS. Such a CS, called here the *strong extension* of the CSDD and denoted as $\mathbb{K}^r(\boldsymbol{X})$, where r is the root node of the CSDD, is defined as the convex hull of the set of joint PMFs obtained by *all* the PMF specifications on the terminal and decision nodes consistent with the corresponding CSs.

By definition of CSDD strong extension and by the Base Theorem for PSDDs, we have the following result.

Theorem 5 (Base) *For each node n of a CSDD, for each instantiation z of its variables $\mathbf{Z}$,*

$$\underline{\mathbb{P}}_n(z) > 0 \quad \textit{iff} \quad z \models \langle n \rangle\,, \tag{1}$$

$$\overline{\mathbb{P}}_n(z) = 0 \quad \textit{iff} \quad z \not\models \langle n \rangle\,, \tag{2}$$

where $\underline{\mathbb{P}}_n(z) = \min_{\mathbb{P}(\mathbf{Z})\in\mathbb{K}^n(\mathbf{Z})} \mathbb{P}(z)$ *and* $\overline{\mathbb{P}}_n(z) = \max_{\mathbb{P}(\mathbf{Z})\in\mathbb{K}^n(\mathbf{Z})} \mathbb{P}(z)$.

CSDD inferences are intended as the computation of lower and upper bounds with respect to the strong extension.

An important remark is that, as the extreme points of the convex hull of a set also belong to the original set, the extreme points of the strong extension are joint PMFs induced by PSDDs (whose local PMFs are consistent with the local CSs in the CSDD). As a consequence of that, a CSDD encodes the same probabilistic independence relations of a PSDD based on the same SDD, but based on the notion of strong independence instead of that of stochastic independence (see Section 3.1). Thus, the variables of a node are *strongly* independent from the ones outside the node when its context is given and feasible. In this sense, the relation between PSDDs and CSDDs retraces that between BNs and credal networks [3].

Figure 3 depicts a CSDD modeling the data and the constraints of the toy example in Section 2. The circuit is that of a SDD modeling the logical constraints among the four Boolean variables, while the probability intervals associated to the wires are obtained by the *local* IDM (Section 3.1). When the application of the IDM involves zero counts because of the logical constraints, the IDM interval associated to the impossible event is replaced by a sharp zero.

Note also that the CSs of a CSDD are associated to conditional probabilities based on the context (Definition 3). The number of variables involved in the context increases with the distance from the root node. For contexts involving many variables we might therefore have only a small amount of data consistent with the context and, consequently, very large probability intervals (see Section 3.1). On one side, this justifies the need of a robust statistical learning of the parameters as the one provided by the IDM when only few learning data are available. On the other side, this issue confirms the need for structural learning algorithm able to decide the optimal structure for a PSDD or CSDD [10, 11].

5. Inference in CSDDs

5.1. Marginal Queries

Algorithm 1 computes the probability of a marginal query in a PSDD. Algorithm 2 provides an extension of this procedure to CSDDs, allowing for the computation of lower/upper marginal probabilities. The procedure follows exactly the same scheme based on a reverse topological order with respect to the terminal and the decision nodes. Unlike Algorithm 1, every time a decision node is processed, Algorithm 2 also requires the solution of a linear programming task whose feasible region is the local CS.

Algorithm 2 Lower probability of evidence

```
input: CSDD, evidence e
for $n \leftarrow N,\dots,1$ do
  $\underline{\pi}(n) \leftarrow 0$
  if node $n$ is terminal then
    $v \leftarrow$ leaf vtree node that $n$ is normalized for
    $\underline{\pi}(n) \leftarrow \underline{\mathbb{P}}_n(\mathbf{e}_v)$
  else
    $((p_i,s_i)_{i=1}^k, \mathbb{K}_n(P)) \leftarrow n$ (decision node)
    $\underline{\pi}(n) \leftarrow \min_{[\theta_1,\dots,\theta_n]\in\mathbb{K}_n(P)} \sum_{i=1}^k \underline{\pi}(p_i)\cdot\underline{\pi}(s_i)\cdot\theta_i$
  end if
end for
output: $\underline{\mathbb{P}}(\mathbf{e}) \leftarrow \underline{\pi}(1)$
```

