International Journal of Approximate Reasoning 126 (2020) 133–157

Contents lists available at ScienceDirect

International Journal of Approximate Reasoning

www.elsevier.com/locate/ijar

Thirty years of credal networks: Specification, algorithms and complexity

Denis Deratani Mauá [a,*], Fabio Gagliardi Cozman [b]

[a] Institute of Mathematics and Statistics, Universidade de São Paulo, Brazil
[b] Escola Politécnica, Universidade de São Paulo, Brazil

ARTICLE INFO

Article history:
Received 20 December 2018
Received in revised form 10 August 2020
Accepted 12 August 2020
Available online 21 August 2020

Keywords:
Imprecise probabilities
Probabilistic graphical models

ABSTRACT

Credal networks generalize Bayesian networks to allow for imprecision in probability values. This paper reviews the main results on credal networks under strong independence, as there has been significant progress in the literature during the last decade or so. We focus on computational aspects, summarizing the main algorithms and complexity results for inference and decision making. We address the question "What is really known about strong extensions of credal networks?" by looking at theoretical results and by presenting a short summary of real applications.

1. Introduction

The now unclassified United States National Intelligence Estimate report number 29-51 concluded that "an attack on Yugoslavia in 1951 should be considered a serious possibility". When asked to quantify the expression "serious possibility", the authors of the report provided odds that ranged from 20/80 to 80/20 in favor of an attack [1]. Translating to probabilities, the experts assessed that

$$0.2 \leq \mathbb{P}(\text{attack}) \leq 0.8\,.$$

This is not an isolated example: imprecision in probability values pervades practical life. Often this imprecision appears when modeling scenarios with many random variables that interact in a non-trivial way.

According to the report, the possibility of an attack was largely dependent on a decision to invade the country. The latter event was influenced by broad Soviet objectives, but also by a possible regime change in Yugoslavia, perhaps due to Tito's assassination, or due to a coup/revolt. A decision to invade should cause a military build-up and an increase in propaganda against supposed Soviet enemies. This intricate set of relations is captured by the directed acyclic graph (DAG) in Fig. 1, where each node is a (binary) random variable.[1] If we could associate a sharp conditional probability value with each value of a variable given each value of those variables that point to it in the graph, we would obtain a Bayesian network [2]. That is, we would have a "probabilistic graphical model" that could be used to calculate the probability of any possible scenario involving the variables in the model (e.g., the probability of an invasion given the observation of military build-up

* Corresponding author.
E-mail address: denis.maua@usp.br (D.D. Mauá).

[1] Note that each one of these events (attack, propaganda, etc) can be associated with a random variable that yields 1 when the event happens and 0 otherwise; to simplify matters, we often do not distinguish between an event and the corresponding random variable that indicates its happening.

https://doi.org/10.1016/j.ijar.2020.08.009
0888-613X/

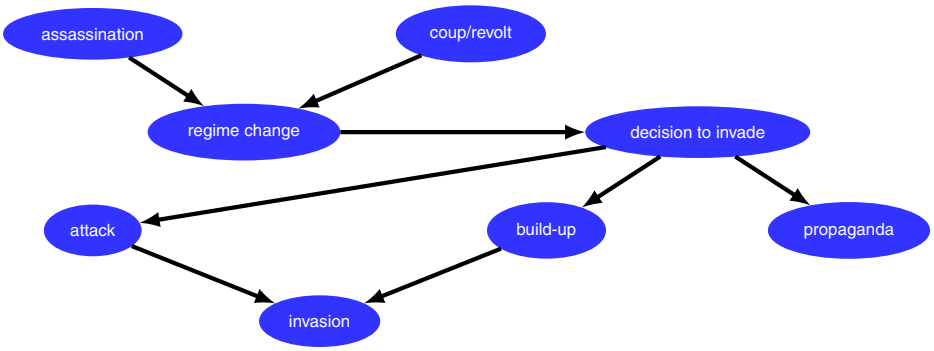


Fig. 1. A simplified graphical representation for the reasoning in the 29-51 report.

and propaganda) [3,4]. If we instead cannot associate a single joint probability distribution with the graph, we obtain a credal network.

In short, credal networks are directed acyclic graphs whose nodes represent random variables associated with "imprecise/indeterminate" probabilistic assessments, and whose arcs (or their absence) represent irrelevance or independence. Credal networks keep the graphical appeal of Bayesian networks [2] while allowing for a more flexible quantification of values. This ability to represent and reason with imprecision in probability values allows credal networks to reach more conservative and robust conclusions than Bayesian networks.

Even though it is difficult to pinpoint when credal networks and their strong extensions first appeared in the literature, it seems fair to say that by 1990 some authors had already touched on their basic features. The literature evolved steadily until 2005, when the second author published a review of graph-based techniques that deal with imprecise probabilities [5]. Since then, research in the field has led to a considerable change in our understanding of credal networks. First, a rich toolbox of algorithms has been put together, containing (fast) approximate techniques and (slow) exact methods. Second, the complexity of various inferences has been studied in depth, so we can identify features that guarantee tractable inference. Third, there is now a significant number of real applications of credal networks in various domains where it is advantageous to encode imprecision in probability values. Finally, the last ten years have witnessed progress also on elicitation of credal networks and on decision making with credal networks.

Several of those more recent results have been discussed in an excellent tutorial on credal networks published in 2014 [6]. Also of note is a previous relevant tutorial focused on the elicitation of credal networks [7]. The present review should be valuable to the novice who has read some of that introductory material, and wants more scholarly detail; this review should also be useful to the researcher who is interested in an up-to-date technical introduction to credal networks. We will examine the main results on the theory of credal networks, including basic (but non-trivial) results about their characterization, algorithmic strategies for producing inferences and computational complexity results. In short, we try to provide some guidance regarding the fair question: "After some thirty years of research on credal networks, what exactly is known about their specification, algorithms and inferential complexity?".

There are many ways to interpret independence relations in a credal network; in this paper we focus on what is known as strong independence and its related concept of strong extension. As we discuss later, there are other well-known extensions for credal networks in the literature [8], but strong extension seems to be the most popular one.

The paper is organized as follows. Section 2 summarizes the literature up until 2005, presenting an abridged version of the survey we have mentioned [5]. Section 3 introduces basic terminology and notation, while Section 4 presents some basic results on credal networks. In Section 5, we provide a fair sample of the algorithms for marginal inference and decision making, while in Section 6 we present results on computational complexity of those tasks. Section 7 contains an overview of some recent applications of credal networks. We finish in Section 8 with comments on recent developments and an optimistic view about the future of credal networks.

2. Credal networks up to 2004: a summary

Several early expert systems and knowledge representation formalisms were developed amidst intense debate concerning the best way to handle uncertainty [9–11]. Bayesian networks appeared by the middle eighties [2] and have since proved that probabilistic modeling can be used successfully in many circumstances. As soon as Bayesian networks started to attract interest, researchers felt the need to generalize them so as to allow the representation of imprecision in probability values. One motivation for this was the difficulty in eliciting the many probabilities required by a Bayesian network. Another motivation was the analysis of robustness to perturbations in Bayesian networks (a parallel line of research has investigated sensitivity to changes in parameters [12,13]). Probably an even stronger motivation, amongst researchers interested in artificial intelligence, was the desire to study many languages in which to represent uncertainty, from Dempster Shafer theory to possibility theory, many of which could be reframed using interval probabilities, interval expectations, or credal sets.

The earliest appearance of Bayesian networks with set-valued parameters seems to be by Lamata and Moral [14]. There was then considerable debate on the meaning of independence for special kinds of assessments, such as interval probabilities, Choquet capacities and qualitative probabilities. The early efforts that adopted set-valued assessments took strong extensions as the sole semantics of any given credal network.

By mid nineties, many consequences of the Markov condition and inference algorithms had been developed for strong extensions. Here *inference* refers to a computation of tight probability bounds for a value of some variable given values of some other variables. Perhaps the most representative reference of that period is the work of Cano et al. [15]. At that point there was great emphasis on inference algorithms that operated by passing messages around nodes, thus mimicking the behavior of inference algorithms for Bayesian networks. Algorithms such as the 2-Updating (2U), for polytrees with binary variables [16], or Tessem's propagation, where interval arithmetics is exploited [17], represented the state-of-art. References that reflect those developments, and that include other graph-theoretical structures, can be found in previous survey papers [5,18].

It became clear by mid nineties that, instead of focusing on special cases such as interval probabilities, one should adopt general credal sets from the outset. Given that the class of closed convex credal sets is closed under elementwise marginalization and conditioning under mild conditions [19], and that most axiomatizations dealing with imprecision in probability values generate convex credal sets [20], most of the literature just assumed closure and convexity throughout. By mid nineties it also became clear that, instead of focusing on a single concept of independence, one must accept that several concepts of independence make sense when credal sets are present; this understanding was championed by the influential work of Walley [21]. The idea that credal networks should be broad enough to encompass several definitions of independence was then proposed by the end of the nineties [22].

Computational experience during the nineties led to the understanding that, instead of trying to adapt Bayesian network message passing algorithms to inference with strong extensions, the most profitable path was to look at inference as an optimization problem. Even though a number of successful message passing techniques have emerged, the main algorithms developed from the late nineties on have been inspired by optimization techniques, as described in Section 5.

3. Background: Bayesian networks and credal sets

Given that credal networks are extensions of Bayesian network that replace conditional probability distributions with credal sets, in this section we review basic concepts for Bayesian networks and credal sets; this section also helps in fixing notation and terminology used later.

3.1. Bayesian networks

In a DAG, a node X is a *parent* of a node Y (conversely, Y is a *child* of X) if there is an arc from X to Y. We denote the parents of a node X by $\mathrm{Pa}(X)$ and its children by $\mathrm{Ch}(X)$. The *in-degree* of a node is the cardinality of $\mathrm{Pa}(X)$. The *ancestors* of a node are its parents, the parents of its parents, and so on, recursively. Similarly, the *descendants* of a node are its children, the children of its children, and so on, recursively; the *nondescendants* are all nodes that are not descendants except the node itself.