To see why the algorithm properly computes $\underline{\mathbb{P}}(\mathbf{e})$ just regard the output of Algorithm 1 as a symbolic expression of the local probabilities involved in the CSDD local CSs. This is a multi-linear function of these probabilities subject to the linear constraints defining the CSs. The optimizations with respect to the CSs of the terminal nodes can be done independently of the others. Afterwards, the decision nodes that are parents of terminal nodes can be safely processed and so on. This is clearly the reverse topological order indexed by the counter n. The algorithm runs in polynomial time as it requires the solution of a single linear programming task for each local CS of the CSDD. Note that for terminal nodes the optimization is trivial as it only consists in the computation of a lower probability for a CS over a Boolean variable. An analogous procedure can be also defined for upper probabilities.

The intuition above is made formal by next theorem, which states that the output of Algorithm 2 is indeed the lower bound of a query with respect to the strong extension of the CSDD.

Theorem 6 *For any node n normalized for vtree v and for an evidence $\boldsymbol{e}$ over some variables in a CSDD:*

$$\underline{\pi}(n) = \underline{\mathbb{P}}_n(\boldsymbol{e})\,. \tag{3}$$

Proof If n is a terminal node, the theorem is true by definition of Algorithm 2 (the computation of $\underline{\mathbb{P}}_n(\mathbf{e}_v)$ is immediate). Let $n = (\{(p_i,s_i)\}_{i=1}^k, \mathbb{K}_n(P))$ be an internal node and assume that the theorem holds for n's descendants. If l and r are the left, respectively right sub-vtree of v, we have that:

$$\underline{\mathbb{P}}_n(\mathbf{e}) = \min_{\mathbb{P}_n(\mathbf{Z})\in\mathbb{K}^n(\mathbf{Z})} \mathbb{P}_n(\mathbf{e})$$

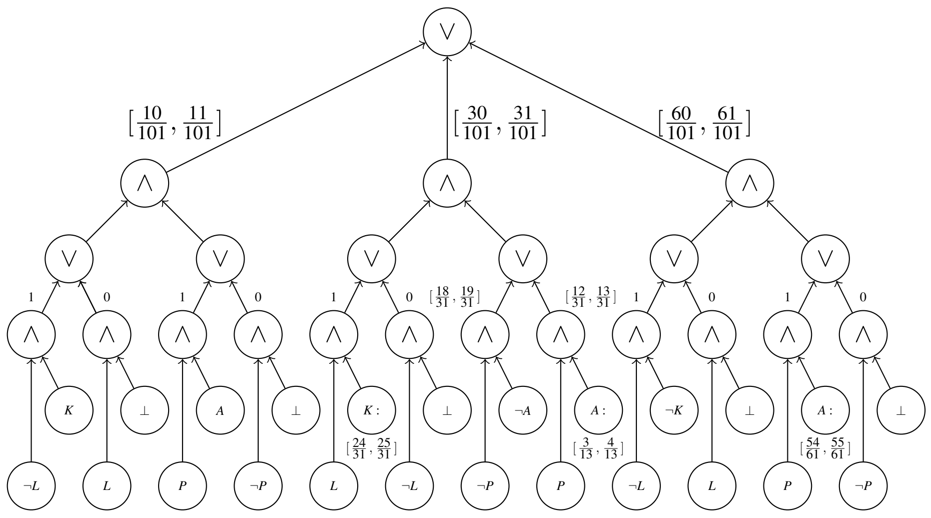

Figure 3: A CSDD for the toy example in Section 2

$$= \min_{\mathbb{P}_n(\mathbf{Z})\in\mathbb{K}^n(\mathbf{Z})} \sum_{i=1}^{k} \mathbb{P}_{p_i}(\mathbf{e}_l)\cdot\mathbb{P}_{s_i}(\mathbf{e}_r)\cdot\theta_i$$

$$= \min_{[\theta_1,\dots,\theta_k]\in\mathbb{K}_n(P)} \sum_{i=1}^{k} \underline{\pi}(p_i)\cdot\underline{\pi}(s_i)\cdot\theta_i\,. \quad (4)$$

The second equality comes from [7, Theorem 7]. The last equality is due to the fact that, at this point, in order to minimize this quantity, one can first minimize separately $\mathbb{P}_{p_i}(\mathbf{e}_l)$ and $\mathbb{P}_{s_i}(\mathbf{e}_r)$ because these minimizations are done over two distinct CSs (the strong extension of the sub-CSDD rooted at p_i and the strong extension of the sub-CSDD rooted at s_i), and then, with the obtained values, solve the LP over the CS $\mathbb{K}_n(P)$ attached to node n. Hence, the induction hypothesis applies. ■

As an example of the application of Algorithm 2 consider the query $(A=\bot, P=\bot)$ for the CSDD in Figure 3. Out of three parent edges of the root decision node, only the second one gives a non-zero contribution. Thus, by simple multiplication of the lower bounds of the IDM intervals we obtain $\underline{\mathbb{P}}(A=\bot,P=\bot)=\frac{18}{31}\cdot\frac{30}{301}=\frac{540}{3131}$ and similarly $\overline{\mathbb{P}}(A=\bot,P=\bot)=\frac{19}{101}$.

5.2. Conditional Queries

In the previous section we showed how to compute the lower (or upper) marginal probabilities in a CSDD by Algorithm 2. This corresponds to a sequence of linear programming tasks whose feasible regions are the local CSs of the CSDD processed in reversed topological order, thus taking polynomial time with respect to the diagram size. In this section we show that something similar can be done for conditional queries under the additional assumption that the CSDD is singly connected.

Let $X=x$ denote the variable and state to be queried and $\boldsymbol{e}$ the available evidence about other variables in a CSDD α rooted at r with variables $\boldsymbol{X}$. The task is to compute the lower conditional probability with respect to the strong extension, i.e.,

$$\underline{\mathbb{P}}(x|\boldsymbol{e}) = \min_{\mathbb{P}(\boldsymbol{X})\in\mathbb{K}^r(\boldsymbol{X})} \frac{\mathbb{P}(x,\boldsymbol{e})}{\mathbb{P}(\boldsymbol{e})}\,. \quad (5)$$

Note that, in order for $\underline{\mathbb{P}}(x|\boldsymbol{e})$ to be defined, we need to assume that $\boldsymbol{e}$ is consistent with the underlying SDD's interpretation $\langle\alpha\rangle$. To see this, assume there is a total instantiation of $\boldsymbol{X}$ extending $\boldsymbol{e}$. Then, given an extreme point $\mathbb{P}(\boldsymbol{X})$ of the strong extension $\mathbb{K}^r(\boldsymbol{X})$, the Base Theorem for PSDDs tells us that $\mathbb{P}(\boldsymbol{x})>0$ if and only if $\boldsymbol{x}\models\langle\alpha\rangle$. This immediately yield that the denominator in the right hand side of Equation (5) is positive for each extreme point of the strong extension $\mathbb{K}^r(\boldsymbol{X})$ if and only if $\boldsymbol{e}$ is consistent with $\langle\alpha\rangle$.

The task in Equation (5) corresponds to the linearly constrained minimization of a (multilinear) fractional function of the probabilities. This prevents a straightforward application of the same approach considered in the previous section. Thus, we consider instead a decision version of the optimization task in Equation (5), i.e., deciding whether or not the following inequality is satisfied for a given $\mu\in[0,1]$:

$$\underline{\mathbb{P}}(x|\boldsymbol{e}) > \mu\,. \quad (6)$$

As for the algorithm in [5], an algorithm able to solve Equation (6) for any $\mu\in[0,1]$ inside a bracketing scheme linearly converges to the actual value of the lower probability.