Formally, a Bayesian network is a pair containing a DAG whose nodes are random variables in a set $\mathbf{X} = \{X_1, \ldots, X_n\}$, and a probability distribution $\mathbb{P}(\mathbf{X})$ over $\mathbf{X}$. The DAG and the distribution satisfy the *Markov condition*: for every node/variable X, the nondescendant nonparents of X are independent of X given the parents of X. To illustrate the Markov condition, consider again Fig. 1. The structure of the graph implies a number of independence relations. For example, the graph implies that Tito's assassination and coup/revolt are independent. If such independence relation is deemed unreasonable, then edges should be added accordingly. Another independence relation extracted from the graph is: given regime change, then the decision to invade is independent of Tito's assassination. This captures the essence of the Markov condition: once we know the parents of a variable, anything about other ancestors is irrelevant to the variable.

In this paper, we consider only random variables with finite support. When this is the case, the Markov condition implies a particular factorization of the joint probability distribution:

$$\mathbb{P}(X_1 = x_1, \ldots, X_n = x_n) = \prod_{i=1}^{n} \mathbb{P}(X_i = x_i | \mathrm{Pa}(X_i) = \pi_i), \tag{1}$$

whenever all conditional probabilities on the right are well-defined. In the equation above π_i is the projection of $\{x_1, \ldots, x_n\}$ over $\mathrm{Pa}(X_i)$, and if $\mathrm{Pa}(X)$ is empty, then we use $\mathbb{P}(X_i = x_i)$ in lieu of $\mathbb{P}(X_i = x_i | \mathrm{Pa}(X_i) = \pi_i)$.

Because "local" conditional probabilities $\mathbb{P}(X = x | \mathrm{Pa}(X) = \pi)$ suffice to specify any Bayesian network, often a Bayesian network is specified through a DAG and a set of probability assessments $\mathbb{P}(X = x | \mathrm{Pa}(X) = \pi)$ for each X, x and π. This leads to a drastic reduction in the number of parameters one must specify and represent with respect to the joint distribution.

The *moral graph* of a DAG is obtained by connecting nodes with a common child and then dropping arc directions. One celebrated feature of Bayesian networks is that one can detect many independence relations implied by the Markov condition using *d-separation* [2]. The latter is a graph-theoretical criterion that we apply as follows to a given Bayesian network. Suppose we have three sets of variables $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{Z}$. Delete all nodes that are not ancestors to at least one of the

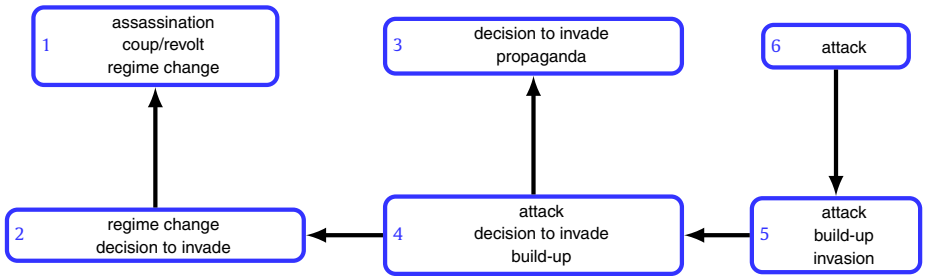


Fig. 2. Rooted tree decomposition for the DAG in Fig. 1.

variables in those sets, and obtain the moral graph of the resulting DAG. Sets **X** and **Y** are d-separated by **Z** if and only if in the resulting undirected graph any path between **X** and **Y** is blocked by **Z** (i.e., it contains a node in **Z**) [23]. The important consequence of this d-separation is that **X** and **Y** are stochastically independent given **Z**. Because d-separation can be quickly computed [24], it is often employed in practice to discard unnecessary variables before any inference is carried out.

The complexity of drawing inferences with Bayesian networks is tightly coupled with the topology of its DAG, and to graph-theoretical concepts such as treewidth.

A *polytree* is a DAG that contains no undirected cycles, that is, the subjacent undirected graph we obtain by ignoring the arc directions has no cycles. A directed tree is a polytree with each node having at most one parent. Polytrees are also called *singly connected* directed graphs. A DAG is *loopy* or *multiply connected* if it is not a polytree. The DAG in Fig. 1 is loopy; the subDAG over the nodes assassination, coup/revolt and regime change is a polytree that is not a tree.

A cycle in an undirected graph contains a chord if there is an edge which connects two nodes of the cycle and is not in that cycle. An undirected graph is *chordal* if and only if every cycle of length four or more has a chord. Every graph can be turned into a chordal graph by inserting edges, a process called *chordalization*.

The *treewidth* of a chordal graph is the size of the largest *clique* minus one, that is, the maximum size of a complete subgraph minus one (a complete graph is a graph such that each pair of nodes is connected). The treewidth of a non-chordal graph is the minimum treewidth over all chordalizations of it. As its name suggests, the treewidth measures the resemblance of an undirected graph to a tree. The treewidth of a directed graph is the treewidth of its moral graph. In the case of polytrees, the treewidth is given by the maximum in-degree of a node (so the treewidth of directed trees is one). All known algorithms for drawing inference in Bayesian networks have a worst-case running time that is at least exponential in the network treewidth.

A *tree-decomposition* of an undirected graph $G = (V, E)$ is a pair (C, T), where C is a collection of subsets $C_i \subseteq V$ of nodes of G, and T is an undirected tree whose nodes are identified with elements of C, and that satisfies the following properties: (1) For every edge (u, v) in E there is a set $C_i \in C$ with $u, v \in C_i$; (2) For every node v in V the subgraph obtained by deleting nodes C_i *not* containing v is a (connected) tree. The width of a tree decomposition is the maximum cardinality of a set C_i minus one. The minimum width over all tree decompositions of a graph is the graph's treewidth. A tree-decomposition can be *rooted* at a certain node $R \in C$ by directing edges away from R; this allows us to refer to parents and children of the nodes in T (note that a rooted T is a directed tree, thus every node has at most one parent). Fig. 2 shows a tree-decomposition for the DAG in Fig. 1 rooted at $R = \{\text{attack}\}$.

3.2. Credal sets

It is often difficult to specify sharp probability values for events due to insufficiency of resources (time, cost, data) or lack of consensus. An alternative is to specify a set of probability distributions that encodes the incompleteness and indeterminacy in the beliefs of the model builder, and arguably leads to more conservative and hence robust conclusions.

We start with some needed concepts of imprecisely specified probability models and their usual notation. A set of probability distributions for a variable X is called a *credal set* and is denoted by $\mathbb{K}(X)$ [19]. Similarly, $\mathbb{K}(X_1, \ldots, X_n)$ denotes a set of joint distributions for variables $X_1, \ldots, X_n$. The convex hull of a set $\mathbb{K}$ is denoted by $\text{co}\,\mathbb{K}$. A finitely generated set is the convex hull of a finite set of points (hence it is closed and convex). A set of conditional probability distributions for a variable X given event A is called a *conditional credal set* and denoted by $\mathbb{K}(X|A)$. The lower probability $\underline{\mathbb{P}}_{\mathbb{K}(X)}(A)$ is equal to $\inf \mathbb{P}(A)$, where the infimum is taken over all possible probability values for event A induced by the distributions in $\mathbb{K}(X)$; similarly we have the lower conditional probability $\underline{\mathbb{P}}_{\mathbb{K}(X|B)}(A|B)$. We also have the upper probability $\overline{\mathbb{P}}_{\mathbb{K}(X)}(A)$ that is $\sup \mathbb{P}(A)$ and similarly upper conditional probabilities. In the operators above, we usually omit the dependence on the credal set when its choice is clear from context and write, for example, $\underline{\mathbb{P}}(A)$ and $\underline{\mathbb{P}}(A|B)$. By marginalizing every distribution in the credal set $\mathbb{K}(X_1, \ldots, X_n)$, we obtain a marginal credal set over a subset of the variables. Whenever $\overline{\mathbb{P}}(Y = y) > 0$, we obtain a conditional credal set $\mathbb{K}(X|Y = y)$ by applying Bayes rule to each distribution in $\mathbb{K}(X, Y)$ such that $\mathbb{P}(Y = y) > 0$. This prescription for conditioning is often called *regular conditioning* [25]. One can find prescriptions in the literature that handle conditioning events differently when they may have zero probability [26–28]. One particularly

popular alternative strategy is to take $\mathbb{K}(X|Y=y)$ to be vacuous[2] whenever $\underline{\mathbb{P}}(Y=y)=0$, and otherwise to apply Bayes rule elementwise as before; because this strategy is obtained with Walley's natural extension [21], we refer to it as *natural* conditioning.[3] In our definitions and results we adopt regular conditioning unless explicitly indicated, and we implicitly assume that the upper probability of the conditioning event is positive.

One basic result is that, if a credal set is convex, both its elementwise marginalization and its elementwise conditioning lead to convex sets [19].

One can specify a credal set by listing constraints on probability values. If a credal set is closed and convex, then it can be described by listing its extreme points (and taking the convex hull). Constraints and extreme points are connected by duality: one can transform one representation into the other using well known algorithms [29]. However, the size of the representation may explode through this transformation: a set of (even linear) constraints may generate an exponential number of extreme points and vice-versa. Some algorithms discussed later assume that the input credal sets are given as sets of constraints, while other algorithms assume the input is a list of extreme points; these are mathematically equivalent, but not computationally interchangeable. Matters are dramatically simpler if we deal with a single binary variable X: in this case, a closed convex credal set $\mathbb{K}(X)$ is fully captured by a single closed interval $[\underline{\mathbb{P}}(X=1), \overline{\mathbb{P}}(X=1)]$.