As $\mathbb{P}(x|\boldsymbol{e})+\mathbb{P}(\neg x|\boldsymbol{e})=1$ for each $\mathbb{P}(\boldsymbol{X})\in\mathbb{K}^r(\boldsymbol{X})$, and assuming that $\mathbb{P}(\boldsymbol{e})>0$, Equation (6) holds if and only if the following inequality holds:

$$\min_{\mathbb{P}(\boldsymbol{X})\in\mathbb{K}^r(\boldsymbol{X})} [(1-\mu)\mathbb{P}(x,\boldsymbol{e})-\mu\mathbb{P}(\neg x,\boldsymbol{e})] > 0\,. \quad (7)$$

For evidence (variable instantiation) $\boldsymbol{e}$ and vtree node v, we will use the notation $\boldsymbol{e}_v$ to denote the subset of instantiation of $\boldsymbol{e}$ that pertains to the variables of vtree v, and $\boldsymbol{e}_{\overline{v}}$ to denote the subset of $\boldsymbol{e}$ that pertains to variables outside v.

If $\boldsymbol{e} \models \neg x$, then $\mathbb{P}(x,\boldsymbol{e}) = 0$, and similarly if $\boldsymbol{e} \models x$, then $\mathbb{P}(\neg x,\boldsymbol{e}) = 0$. Otherwise both $x,\boldsymbol{e} = x,\boldsymbol{e}_{\overline{v}}$ and $\neg x,\boldsymbol{e} = \neg x,\boldsymbol{e}_{\overline{v}}$, and therefore $\mathbb{P}(x,\boldsymbol{e}) = \mathbb{P}(x,\boldsymbol{e}_{\overline{v}})$ and $\mathbb{P}(\neg x,\boldsymbol{e}) = \mathbb{P}(\neg x,\boldsymbol{e}_{\overline{v}})$, where v is the leaf node with variable X in the vtree the CSDD is normalized for. In the following we might therefore assume $\boldsymbol{e}_{\overline{v}} = \boldsymbol{e}$.

Let $n = (p_1,s_1,\theta_1),\dots,(p_k,s_k,\theta_k)$ be a decision node of a PSDD that is normalized for vtree node v with left child l and right child r. By [7, Theorem 6], for evidence $\boldsymbol{e}'$ it holds that $\mathbb{P}_n(\boldsymbol{e}') = \sum_{i=1}^k \mathbb{P}_{p_i}(\boldsymbol{e}'_l)\mathbb{P}_{s_i}(\boldsymbol{e}'_r)\theta_i$. Now suppose that the decision root of the CSDD $r = (\{(p_i,s_i)\}_{i=1}^k, \mathbb{K}_r(P))$ is normalized for vtree node v and that X occurs in the left sub-vtree, the case when X occurs in the right sub-vtree being *mutatis mutandis* the same. Hence, assuming the CSDD is singly connected, we have that Equation (7) rewrites as:

$$\min_{[\theta_1,\dots,\theta_k]\in\mathbb{K}_r(P)} \sum_{i=1}^k \underline{\pi}(p_i)\,\underline{\sigma}(s_i)\,\theta_i > 0\,. \tag{8}$$

where

$$\underline{\pi}(p_i) := \min_{\mathbb{P}_{p_i}(\boldsymbol{Z})\in\mathbb{K}^{p_i}(\boldsymbol{Z})} [(1-\mu)\mathbb{P}_{p_i}(x,\boldsymbol{e}_l) - \mu\mathbb{P}_{p_i}(\neg x,\boldsymbol{e}_l)] \tag{9}$$

and

$$\underline{\sigma}(s_i) = \begin{cases} \overline{\mathbb{P}}_{s_i}(\boldsymbol{e}_r) & \text{if } \underline{\pi}(p_i) < 0 \\ \underline{\mathbb{P}}_{s_i}(\boldsymbol{e}_r) & \text{otherwise.} \end{cases} \tag{10}$$

The evaluation of $\underline{\pi}(p_i)$ in Equation (9) is just another instance of the original minimization in Equation (7). If the CSDD in Equation (7) is a terminal, call it n, and assume X occurs in n. As optimal values are attained on the borders of the domain, the left hand side of the equation rewrites as:

$$\lambda_n(\mu) := \min\left\{ \begin{array}{l} (1-\mu)\underline{\mathbb{P}}_n(x) - \mu\overline{\mathbb{P}}_n(\neg x), \\ (1-\mu)\overline{\mathbb{P}}_n(x) - \mu\underline{\mathbb{P}}_n(\neg x) \end{array} \right\}, \tag{11}$$

where the lower and upper probabilities in the above expression are those associated to the bounds in the CS specification for $X = \top$ and the other values are obtained by the conjugacy relation $\underline{\mathbb{P}}(x) = 1 - \overline{\mathbb{P}}(\neg x)$.

The idea of the procedure to solve Equation (7) goes therefore as follows. Consider any terminal node n and the corresponding vtree node v. If X occurs in v, we associate to n the value $\underline{\pi}(n) := \lambda_n(\mu)$. Otherwise we may associate any value to it, say $\underline{\pi}(n) = 0$. The reason for the latter choice will become obvious in a moment. We then proceed bottom up. Let n be a decision node with corresponding vtree node v. Consider its elements $(p_1,s_1),\dots,(p_k,s_k)$, with associated values $\underline{\pi}(p_i)$, resp. $\underline{\pi}(s_i)$. If X does not occurs in v, then as before the value we are going to associate to n does not really matter, and thus let $\underline{\pi}(n) = 0$. Otherwise, X occurs in either v^l or v^r, say v^l. Hence, we use the left hand side of Equation (8) to determine the value associated with n, that is $\underline{\pi}(n) := \min_{[\theta_1,\dots,\theta_k]\in\mathbb{K}_n(P)} \sum_{i=1}^k \underline{\pi}(p_i)\,\underline{\sigma}(s_i)\,\theta_i$. Notice that, since X does not occur in v^r, the value $\underline{\pi}(s_i)$ previously associated to a subs s_i of n does not matter in the calculation of $\underline{\pi}(n)$. This justifies the arbitrary choice above when X does not occur in the vtree corresponding to a node.

To sum up, define, for a terminal node n:

$$\Lambda_n(\mu) := \begin{cases} \lambda_n(\mu) & \text{if X occurs in } n \\ 0 & \text{otherwise}\,. \end{cases} \tag{12}$$

The discussion above then justifies Algorithm 3 and its correctness in deciding whether or not inequality (6) is satisfied for a given $\mu \in [0,1]$ for singly connected CSDDs.

Algorithm 3 Lower conditional probability

```
input: CSDD, $\mu$, $X = x$, $\boldsymbol{e}$
for $n \leftarrow N,\dots,1$ do
    $\underline{\pi}(n) \leftarrow 0$
    $v \leftarrow$ vtree node that $n$ is normalized for
    if node $n$ is terminal then
        $\underline{\pi}(n) \leftarrow \Lambda_n(\mu)$ as in Eq. (12)
    else
        $((p_i,s_i)_{i=1}^k, \mathbb{K}_n(P)) \leftarrow n$ (decision node)
        if $X$ occurs in $v$ then
            if $X$ occurs in $v^l$ then
                $u \leftarrow v^l$ and $w \leftarrow v^r$
                $u_i \leftarrow p_i$ and $w_i \leftarrow s_i$ for $1 \le i \le k$
            else if $X$ occurs in $v^r$ then
                $w \leftarrow v^l$ and $u \leftarrow v^r$
                $u_i \leftarrow s_i$ and $w_i \leftarrow p_i$ for $1 \le i \le k$
            end if
            $\underline{\pi}(n) \leftarrow \min_{[\theta_1,\dots,\theta_k]\in\mathbb{K}_n(P)} \sum_{i=1}^k \underline{\pi}(u_i)\cdot\underline{\sigma}(w_i)\cdot\theta_i$
            with $\underline{\sigma}$ as in Eq. (10)
        end if
    end if
end for
output: sign$[\underline{\mathbb{P}}(x|\boldsymbol{e}) - \mu] \leftarrow$ sign$[\underline{\pi}(1)]$
```

The procedure described by Algorithm 3 requires the solution of a number of linear programming tasks, whose feasible regions are the local CSs associated with the CSDD, equal to the number of decision nodes. The computation of the coefficients of the objective function in these tasks requires a call of Algorithm 2 for each optimization variable to compute the quantities in Equation (10). Note also that the optimization in Equation (9) should be performed before the one in Equation (10). As discussed before, by iterated calls of Algorithm 3, we can therefore compute lower conditional queries in polynomial time in case of singly connected CSDDs.