There are several concepts of independence that apply to sets of probability distributions [28,30]. A conceptually simple concept is *complete independence*: variables X and Y are completely independent with respect to a credal set $\mathbb{K}(X,Y)$ if they are stochastically independent with respect to every joint distribution in $\mathbb{K}(X,Y)$. This definition extends to conditional independence in the trivial way: in essence, it requires elementwise stochastic independence.

If X and Y are completely independent with respect to some $\mathbb{K}(X,Y)$, then in general $\mathbb{K}(X,Y)$ cannot be a convex set [28]. Complete independence seems to require one to accommodate non-convex credal sets. As many principled ways to justify credal sets postulate that they ought to be convex [19,31], there has been steady interest in "convexifying" complete independence, a move that yields *strong independence*.

Variables X and Y are *strongly independent* with respect to a credal set $\mathbb{K}(X,Y)$ if the latter can be written as the convex hull of a credal set for which X and Y are completely independent. When $\mathbb{K}(X,Y)$ is closed and convex, strong independence is equivalent to assuming that X and Y are stochastically independent with respect to each extreme point of $\mathbb{K}(X,Y)$.

Another popular concept is *epistemic irrelevance* [21]: Y is epistemically irrelevant to X given Z when the convex hull of $\mathbb{K}(X|Y=y,Z=z)$ is equal to the convex hull of $\mathbb{K}(X|Z=z)$ for every possible (y,z). Epistemic irrelevance is asymmetric: X may be epistemically irrelevant to Y while the opposite fails [21]. *Epistemic independence* is a symmetrized version of epistemic irrelevance: X and Y are epistemically independent when X is epistemically irrelevant to Y and Y is epistemically irrelevant to X.

Yet another concept of independence is due to Kuznetsov, a generalization of the usual factorization of expectations that is implied by stochastic independence [32]. Little is known about Kuznetsov independence, and the few existing inference algorithms that can handle it rely on optimization to look for bounds on probabilities [33].

4. Credal networks and their strong extensions

A credal network consists of a pair containing a DAG where each node is a variable, and a credal set for those variables, such that, for every node/variable X, the nondescendant nonparents of X are "independent" of X given the parents of X. Obviously the Markov condition has a different effect depending on the concept of independence that is adopted.

Suppose for instance that "independence" in the Markov condition means "epistemic irrelevance". As epistemic irrelevance is not symmetric, the direction of edges in the credal network then gets denser content. During the last decade a rich theory and a powerful set of algorithms have been developed around extensions of credal networks that satisfy the Markov condition with epistemic irrelevance — we do not dwell on that topic as a recent in-depth review paper has covered all necessary territory [8]. As for epistemic independence and Kuznetsov independence, little is known in connection with credal networks (even though credal networks under epistemic independence have received some attention [34]).

In the remainder of this paper we adopt the Markov condition with strong independence; that is, each variable is strongly independent of its nondescendants given its parents.

4.1. Specifying credal networks

The most common way to specify sets of probabilities associated with credal networks is to mimic the strategy used for Bayesian networks. That is, to specify a closed convex credal set $\mathbb{K}(X|\mathrm{Pa}(X)=\pi)$ for each variable X, for each value π of parents $\mathrm{Pa}(X)$. As a very simple example, we might have three binary variables X, Y and Z, the simple DAG

[2] A credal set is *vacuous* if it contains all distributions of that variable.

[3] Yet another strategy is to replace usual probability measures by *full conditional probabilities* and their variants [27], where conditional probability is defined even when the conditioning event has probability zero — this is a move we do not consider in this paper.

and probabilistic constraints

$$\mathbb{P}(X=1) \in [1/10, 1/3], \qquad \mathbb{P}(Y=1) = 4/5, \tag{2}$$

$$\mathbb{P}(Z=1|Y=0) \in [2/5, 3/5], \qquad \mathbb{P}(Z=1|Y=1) \in [7/10, 9/10]. \tag{3}$$

Recall that an interval suffices to specify the credal set for a binary variable. In general a more complex language may be needed to specify credal sets for variables with more than two values (Section 4.4).

When a credal network is specified through a DAG and a set of credal sets $\mathbb{K}(X|\mathrm{Pa}(X)=\pi)$, one for each variable X and each value π of parents of X, we say the network is *separately specified*.

A different specification strategy is to explicitly describe a set of functions $\mathbb{P}(X|\mathrm{Pa}(X))$ for each variable X (note: here $\mathbb{P}(X|\mathrm{Pa}(X))$ is a function both of X and of $\mathrm{Pa}(X)$). We then say that the credal set for X is *extensively specified*, or just *extensive* for short. For instance, one might replace assessments in Expression (3) by two functions $\mathbb{P}_1(Z|Y)$ and $\mathbb{P}_2(Z|Y)$, taking their convex hull as the set of possible functions $\mathbb{P}(Z|Y)$.

A few kinds of non-separately specified credal networks have been explored in the literature. One example is the (large) family of *qualitative probabilistic networks (QPNs)* [35]. A QPN is usually interpreted as a partially specified Bayesian network, where in lieu of the conditional probability values one writes down qualitative constraints amongst these probabilities. The goal is to simplify elicitation efforts. For instance, the qualitative constraint referred to as *positive additive synergy* on a variable X with parents Y and Z imposes

$$\begin{aligned}\mathbb{P}(X=1|Y=1,Z=1)+\mathbb{P}(X=1|Y=0,Z=0) \geq \\ \mathbb{P}(X=1|Y=0,Z=1)+\mathbb{P}(X=1|Y=1,Z=0),\end{aligned}$$

meaning, roughly, that the "concerted" influence of X and Z on X is greater than the sum of their "discordant" influence. Note that constraints are imposed across values of conditioning variables in a non-separately specified scheme. Many other constraints have been proposed to enlarge the expressivity of QPNs [36,37] and to apply them to practical scenarios [38]. Another example of non-separately specified credal networks can be found in the *Bayesian logic* of Andersen and Hooker [39], where a directed graph can be associated with probabilities over events defined with a flexible logical language.[4]

Whatever specification strategy one adopts, one ends up with a set of constraints over probabilities. The largest set of joint probability distributions that satisfies those constraints *and* the Markov condition, where "independence" means "strong independence", is the *strong extension* of the credal network. Of course one might consider sets of joint distributions that are smaller than the strong extension; the strong extension corresponds to a minimum commitment given the assessments.

4.2. The strong extension for closed convex local credal sets

Suppose that one has a separately specified credal network where each variable X is associated with a closed and convex conditional credal set $\mathbb{K}(X|\mathrm{Pa}(X)=\pi)$, for each configuration π of the parents. This is the most common situation discussed in the literature.

Define the *complete extension* of the network, denoted by $\mathbb{K}_C(X_1,\ldots,X_n)$, to be the following (possibly non-convex) set:

$$\left\{\mathbb{P}: \quad \mathbb{P}(X_1=x_1,\ldots,X_n=x_n)=\prod_{i=1}^{n}\mathbb{P}(X_i=x_i|\mathrm{Pa}(X_i)=\pi_i), \right. \\ \left. \text{with } \mathbb{P}(X_i|\mathrm{Pa}(X_i)=\pi_i) \in \mathbb{K}(X_i|\mathrm{Pa}(X_i)=\pi_i)\right\}, \tag{4}$$

where each π_i is the projection of $\{x_1,\ldots,x_n\}$ over $\mathrm{Pa}(X_i)$.

The set $\mathbb{K}_C(X_1,\ldots,X_n)$ and the input DAG satisfy the Markov condition with respect to complete independence: for every X, the nondescendants of X are *completely* independent of X given $\mathrm{Pa}(X)$.

We have that:

Theorem 1. *The convex hull of the complete extension of a credal network is the strong extension of the credal network: it is the largest credal set that satisfies the Markov condition with respect to strong independence. This is also true when we replace each credal set $\mathbb{K}(X_i|Pa(X_i)=\pi_i)$ in Expression (4) by the set of its extreme points.*

The proofs of all theorems are in the appendix.

[4] Non-separately specified constraints also surface in connection with *conservative updating* on probabilistic graphical models, where missing data may lead to credal sets [40–42]; Antonucci et al. [6] present an interesting discussion of this phenomenon.

Hence every extreme point of the strong extension of a separately specified credal network with closed convex local credal sets is built by a combination of "local" extreme points, and thus it behaves like a Bayesian network. Importantly, d-separation in the DAG implies strong independence; in this sense strong extensions mimic the semantics of independence commonly associated with Bayesian networks.[5] Interestingly:

Theorem 2. *Any combination of extreme points from the local credal sets is an extreme point of the strong extension.*

Thus the strong extension can actually be built directly from the extreme points of local credal sets, that is, $\text{co}\,\mathbb{K}_C(X_1,\ldots,X_n)$ is equivalently produced if we replace each local credal set $\mathbb{K}(X_i|\text{Pa}(X_i)=\pi_i)$ in Expression (4) by the set of its extreme points, denoted by $\text{ext}\,\mathbb{K}(X_i|\text{Pa}(X_i)=\pi_i)$. For instance, take the three-variable network with assessments in Expressions (2) and (3). The strong extension is the convex hull of eight distributions, generated by multiplying the extreme points of $\mathbb{K}(X)$, $\mathbb{K}(Z|Y=0)$ and $\mathbb{K}(Z|Y=1)$, and the distribution $\mathbb{P}(Y=1)$.

4.3. A word on extensive local credal sets

Expression (4) focuses on separately specified credal networks; if instead we have an extensive specification, the strong extension is the convex hull of all distributions $\mathbb{P}(\mathbf{X})=\prod_{X_i\in\mathbf{X}}\mathbb{P}(X_i|\text{Pa}(X_i))$ built so that each *function* $\mathbb{P}(X_i|\text{Pa}(X_i))$ is an extreme point of the extensive specification related to X_i.