To conclude the section, let us demonstrate how Algorithm 3 works in practice by considering the CSDD in

Figure 3. We consider the query $L = \top$ given evidence $A = \top$. Let us leave unspecified the value of $\mu \in (0,1)$. The last linear programming task required by the algorithm is the one associated to the root of the CSDD. The (IDM) linear constraints of this task are $\theta_1 \in [\frac{10}{101}, \frac{11}{101}]$, $\theta_2 \in [\frac{30}{101}, \frac{31}{101}]$ and $\theta_3 \in [\frac{60}{101}, \frac{61}{101}]$ and $\sum_{i=1}^{3} \theta_i = 1$. In the rest of this section we show that the objective function for this task has form:

$$f(\theta_1, \theta_2, \theta_3) := -\mu \cdot \theta_1 + (1-\mu) \cdot \frac{36}{403} \cdot \theta_2 - \mu \cdot \frac{55}{61} \cdot \theta_3 \,. \tag{13}$$

Consider first the terminal nodes normalized for vtree node 4 in Figure 2, i.e., those terminal nodes n concerning variable L and whose contribution is given by $\lambda_n(\mu)$. Between those, nodes labelled by L give contribution $(1-\mu)$ while those labelled by $\neg L$ give contribution $-\mu$. This is because $\underline{\mathbb{P}}_n(L = \top) = \overline{\mathbb{P}}_n(L = \top) = 1$ if $n = L$ and zero otherwise and, similarly, $\underline{\mathbb{P}}_n(L = \bot) = \overline{\mathbb{P}}_n(L = \bot) = 1$ if $n = \neg L$ and zero otherwise.

As explained in the previous section, at the beginning of the calculation $\underline{\pi}(n)$ is set to 0 for all terminal nodes not associated to vtree node 4, and their contribution will be eventually computed (as a $\underline{\sigma}$) when the procedure reach their unique parent decision node. We then proceed bottom up, starting from the decision nodes constituted by the primes and subs occurring as elements of the root. Notice that L occurs only in the primes as (left) variable, meaning that we need to consider those three decision nodes first. For what concerns the first prime $p_1 = \{(\neg L, K), (L, \bot)\}$, we have $\underline{\pi}(p_1) = \underline{\pi}(\neg L) \cdot \underline{\sigma}(K) \cdot 1 + \underline{\pi}(L) \cdot \underline{\sigma}(\bot) \cdot 0 = -\mu$. The parameters of p_1 are sharp, and therefore there is no minimization to perform. Moreover, as the evidence does not contain K, the computation of $\underline{\sigma}$ is trivial. Similarly, for p_3 it holds $\underline{\pi}(p_3) = -\mu$. For $p_2 = \{(L, \top), (\neg L, \bot)\}$ the situation is analogous, and the computation gives:

$$\underline{\pi}(p_2) = \underline{\pi}(L) \cdot \underline{\sigma}(\top) \cdot 1 + \underline{\pi}(\neg L) \cdot \underline{\sigma}(\bot) \cdot 0 = 1 - \mu \,.$$

As the sign of the contributions $\underline{\pi}(p_i)$ are known, we can thence compute $\underline{\sigma}(s_i)$, for $i = 1,2,3$. Since $\underline{\pi}(p_1)$ and $\underline{\pi}(p_3)$ are negative, both $\underline{\sigma}(s_1)$ and $\underline{\sigma}(s_3)$ are obtained by taking the upper probability of the evidence, i.e., $\underline{\sigma}(s_1) = 1$ and $\underline{\sigma}(s_3) = \frac{55}{61}$. On the other hand, the positivity of $\underline{\pi}(p_2)$ yields that in case of $\underline{\sigma}(s_2)$ we have to minimize $0 \cdot \theta_1 + \frac{3}{13} \cdot \theta_2$ with respect to the local CS attached to s_2. This is achieved for the lower bound of the right branch and gives $\frac{3}{13} \cdot \frac{12}{31} = \frac{36}{403}$. To conclude with the example, the minimization of the objective function in Eq. (13) with respect to the above described IDM constraints for different values of μ according to a bisection scheme allows to approximate the value for which the minimum is equal to zero as $\mu \simeq 0.0395$. The obtained value can be regarded as an estimate for the exact value of $\underline{\mathbb{P}}(L = \top | A = \top)$.