Note that if we have a separately specified credal network we can always build a set of conditional probability functions $\mathbb{P}(X_i|\text{Pa}(X_i))$ by combining all extreme points of each $\mathbb{K}(X_i|\text{Pa}(X_i)=\pi)$ across all π. Once we do that, the strong extension of these latter extensively specified credal sets will be exactly the strong extension of the original separately specified credal network. Such a transformation is however not harmless from a computational point of view: the extensive specification related to a variable X_i may be exponentially larger than the separately specified credal sets for X_i.

4.4. Specification languages for credal networks

There has been relatively little work on specification languages for credal networks. The contrast with research in Bayesian networks is striking. Since the appearance of the first inference engines for Bayesian networks, there has been active work in languages that encode probability distributions. One example is the BUGS package [47], where a declarative language is used to describe the structure and the probability values of any Bayesian network. There are other visual languages such as DAPER [48], textual languages such as BLOG [49], and languages that explore logic programming [50,51] among other proposals. Almost none of this activity can be found in connection with credal networks, despite the fact that there is arguably more structure to be explored: possible ways to list extreme points and exploit regularities amongst them; possible ways to encode qualitative and quantitative constraints, conceivably by recycling constructs from constraint programming. Existing packages for credal networks usually adopt simple strategies for specification where each local credal set is represented as a set of linear inequalities, or where each extreme point is a distribution represented as a table of probability values.

It has recently been noted that credal sets can be specified using probabilistic logic programming [52], but this insight has not been used to specify credal networks. Another relatively recent effort mixes Boolean operators and interval-valued probabilities in specifications such as [53]:

$$\text{regime change} = \text{assassination} \vee (\text{coup/revolt} \wedge \neg\text{inhibitor}),$$
$$\mathbb{P}(\text{inhibitor}) \in [0.9, 1],$$

where the first sentence reads "regime change obtains if and only if there is an assassination or a coup/revolt (provided the latter is not inhibited)" and the second sentence reads "probability of inhibitor is no smaller than 0.9". In this particular example one obtains a Noisy-OR gate with interval-valued inhibitory probabilities.

There seems to be significant room for refined and extended specification strategies in the future.

5. Algorithms for marginal inference and decision making

The most common computation applied to credal networks is *marginal inference*, where we must compute bounds for the marginal probability of a value of a *query* variable conditional on some *evidence* on some other variables. For instance, in the credal network representing a possible attack on Yugoslavia in 1951, one might be interested in the probability of an attack after observing an increase in propaganda.

Credal networks are also used to support decision making; there the problem is to select actions that are endorsed by some appropriate criteria — maximality, E-admissibility, minimality.

In this section we review the most effective known algorithms for marginal inference and decision making with credal networks.

[5] Other concepts of independence, such as epistemic independence, do not necessarily interact well with d-separation [43], and alternatives to d-separation have been designed for them [44–46].

5.1. Marginal inference

Given a credal network over variables $\mathbf{X} = \{X_1, \ldots, X_n\}$, an event of interest $Z = z$ and some evidence $\mathbf{Y} = \mathbf{y}$, we are interested in obtaining:

$$\underline{\mathbb{P}}(Z = z|\mathbf{Y} = \mathbf{y}) \text{ and } \overline{\mathbb{P}}(Z = z|\mathbf{Y} = \mathbf{y}), \tag{5}$$

where $\{Z\}$ and $\mathbf{Y}$ are assumed to be disjoint subsets of $\mathbf{X}$ such that $\overline{\mathbb{P}}(\mathbf{Y} = \mathbf{y}) > 0$, and we take the lower and upper probabilities with respect to the regular conditioning of the strong extension of the network.

The following result is fundamental as it shows that marginal inference can be accomplished by operating only on distributions that satisfy the Markov property:

Theorem 3. *The lower/upper probability in Expression (5) are obtained, respectively, by*

$$\inf / \sup \ \frac{\sum_{\mathbf{x}'} \prod_{i=1}^{n} \mathbb{P}(X_i = x_i|Pa(X_i) = \pi)}{\sum_z \sum_{\mathbf{x}'} \prod_{i=1}^{n} \mathbb{P}(X_i = x_i|Pa(X_i) = \pi)}, \tag{6}$$

where the sum in the numerator and the inner sum in the denominator are over the values of the set of marginalized *variables* $\mathbf{X}' = \mathbf{X} \setminus (\{Z\} \cup \mathbf{Y})$*, the outer sum in the denominator is over the values of the query variable* Z*, and the optimization is over the distributions from the complete extension such that* $\mathbb{P}(\mathbf{Y} = \mathbf{y}) > 0$*.*

The most popular type of specification for credal networks employs finitely generated local credal sets; for these cases we can further reduce the search space in Expression (6)[6]:

Theorem 4. *Suppose a credal network is separately specified with closed convex local credal sets. Then the solution to Equation (6) is attained at some extreme point* $\mathbb{P}$ *of the strong extension such that* $\mathbb{P}(\mathbf{Y} = \mathbf{y}) > 0$*; hence it is attained at some combination of extreme points of the local credal sets* $\mathbb{K}(X|Pa(X) = \pi)$*.*

Solving the optimization in Equation (6) presents two challenges. First, the large (exponential) number of combinations of local extreme points, assuming one has already obtained these. Second, the large (exponential) numbers of terms in the sum in the denomination and numerators. There are essentially two approaches to the problem.

One approach is to cast the problem as a combinatorial search over the extreme distributions of the local credal sets. This approach assumes a vertex-based representation of credal sets. Typically these search schemes reproduce, at some level, the message-passing schemes that succeed with Bayesian networks, with the condition that now messages have to be set-valued (perhaps sets of real numbers, perhaps sets of functions). Message passing schemes received attention in particular during the nineties. Among the insights developed in that investigation, we must mention the 2U algorithm, a clever exact algorithm that solves a particular class of inference problems in polynomial time. Many other ideas in combinatorial search have been explored since then. Notable examples of this approach are the integer-programming formulation of de Campos and Cozman [54], the algebraic system of Mauá et al. [55], the local search methods of Cano et al. [56].

The other approach is to cast the problem as a continuous optimization, and to adapt known techniques from the numerical optimization literature for the resulting (difficult) optimization problem. Notable methods in this category are the multilinear programming reformulation of de Campos and Cozman [57], and the linear programming approximation of Antonucci et al. [58]. An important feature of this approach is the ability to naturally handle both the constraint-based and vertex-based representations of credal sets.

We next present some of the most general and effective approaches to marginal inference, emphasizing the ones that have demonstrated leading performance in the literature. Note that we assume separately specified networks with finitely generated local credal sets (that is, each set is the convex hull of finitely many distributions), unless explicitly indicated.

5.1.1. Message-passing: basic ideas and the 2U algorithm

One (intractable) simple idea is to generate all possible extreme points of the strong extension by combining the local extreme points, and for each one run standard message passing algorithms of Bayesian network. A slightly more efficient approach is to exploit the redundancies among these shared Bayesian network inferences, and to propagate messages that instead of containing a single function, contain sets of functions that correspond to (some of) the extreme points.

We have already mentioned some important landmarks within this approach in Section 2. For instance, the whole exhaustive message passing scheme is captured by the algorithm of Cano et al. [15], while approximate inference is obtained by Tessem's algorithm, which propagates only bounds for each set [17].

There is a very distinguished case where message propagation can be performed both efficiently and accurately: this happens when the DAG is a polytree and all variables are binary. The well known 2U algorithm exploits this pocket of

[6] The proof of Theorem 4, presented in the appendix, also shows that regular conditioning on a finitely generated credal set produces a finitely generated conditional credal set; if one starts with the latter fact, then Theorem 4 is a direct corollary.

tractability [16]; its main ideas can be explained with relative ease.[7] A brief description of Pearl's propagation algorithm for inference with polytrees is needed first [2]. Say we have a Bayesian network whose underlying graph is a polytree and some evidence $\mathbf{Y} = \mathbf{y}$. To facilitate the notation, assume that the evidence sets only the values of leaves of the graph with a single parent. This can be always made true: if X is a variable set by evidence that does not satisfy those conditions, then insert a fresh "dummy" node X' with only X as parent and such that $\mathbb{P}(X' = 1|X = x) = 1$ and $\mathbb{P}(X' = 1|X \neq x) = 0$, where x is the original value set for X by the evidence. Then set evidence $X' = 1$ (and remove X from the evidence set). One can check that any marginal inference has its value unchanged by this transformation.

In Pearl's algorithm one focuses on one variable X at a time, and passes messages to its neighbors as follows. Assume that X has parents $U_1, \ldots, U_m$ and children $Y_1, \ldots, Y_n$, and no evidence is set for X.

- From X to each child Y_j, send the message:

$$f_{X \to Y_j}(x) = \prod_{k \neq j} f_{Y_k \to X}(x) \sum_{\pi} \mathbb{P}(X = x|\mathrm{Pa}(X) = \pi) \prod_{i=1}^{m} f_{U_i \to X}(u_i)\,,$$

  where π ranges over the values of $\mathrm{Pa}(X)$ and u_i is the projection of π on $U_i \in \mathrm{Pa}(X)$.
- From X to each parent U_i, send the message:

$$f_{X \to U_i}(u_i) = \sum_{x} \prod_{j=1}^{m} f_{Y_j \to X}(x) \sum_{\pi \sim u_i} \mathbb{P}(X = x|\mathrm{Pa}(X) = \pi) \prod_{k \neq i} f_{U_k \to X}(u_k)\,,$$

  where $\pi \sim u_i$ are the configurations whose projection on U_i is u_i, and u_k is the projection π on U_k.

In the expressions above, the products are identical to one if the corresponding range is empty (e.g. if X is a root then there are no messages $f_{U_i \to X}$). So for example, if X is a leaf with a parent U and no evidence is set for X, then $f_{X \to U} = 1$.

Consider now the case of a node X which has value x fixed by the evidence. Recall that we assumed that evidence was set only for leaf nodes with a single parent. Then X sends message $f_{X \to U}(u) = \mathbb{P}(X = x|U = u)$ to its single parent U.

Since the graph is a polytree, it is always possible to find a node that can send a message to a neighbor based on the messages it has received. For example, if X is a leaf with a single parent U then its message $f_{X \to U}$ does not depend on any other message. Another example is the case of a root node X with a single child Y, which can send message $f_{X \to Y}(x) = \mathbb{P}(X = x)$.

After all messages are sent, for each node X in the network we have:

$$\mathbb{P}(X = x|\mathbf{Y} = \mathbf{y}) \propto \prod_{j=1}^{m} f_{Y_j \to X}(x) \sum_{\pi} \mathbb{P}(X = x|\mathrm{Pa}(X) = \pi) \prod_{i=1}^{n} f_{U_i \to X}(u_i)\,.$$

The message-passing algorithm above can be extended for credal networks by considering the whole set of messages $f_{A \to B}$ that go through any edge $A \to B$ as we vary the distributions in the complete extension. That fact by itself does not render the problem easy. A key insight behind the 2U algorithm is that these sets of messages can be computed efficiently with a very similar message passing scheme for the case of binary variables. The 2U algorithm operates as follows.

- From X to each child Y_j, send the message:

$$\underline{f}_{X \to Y_j}(x) = \left(1 + \frac{(1 - \mu(x))}{\mu(x)} \frac{1}{\prod_{k \neq j} \underline{f}_{Y_k \to X}(x)}\right)^{-1},$$

  where

$$\mu(x) = \min \sum_{\pi} \underline{\mathbb{P}}(X = x|\mathrm{Pa}(X) = \pi) \prod_{i=1}^{m} f_{U_i \to X}(u_i)\,,$$

  with the minimization being performed over the extreme points of the intervals $[\underline{f}_{U_i \to X}(u_i), \overline{f}_{U_i \to X}(u_i)]$. Upper bounds are obtained by replacing minimization/lower bounds with maximizations/upper bounds in the equations above.
- From X to each parent U_i, send the message:

$$\underline{f}_{X \to U_i}(u_i) = \min \frac{\lambda(x)\rho(x|u_i) + \lambda(\bar{x})(1 - \rho(x|u_i))}{\lambda(x)\rho(x|\bar{u}_i) + \lambda(\bar{x})(1 - \rho(x|\bar{u}_i))}\,,$$

[7] The fact that the 2U algorithm can be derived as a message passing algorithm has been mentioned before [59], but the relevant details have not appeared in the literature.

where

$$\rho(x|u_i) = \sum_{\pi \sim u_i} \mathbb{P}(X = x|\text{Pa}(X) = \pi) \prod_{k \neq i} f_{U_k \to X}(u_k)$$

and the minimizations are performed over

$$\lambda(x) \in \left\{ \prod_{j=1}^{n} \underline{f}_{Y_k \to X}(x), \prod_{j=1}^{n} \overline{f}_{Y_k \to X}(\bar{x}) \right\},$$
$$\mathbb{P}(X = x|\text{Pa}(X) = \pi) \in \text{ext}\,\mathbb{K}(X|\text{Pa}(X) = \pi),$$
$$f_{U_k \to X}(u_k) \in \left\{ \underline{f}_{U_k \to X}(u_k), 1 - \underline{f}_{U_k \to X}(\bar{u}_k) \right\}.$$

As in Pearl's algorithm, the products in the above expressions are replaced with value 1 when their ranges are empty.

If X is a leaf node with evidence $X = x$, then it sends message

$$\underline{f}_{X \to U}(u) = \min_{\mathbb{P} \in \text{ext}\,\mathbb{K}(X|U=u)} \frac{\mathbb{P}(X = x|U = u)}{\mathbb{P}(X = x|U = \bar{u})}$$

to its single parent U. This is equivalent to setting $\lambda(x) = 1$ and $\lambda(\bar{x}) = 0$ in the equation for a node X with no evidence.

At the end one can compute exact bounds on $\mathbb{P}(X = x|\mathbf{Y} = \mathbf{y}) = (1 + (1/\mu(x) - 1)\lambda(\bar{x})/\lambda(x))^{-1}$ for each variable X in the network by optimizing over variables $\mu(x)$ and $\lambda(x)$ as in the equations above.

The expressions above assume that probabilities are always positive. Regular conditioning with zero probabilities is obtained by extending the domain of the messages to include ∞, and by extending sums and products accordingly (e.g. using $1/0 = \infty$, $1/\infty = 0$, $1 + \infty = \infty$). This can be done without hurting computational performance [16].

The 2U algorithm has inspired a few other approximate methods. The Loopy 2U (L2U) algorithm focuses on credal networks with binary variables where the underlying graph may fail to be a polytree: in this case one can always keep iterating the interval-valued messages, hoping to reach convergence [60]. Such a scheme corresponds to belief propagation for Bayesian networks, where Pearl's propagation is run until messages converge or until some prescribed time limit is reached, an idea that may seem far-fetched but that often obtains reasonable accuracy [61]. This approach has been further extended in the Generalized L2U (GL2U) for credal networks that may contain non-binary variables; the idea is to turn non-binary variables into connected clusters of binary variables that are then processed by L2U [62]. A different possibility is to "cut" edges so as to produce approximate credal networks whose underlying graphs are polytrees that can be processed by 2U; the IPE algorithm explores this path [60]. These approximate schemes resemble in several ways variational methods, and indeed some preliminary efforts have been made to apply variational methods to credal networks [60].

5.2. Message-passing: algebraic variable elimination

As we discuss later in Section 6, the 2U algorithm is a truly isolated island of tractability for inference with strong extensions. To handle other situations one may resort for instance to stochastic search techniques. Cano and Moral [63] employed genetic programming to find approximate solutions. Cano et al. [64] implemented a local search with simulated annealing in order to avoid poor local optima. These approximate methods were later used in conjunction with outer approximation algorithms to develop a branch-and-bound algorithm [56].

A more sophisticated approach, based on the manipulation of sets of pairs of functions, was developed by Mauá et al. [55]. The approach is able to produce both exact and approximate solutions, and has been shown to outperform the integer programming formulation (described later) on a large collection of synthetic problems. We focus on this algorithm in the remainder of this subsection; before describing it, we must review the operations behind variable elimination in the context of Bayesian networks.

In short, variable elimination extends Pearl's message passing algorithm to networks of arbitrary topology.[8] So let (C, T) be a tree-decomposition for the DAG of a Bayesian network and $R \subset C$ be such that $Z \in R$. Assume with no loss of generality that $R = \{Z\}$ (otherwise we can add a new node to T satisfying such property and connect it to R and to no other node). Root T at R, orienting the arcs away from R. The variable elimination procedure in Algorithm 1 computes $\mathbb{P}(Z = z|\mathbf{Y} = \mathbf{y})$ for a given rooted tree decomposition (C, T) and set of conditional probability mass functions $\Theta_0 = \{p(X|\text{Pa}(X)) : X \in \mathbf{X}\}$.[9] The functions $\delta_y(Y)$ in the algorithm denote the Kronecker delta function that returns one at $Y = y$ and zero elsewhere. The algorithm starts by creating and inserting these functions into Θ_0. This somewhat uncommon step is only a convenience, as it allows us to dispense with the need of either fixing the values of evidences in the initial distributions in Θ_0, or to treat evidence variables separately. The second "for loop" of the algorithm is the elimination step: at each iteration, the

[8] The variable elimination we present differs marginally from other algorithms such as bucket elimination [65], junction tree propagation [66], Shenoy-Shaffer algorithm [67], and the collect algorithm [68].

[9] Note that $p(X|\text{Pa}(X))$ represents a function over the values of X and $\text{Pa}(X)$.

Algorithm 1: Variable elimination.

Input: Tree decomposition (C, T) rooted at $R = \{Z\}$, set of functions $\Theta_0 = \{p(X|\text{Pa}(X)) : X \in \mathbf{X}\}$, query $Z{=}z$, evidence $\mathbf{Y}{=}\mathbf{y}$
Output: The value of $\mathbb{P}(Z{=}z|\mathbf{Y}{=}\mathbf{y})$
for $Y \in \mathbf{Y}$ **do**
  Add function δ_y to Θ_0, where y is the corresponding value of Y in $\mathbf{y}$;
end
Let $C_1, \ldots, C_m = R$ be a reverse topological ordering of T ;
for $i = 1, \ldots, m-1$ **do**
  Obtain $\mathbf{X}_i = C_i \setminus \text{Pa}(C_i)$ and $\Gamma_i = \{f \in \Theta_{i-1} : X \in \mathbf{X}_i, f \text{ is a function of } X\}$;
  Compute $f_i(C_i \cap \text{Pa}(C_i)) = \sum_{\mathbf{x}_i} \prod\{f \in \Gamma_i\}$;
  Update $\Theta_i = (\Theta_{i-1} \setminus \Gamma_i) \cup \{f_i\}$;
end
Let $f_m(Z) = \prod\{f \in \Theta_{m-1}\}$;
return $f_m(z)/\sum_{z'} f_m(z')$

set of variables $\mathbf{X}_i$ is marginalized out of the product of all functions in the current set Θ_{i-1} with $\mathbf{X}_i$ in their domains.[10] A new set Θ_i is then generated replacing all multiplied functions by this new function f_i. The last step, multiplies all remaining functions in Θ_{m-1} and normalizes the result. This produces the desired conditional probability inference. Note the resemblance of the marginalization operation that computes f_i and the computation of the function $f_{X\to Y}$ in Pearl's message passing algorithm.