6. Conclusions

We proposed a new class of imprecise probabilistic graphical models based on a credal set extension of *probabilistic sentential diagrams*. Two efficient algorithms for marginal and conditional queries are provided. The results are perfectly analogous to those already derived in [12] for sum-product networks. Despite the strong analogies between SPNs and PSDDs, the latter models can also embed logical constraints, something that cannot be achieved by SPNs. This makes the present contribution relevant and, at the present moment, the most natural way to embed logical constraints in imprecise probabilistic graphical models. A lot of future work has to be done. First of all, we intend to explore the possibility of performing efficient credal classification with CSDDs in the presence of logical constraints (e.g., multi-label classification as in [1] or spatio-temporal data managing as in [6]). On the theoretical level the characterisation of the computational complexity of various inference tasks in these models (without imposing specific restrictions on the underlying structure) should be provided together with dedicated studies about the relations between credal networks and CSDDs, but also about the coherence relations characterising these new class of models.

Acknowledgements

The authors would like to express their gratitude to four anonymous reviewers, for their time, effort and constructive feedback. In particular the comments of a reviewer have been crucial to solve an issue with Algorithm 3.

References

[1] Alessandro Antonucci and Giorgio Corani. The multilabel naive credal classifier. *International Journal of Approximate Reasoning*, 83:320–336, 2016.

[2] Arthur Choi and Adnan Darwiche. Dynamic minimization of sentential decision diagrams. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*, 2013.

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

[4] Fabio G. Cozman and Denis Deratani Mauá. On the complexity of propositional and relational credal networks. *International Journal of Approximate Reasoning*, 83:298–319, 2017.

[5] Gert de Cooman, Filip Hermans, Alessandro Antonucci, and Marco Zaffalon. Epistemic irrelevance in credal nets: the case of imprecise Markov trees. *International Journal of Approximate Reasoning*, 51(9): 1029–1052, 2010.

[6] John Grant, Cristian Molinaro, and Francesco Parisi. Probabilistic spatio-temporal knowledge bases: Capacity constraints, count queries, and consistency checking. *International Journal of Approximate Reasoning*, 100:1–28, 2018.

[7] Doga Kisa, Guy Van den Broeck, Arthur Choi, and Adnan Darwiche. Probabilistic sentential decision diagrams. In *Proceedings of the Fourteenth International Conference on the Principles of Knowledge Representation and Reasoning*, 2014.

[8] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[9] Johan Kwisthout, Hans L Bodlaender, and Linda C van der Gaag. The necessity of bounded treewidth for efficient inference in Bayesian networks. In *ECAI*, volume 215, pages 237–242, 2010.

[10] Yitao Liang, Jessa Bekker, and Guy Van den Broeck. Learning the structure of probabilistic sentential decision diagrams. In *UAI*, 2017.

[11] Lilith Mattei, Decio Soares, Alessandro Antonucci, Denis Mauà, and Alessandro Facchini. Exploring the space of probabilistic sentential decision diagrams. In *3rd Workshop of Tractable Probabilistic Modeling*, 2019. Accepted.

[12] Denis Mauá, Fabio G. Cozman, Diarmaid Conaty, and Cassio P. de Campos. Credal sum-product networks. In *Proceedings of the Tenth International Symposium on Imprecise Probability: Theories and Applications*, pages 205–216, 2017.

[13] Denis Deratani Mauá, Diarmaid Conaty, Fabio G. Cozman, Katja Poppenhaeger, and Cassio Polpo de Campos. Robustifying sum-product networks. *International Journal of Approximate Reasoning*, 2018.

[14] Robert Peharz, Antonio Vergari, Karl Stelzner, Alejandro Molina, Martin Trapp, Kristian Kersting, and Zoubin Ghahramani. Probabilistic deep learning using random sum-product networks. *arXiv preprint arXiv:1806.01910*, 2018.

[15] Hoifung Poon and Pedro Domingos. Sum-product networks: A new deep architecture. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 689–690. IEEE, 2011.

[16] Peter Walley. Inferences from multinomial data: learning about a bag of marbles. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1): 3–34, 1996.