To better grasp the steps of variable elimination, let A, C, R, D, O, S, B, I denote binary variables representing, respectively, events attack, coup/revolt, regime change, decision to invade, propaganda, assassination, build-up and invasion, respectively. Running variable elimination with the tree-decomposition in Fig. 2, query $A{=}1$, evidence $O{=}1$, and conditional distributions

$$\Theta_0 = \{p(S), p(C), p(R|S,C), p(D|R), p(O|D), p(A|D), p(B|D), p(I|A,B)\}$$

generate functions

$$f_1(R) = \sum_{s,c} p(s)p(c)p(R|s,c)\,, \qquad f_2(D) = \sum_r p(D|r)f_1(r)\,,$$

$$f_3(D) = \sum_o p(o|D)\delta_1(o)\,, \qquad f_4(A,B) = \sum_d p(A|d)p(B|d)f_2(d)f_3(d)\,,$$

$$f_5(A) = \sum_{b,i} p(i|A,b)f_4(A,b)\,,$$

and outputs $\mathbb{P}(A{=}1|O{=}1) = f_5(1)/\sum_a f_5(a)$.

Now return to credal networks. To describe the Algebraic Variable Elimination algorithm, we need to introduce a few more definitions. A *potential* $\phi(\mathbf{Y})$, where $\mathbf{Y}$ is a set of variables, is a finite set of pairs (p, q), where p and q are nonnegative real-valued functions of $\mathbf{Y}$. Each pair of functions (p, q) in a potential represent a possible solution obtained by some configuration of the extreme points of local credal sets; the function p is used to compute the probability $\mathbb{P}(Z \neq z, \mathbf{Y} = \mathbf{y})$, while the function q is used to compute the probability $\mathbb{P}(Z = z, \mathbf{Y} = \mathbf{y})$. The product of two potentials $\phi(\mathbf{Y})$ and $\psi(\mathbf{Z})$ is:

$$\phi(\mathbf{Y}) \cdot \psi(\mathbf{Z}) := \{(p \cdot r, q \cdot s) : (p, q) \in \phi, (r, s) \in \psi\}\,.$$

The sum-marginal $\sum_{\mathbf{Z}} \phi(\mathbf{Y})$ of a potential $\phi(\mathbf{Y})$ with respect to a set of variables $\mathbf{Z} \subseteq \mathbf{Y}$ is obtained by marginalization of each element in each pair:

$$\sum_{\mathbf{Z}} \phi(\mathbf{Y}) := \left\{\left(\sum_{\mathbf{Z}} p(\mathbf{Y}), \sum_{\mathbf{Z}} q(\mathbf{Y})\right) : (p, q) \in \phi\right\}.$$

We also define a *pruning* operation that potentially reduces the cardinality of a potential. The potential $\max \phi(\mathbf{Y})$ returns the set of nonzero maximal elements of $\phi(\mathbf{Y})$ under the partial order that compares the first functions of two potentials with $\leq$ and the second functions with $\geq$, that is:

[10] The product of real-valued functions $f(\mathbf{X})$ and $g(\mathbf{Y})$ is the function $h(\mathbf{Z})$ such that $\mathbf{Z} = \mathbf{X} \cup \mathbf{Y}$ and $h(\mathbf{z}) = f(\mathbf{x}) \cdot g(\mathbf{y})$ for all $\mathbf{z}$ with $\mathbf{x}$ and $\mathbf{y}$ denoting the projection of $\mathbf{z}$ on $\mathbf{X}$ and $\mathbf{Y}$, respectively.

Algorithm 2: Credal Variable Elimination.

Input: Local extreme distributions $\text{ext}\,\mathbb{K}(X|\text{Pa}(X))$, query $Z{=}z$ and Evidence $\mathbf{Y}{=}\mathbf{y}$
Output: The value of $\max p(Z{=}z|\mathbf{Y}{=}\mathbf{y})$
Let Θ_0 be an initially empty set;
Add potential $\phi'_Z = \{(1-\delta_z, \delta_z)\}$ to Θ_0;
for $X \in \mathbf{X}$ **do**
  Add potential $\phi_X = \{(p,p) : p \in \text{ext}\,K(X|\text{Pa}(X))\}$ to Θ_0;
  if $Y \in \mathbf{Y}$ **then**
    Add potential $\phi'_Y = \{(\delta_y, \delta_y)\}$ to Θ_0;
  end
end
Construct tree decomposition (C, T) rooted at $R = \{Z\}$;
Let $C_1, \ldots, C_m = R$ be a reverse topological ordering of T;
for $i = 1, \ldots, m-1$ **do**
  Obtain $\mathbf{X}_i = C_i \setminus \text{Pa}(C_i)$ and $\Gamma_i = \{\psi \in \Theta_{i-1} : X \in \mathbf{X}_i, \psi \text{ is a function of } X\}$;
  Compute $\psi_i(C_i \cap \text{Pa}(C_i)) = \max \sum_{Z_i} \prod\{\phi \in \Gamma_i\}$;
  Update $\Theta_i = (\Theta_{i-1} \setminus \Gamma_i) \cup \{\psi_i\}$;
end
Let $\psi_m(Z) = \prod\{\psi \in \Theta_{m-1}\}$;
return $\max\{q/(p+q) : (p,q) \in \psi_m\}$

$$\max \phi(\mathbf{Y}) = \Big\{ (p,q) \in \phi \setminus \{(0,0)\} : \text{there is no } (r,s) \in \phi \setminus \{(p,q)\} \text{ satisfying } r \le p \text{ and } s \ge q \Big\}. \tag{7}$$

Note that the operation above effectively performs regular conditioning by removing distributions that assign probability zero to the evidence.

We abuse notation and write $\text{ext}\,\mathbb{K}(X|\text{Pa}(X))$ to denote also the set of all conditional probability mass functions $p(X|\text{Pa}(X))$ such that $p(x|\pi) = \mathbb{P}(X = x|\text{Pa}(X) = \pi) \in \text{ext}\,\mathbb{K}(X|\text{Pa}(X) = \pi)$ (i.e., $\text{ext}\,\mathbb{K}(X|\text{Pa}(X))$ is an extensive set of functions).

Marginal inference can be accomplished by the variable elimination procedure described in Algorithm 2. The algorithm creates a singleton potential ϕ'_Z to represent the query, a potential ϕ_X for each $\text{ext}\,\mathbb{K}(X|\text{Pa}(X))$ as discussed, and singleton potentials ϕ'_Y for representing the evidence. The only differences with respect to the standard variable elimination procedure are that the operations involve potentials (that is, sets of pairs of functions), and that pruning is applied at each iteration to discard non-maximal elements.

To obtain the lower probability, we can either modify the potential ϕ'_Z to be $(\delta_z, 1-\delta_z)$, so that the algorithm returns $\overline{\mathbb{P}}(Z \neq z|\mathbf{Y} = \mathbf{y})$, from which we can compute $\underline{\mathbb{P}}(Z = z|\mathbf{Y} = \mathbf{y}) = 1 - \overline{\mathbb{P}}(Z \neq z|\mathbf{Y} = \mathbf{y})$, or modify the pruning operation, so that max is replaced by min:

$$\min \phi(\mathbf{Y}) = \Big\{ (p,q) \in \phi \setminus \{(0,0)\} : \text{there is no } (r,s) \in \phi \setminus \{(p,q)\} \text{ satisfying } r \ge p \text{ and } s \le q \Big\}. \tag{8}$$

The complexity of the algorithm is given by the width of the tree decomposition used (which is lower bounded by the network treewidth) and the cardinality of the potentials ψ_i generated during the bottom loop of the algorithm. While obtaining a minimum-width tree-decomposition is NP-hard there are effective heuristics that most often produce small width tree decompositions (if they exist). In the worst-case, the size of ψ_i is exponential in the cardinalities of the input sets $\text{ext}\,\mathbb{K}(X|\text{Pa}(X))$. In practice, Mauá et al. [55] showed that the cardinality of ψ_i is most often low enough to be handled by current computers within reasonable time and memory limits.

By relaxing the maximality criteria to allow elements to be *slightly* dominated by some other element, Mauá et al. [55] showed that the same algorithm can be adapted to provide a provably good approximation algorithm, that is, a procedure that outputs a solution whose error is at most a given value $\epsilon > 0$. The authors showed that the same algorithm runs in time polynomial in the input and in $1/\epsilon$ on networks of bounded treewidth and bounded variable cardinality. Thus, the variable elimination algorithm with ϵ-maximality pruning produces a fully polynomial-time approximation scheme (FPTAS). The algorithm is optimal in the sense that polynomial-time provably optimal algorithms for marginal inference do not exist when variables can take arbitrarily many states, unless P equals NP [69]. However, the algorithm has large hidden constants, and in practice the performance of the approximation version is comparable to that of the exact version.

5.2.1. Multilinear programming

The algebraic variable elimination obtains state-of-the-art results when credal sets are specified as finite sets of distributions (not necessarily extreme points), or when one such representation can be efficiently obtained. This is the case, for example, when credal sets are estimated from data independently using the Imprecise Dirichlet Model [70], when they are obtained by perturbation of a distribution [71], or when they are fabricated so as to conservatively model data missing not at random [40].

In many other cases, however, credal sets are more conveniently represented by sets of linear constraints on probability values [38,72,73]. As discussed in Section 3.2, converting from such form to a vertex-based form can lead to a drastic increase in size. A better approach in these cases is to formulate the inference problem as a nonlinear optimization subject to linear constraints.

Expression (6) defines a fractional-multilinear program over variables $\mathbb{P}(X = x|\text{Pa}(X) = \pi)$ and linear constraints $\mathbb{P}(X|\text{Pa}(X)) \in \mathbb{K}(X|\text{Pa}(X))$. This connection however is of little use for computing marginal inferences for most practical models, as the sums in the numerator and denominator are carried over a very large an exponential number of terms. Nevertheless, when the treewidth of the network is small, the program can be efficiently reformulated as a multilinear program by means of a symbolic variable elimination procedure.

The gist of the multilinear programming reformulation is to look at the functions f_i produced during variable elimination (Algorithm 1) as equality constraints on the optimization variables $\mathbb{P}(X = x|\text{Pa}(X) = \pi)$. When the treewidth of the network is small, a succinct multilinear programming formulation of the marginal inference problem is given by

$$\text{optimize} \qquad f'_m(z)\,, \tag{9a}$$

$$\text{subject to} \qquad f'_m(Z) = t \cdot f_m(Z)\,, \quad \sum_{z'} f'_m(z') = 1\,, \tag{9b}$$

$$f_i(C_i \cap \text{Pa}(C_i)) = \sum_{\mathbf{x}_i} \prod \{f \in \Gamma_i\}\,, \quad i = 1, \ldots, m-1\,, \tag{9c}$$

$$p(X_i|\pi_i) \in \mathbb{K}(X_i|\pi_i)\,, \quad \forall X_i \in \mathbf{X}, \pi_i \sim \text{Pa}(X_i)\,. \tag{9d}$$

Equations (9b) encode the Charnes-Cooper transformation of fractional to linear expressions [74]. The normalization equality, in particular, discards any distribution assigning probability zero to evidence (such a distribution would produce f_m everywhere equal to zero), hence naturally leads to regular conditioning. Equation (9c) represents a set of equations, one for each configuration of the random variables $C_i \cap \text{Pa}(C_i)$, for $i = 1, \ldots, m-1$. Similarly, Expression (9d) represents a multilinear constraint-based specification of the credal set $\mathbb{K}(X_i|\pi_i)$ in terms of optimization variables $p(x_i|\pi_i)$. Thus, the number of constraints is at most exponential in the width of the tree decomposition.

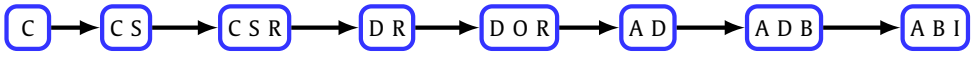
The above multilinear formulation can be fed into any multilinear programming solver. These solvers usually implement a branch-and-bound procedure based on convex relaxations to obtain outer bounds, similar to the way that most integer programming solvers operate (except that generating the convex relaxations of multilinear programs is more intricate). While the performance of multilinear solvers has been improving, they are still slow compared to other optimizers (e.g., mixed-linear program solvers) and suffer from numerical instability. De Campos and Cozman [57] suggested using efficient approximate inference methods such as A/R+ [75] for producing inner and outer bounds on the solution, speeding up the convergence of branch-and-bound (building and refining the interval propagation methods of Tessem [17]). A similar suggestion was made by Ide and Cozman [60] for their approximate IPE algorithm, which is able to produce outer bounds.

A short note: while the multilinear formulation just presented is based on the work of de Campos and Cozman [57], the presentation here differs from their work in that we use tree decompositions to guide the (symbolic) variable elimination procedure, instead of (symbolic) bucket elimination (these methods are largely equivalent, with tree decomposition being slightly more flexible as we can sum-out multiple variables at once, something that is not possible in standard bucket elimination); the use of tree decompositions here is mainly to facilitate notation, as the domains of the functions generated can be related to the cliques of the tree.

5.2.2. Digression: an integer program formulation

With a few clever changes the multilinear programs described in the previous section may be turned into linear integer programs [54]. Even though the resulting scheme has not generated leading results in empirical testing, it may suggest new ideas for further investigation, so we summarize the main points here.

A *path decomposition* is a tree decomposition (C, T) where T is a path, that is, each node has at most two neighbors. By using a path decomposition in lieu of the tree-decomposition we can ensure that each constraint in the form of Expression (9c) involves a product of a function f and a function $p(x|\pi)$ [54]. For example, if we use the rooted path-decomposition



with query $A = 1, O = 1$ and no evidence, a symbolic variable elimination generates a multilinear program where γ must be optimized subject to a variety of constraints such as

$$f_1(A,B) = \sum_i p(i|A,B)\delta_1(A), \qquad f_2(A,D) = \sum_b f_1(A,b)p(b|D).$$

The important point is that each constraint contains only products of two variables, one of which is a variable $p(x|\pi)$ related to the specification of the local credal sets.

If the local credal sets are finitely generated then any $p(X|\pi)$ can be written as a convex combination of functions $p_1(X|\pi), \ldots, p_K(X|\pi)$ obtained from the extreme distributions in the set. Since the result of the inference is attained at the extreme distributions of the local credal sets, the value of the inference is not changed if we constraint $p(X|\pi)$ by

$$p(X|\pi) = \sum_{k=1}^{K} b_k \cdot p_k(X|\pi), \quad b_k \in \{0,1\}, \quad \sum_{k=1}^{K} b_k = 1.$$

By substituting the $p(X|\pi)$ in Expression (9c) by the expression in the right-hand side of the equation above, we end up with a bilinear constraint where each term is a multiplication of a linear variable $f_j(\mathbf{w})$, a Boolean variable b_k and a constant $p_k(x|\pi)$.

A well-known, and very useful, fact is that the product of a Boolean and a linear variable can be transformed into linear constraints. Let r be a variable taking values in $[0,1]$ and b be a $\{0,1\}$-valued variable. Then the product $r \cdot b$ is equivalent to a real-valued variable t subject to

$$0 \le t \le b, \qquad r - 1 + b \le t \le r.$$

Thus the whole program can be turned into a linear integer program, solved by off-the-shelf mixed-integer linear programming solvers.

In case evidence is present, de Campos and Cozman [54] show how to derive a linear integer program by replicating constraints and auxiliary variables.

The conceptual advantage of the whole method is that it shows how to benefit from the available highly efficient solvers for integer programs. However, it seems that current solver technology is not yet able to beat the other specialized algorithms described previously.

5.2.3. Linearization

The multilinear formulation in Expression (9) attempts to optimize all the local distributions $p(X_i|\pi_i)$ at once, which is usually time-consuming and leads to numerical problems. A simple and effective inner approximation to the problem based on Lukatskii and Shapot [76]'s ideas was proposed by de Campos et al. [38] and Antonucci et al. [58]. The proposal consists in performing a local search on the space of distributions by optimizing all local distributions $p(X_i|\pi_i)$ for a variable X_i at a time, while fixing the values of the remaining distributions. The algorithm is initialized with arbitrary distributions selected inside their corresponding credal set (e.g., the center of mass of the set). Then at each iteration, all distributions but one in Expression (9) remain fixed and the resulting linear program is optimized. The optimal solution replaces the incumbent solution and the process is repeated. The algorithm stops when no further improvement can be made, or when a maximum number of iterations is reached. The method is efficient in low-treewidth networks, as every iteration consists in solving a linear program with a reasonably small number of variables and constraints. Moreover, the constraints of the program can be updated from one iteration to another in an efficient way using methods from inference in Bayesian networks.

Preliminary experiments carried out by Antonucci et al. [58] suggest that linearization of Expression (9) leads to an efficient approximate procedure which provides more accurate results than other approximate methods such as GL2U [62] and ILS [59], which were previously shown to be state-of-the-art.

5.3. Decision-making

Probabilistic models are often used to produce decisions. The standard theory of statistical decision making postulates that a rational agent should act so as to maximize her expected utility [71]. As an example, take 0/1 utility, a commonly used metric in machine learning that assigns utility 1 if the decision matches the true value and 0 otherwise, and consider the selection of a configuration for some subset of categorical variables as a decision. Then, using a single probability distribution, a decision for the agent boils down to selecting the configuration that maximizes the posterior probability, a procedure known as Maximum-A-Posteriori inference or Most Probable Explanation [4].

The situation is less clear when the agent's belief is modeled by a credal set; several criteria have been proposed for optimality of decisions, most of them allowing for indeterminacy (i.e., for prescribing more than one optimal decision). Perhaps the most intuitive criterion is *E-admissibility*: a rational agent should choose an act that maximizes expected utility for at least one distribution in her credal set [19]. The analogue of the Maximum-A-Posteriori procedure is to have the agent select any configuration that maximizes posterior probability with respect to some distribution in the credal set. More formally: a configuration $\mathbf{z}^*$ of a set of random variables $\mathbf{Z}$ is E-admissible given some evidence $\mathbf{Y} = \mathbf{y}$ when there is $\mathbb{P} \in \mathbb{K}(\mathbf{Z}|\mathbf{y})$ such that $\mathbb{P}(\mathbf{z}^*|\mathbf{y}) = \max_{\mathbf{z}} \mathbb{P}(\mathbf{z}|\mathbf{y})$; in symbols, $\mathbf{z}^*$ is E-admissible iff:

$$\exists \mathbb{P} \in \mathbb{K}(\mathbf{Z}|\mathbf{y}) : \Big[\mathbb{P}(\mathbf{Z}=\mathbf{z}^*|\mathbf{Y}=\mathbf{y}) - \max_{\mathbf{z}} \mathbb{P}(\mathbf{Z}=\mathbf{z}|\mathbf{Y}=\mathbf{y}) \Big] \geq 0\,. \tag{10}$$

When the local credal sets are finitely generated, the above criterion can be cast as a maximization; thus $\mathbf{z}^*$ is E-admissible iff:

$$\max_{\mathbb{P} \in \mathbb{K}(\mathbf{Z}|\mathbf{y})} \Big[\mathbb{P}(\mathbf{Z}=\mathbf{z}^*|\mathbf{Y}=\mathbf{y}) - \max_{\mathbf{z}} \mathbb{P}(\mathbf{Z}=\mathbf{z}|\mathbf{Y}=\mathbf{y}) \Big] \geq 0\,. \tag{11}$$

The solution to the above maximization is *not* guaranteed to be attained at the extremes of $\mathbb{K}(\mathbf{Z}|\mathbf{y})$, because of the (nonlinear) inner maximization. The inequality in Expression (11) can however be rewritten as a multilinear program by taking probabilities jointly on $\mathbf{Z}$ and $\mathbf{Y}$ instead of conditioning on $\mathbf{Y}=\mathbf{y}$, and adding a constraint of positivity of the probability of evidence. Alternatively, we can employ the Charnes-Cooper transformation, and we can rewrite the problem as verifying if the objective of the following program is nonnegative:

$$\begin{aligned}
\text{maximize} \quad & \mathbb{P}'(\mathbf{Z}=\mathbf{z}^*, \mathbf{Y}=\mathbf{y}) - \gamma\,, \\
\text{subject to} \quad & \mathbb{P}'(\mathbf{Z}=\mathbf{z}, \mathbf{Y}=\mathbf{y}) \leq \gamma && \text{for each } \mathbf{z}, \\
& \mathbb{P}'(\mathbf{Z}=\mathbf{z}, \mathbf{Y}=\mathbf{y}) = t \cdot \mathbb{P}(\mathbf{Z}=\mathbf{z}, \mathbf{Y}=\mathbf{y}) && \text{for each } \mathbf{z}, \\
& \sum_{\mathbf{z}} \mathbb{P}'(\mathbf{Z}=\mathbf{z}, \mathbf{Y}=\mathbf{y}) = 1 \\
& \mathbb{P} \in \mathbb{K}(\mathbf{Z}, \mathbf{Y})\,,
\end{aligned}$$

where t and γ are fresh continuous variables, and the number of inequalities grows exponentially with the number of configurations of $\mathbf{Z}$. The latter form can in turn be recast as a multilinear program with the number of inequalities bounded by the network treewidth, analogously to the approach presented in Section 5.2.1.

Researchers in imprecise probability have sometimes argued that a better criterion to select decisions is *maximality*, where a configuration is selected when it is *not* assigned a smaller probability value than some other configuration under any distribution in the credal set. That is, a configuration $\mathbf{z}^*$ is maximal when there is no other configuration $\mathbf{z}$ such that $\mathbb{P}(\mathbf{z}|\mathbf{y}) > \mathbb{P}(\mathbf{z}^*|\mathbf{y})$ for every $\mathbb{P} \in \mathbb{K}(\mathbf{Z}|\mathbf{y})$. We have that:

Theorem 5. *Given a separately specified credal network with finitely generated local credal sets, a configuration $\mathbf{z}^*$ is maximal iff:*

$$\max_{\mathbf{z}} \min_{\mathbb{P} \in \operatorname{ext} \mathbb{K}(\mathbf{Z}|\mathbf{y})} \Big[\mathbb{P}(\mathbf{Z}=\mathbf{z}|\mathbf{Y}=\mathbf{y}) - \mathbb{P}(\mathbf{Z}=\mathbf{z}^*|\mathbf{Y}=\mathbf{y}) \Big] \leq 0\,. \tag{12}$$

The optimization above must again deal with multilinear terms that are built from the structure of the input credal network; the problem is thus one of multilinear programming, a problem that can exploit the network structure much in the same way as the multilinear programming and integer-linear programming reformulations for inference.

In some tasks, we are interested in obtaining all maximal configurations of a large number of variables. Running a multilinear program for each possible configuration $\mathbf{z}^*$ may not be feasible as there are exponentially many of them (in the cardinality of $\mathbf{Z}$). Looking at a special case, De Boom et al. [77] developed an algorithm for computing the maximal hidden state sequences of HMM-shaped credal networks conditional on an observation of "manifest" variables.[11] The algorithm is polynomial in both the input and in the number of maximal sequences, so it bypasses the need to look at every configuration. However, the number of maximal sequences is not polynomially bounded by the input; thus the overall worst-case complexity of the algorithm is not, in a strict sense, polynomial. Still, for many models the number of maximal sequences is reasonably small.

De Bock et al. [78] used maximality/E-admissibility to measure the robustness of decision making with a Bayesian network. They considered credal networks obtained as perturbations of a reference Bayesian network, and judged a decision made by the precise model to be robust to a certain level if the corresponding credal network admitted no other maximal/E-admissible configuration (by construction, the decision obtained from the Bayesian network is E-admissible, hence maximal, w.r.t. the credal network). Importantly, their approach scales to decisions involving several variables, with complexity similar to computing inferences in Bayesian networks.

A common alternative to producing set-valued decision is to consider only the maximin and minimax decision rules, which produce a single configuration by either minimizing the worst-case expected loss or maximizing the worst-case expected utility. For example, the maximin criterion selects

$$\arg\max_{\mathbf{z}} \underline{\mathbb{P}}(\mathbf{Z}=\mathbf{z}|\mathbf{Y}=\mathbf{y})\,.$$

[11] An HMM-shaped network is a tree-shaped model in which variables are either hidden or manifest. The manifest variables represent observations and are leaves of the graph. The hidden variables represent unobserved quantities and are arranged as a chain. Each manifest variable has a single hidden variable as parent. Each hidden variable has at most one other hidden variable as parent.

There are also more heuristic criteria aiming at efficient computation that produces set-valued decisions. Interval dominance considers a configuration $\mathbf{z}^*$ optimal when

$$\overline{\mathbb{P}}(\mathbf{Z}=\mathbf{z}^*|\mathbf{Y}=\mathbf{y}) \geq \max_{\mathbf{z}} \underline{\mathbb{P}}(\mathbf{Z}=\mathbf{z}|\mathbf{Y}=\mathbf{y})\,.$$

Optimal configurations under interval dominance can be computed by verifying the lower and upper probabilities of every pair of configurations. This is efficient only if the number of variables in $\mathbf{Z}$ is small.

We have so far limited our discussion to 0/1 utilities; very often one is interested in making decisions involving more general utility functions (e.g., quadratic loss, concave utilities, etc). A well-known result by Cooper [79] on Bayesian networks is that, for random variables with finite domains, the computation of a conditional expected utility $U(\mathbf{Z})$ can be translated to a marginal inference by augmenting the model with a Boolean variable Z whose conditional probability is obtained by a min-max normalization of the utility function:

$$\mathbb{P}(Z=1|\mathbf{Z}=\mathbf{z}) = \frac{U(\mathbf{z}) - \min_{\mathbf{z}'} U(\mathbf{z}')}{\max_{\mathbf{z}'} U(\mathbf{z}') - \min_{\mathbf{z}'} U(\mathbf{z}')}\,.$$

Then $\mathbb{E}[U(\mathbf{Z})|\mathbf{Y}=\mathbf{y}]$ equals $\mathbb{P}(Z=1|\mathbf{Y}=\mathbf{y})$ up to a positive constant (note that the decision criteria discussed are invariant to positive scaling). Since that is a linear transformation, and the decision making criteria that we discussed can be cast as multilinear programs, the properties of E-admissibility and maximality extend also to more general utility functions. Note however that Cooper's transformation adds arcs to the network (from $\mathbf{Z}$ into Z), so that the complexity of inferences may drastically change (Mauá et al. [80] show that for additive utility functions the transformation does not increase complexity [80]).

Even more complex decision scenarios may require sequential decisions. Influence diagrams extend Bayesian networks with decision and utility nodes in order to represent structured sequential decision making problems. Solving an influence diagram means finding a combination of policies, functions that map observations of the parents of a decision node to values of the corresponding variable, so as to maximize the sum of the utilities. The first algorithms for influence diagrams with imprecise probabilities appeared around 1990 [81,82]. More recently, Cabañas et al. [83] developed algorithms for influence diagrams with set-valued probability and utility assessments. The algorithms implement a variable elimination like procedure with the same complexity of algorithms for standard influence diagrams.

A more detailed list of decision making criteria with imprecise probabilistic models has been given by Troffaes [84], but a similarly comprehensive study of decision making with respect to credal networks is still lacking.

6. The complexity of marginal inference and decision-making

Initial results on the complexity of marginal inference appeared in 2003 and 2004 [85,86]: NP-hardness of marginal inference with polytrees, $\mathsf{NP}^{\mathsf{NP}}$-completeness of marginal inference. These results were given with sketchy, sometimes flawed proofs. The first thorough analysis of the complexity of marginal inference and related problems appeared in 2005 [69]. These results have been enlarged and refined in a number of ways since then. In order to survey the results, we first present a quick and intuitive introduction to complexity theory [87].

6.1. A bit of background

Readers who are familiar with computational complexity theory may wish to skip this section.

The main idea in computational complexity theory is to classify problems according to the amount of computational resources it takes to solve them in the worst-case. A *problem* in this context usually corresponds to a set of strings containing binary symbols (say 0 and 1); the idea is that one takes a string as input, and the challenge is to determine whether this input actually belongs to the set of strings of interest. Thus our task is to read a string as input and to *accept* the string, or to *reject* it. If we can do this for a problem, then the problem is *solved*. This sort of task can be encoded in a Turing machine, a particularly simple abstraction that captures the essence of computing machinery. A Turing machine reads the input written into a tape; then it operates on the tape using its own internal states and transitions; and then it either stops by accepting or rejecting the input, or it gets into an infinite loop of transitions. A sequence of transitions in a Turing machine, from initial state on, is a *computation path*.

Suppose then that we have a problem and we can determine whether any given string belongs or not to the problem by using a Turing machine with deterministic transitions, using a number of transitions that is a polynomial on the size of the input string. Then the complexity of the problem is said to be *polynomial*; the class containing all these problems is denoted by P. A different kind of transition is a *nondeterministic* one: here the machine can jump from one state to a set of other states in parallel. Note that "nondeterministic" does not mean "uncertain" in this setting; it rather means that one cannot pin down a single state for the machine, as it is simultaneously running through a set of possibilities. A machine with nondeterministic transitions accepts a string if any of its computation paths accepts the string; otherwise it rejects it. The set of problems that can be solved using Turing machines with a polynomial number of nondeterministic transitions (in a computation path) in the size of the input string forms the class NP. It seems that a problem in NP can be a lot more

complicated than a problem in P, for the former can be solved by pursuing many paths at once; however one of the open, and most difficult, questions in computational complexity theory is whether $\mathsf{P} = \mathsf{NP}$.

Yet another transition is a *probabilistic* one: from one state the machine moves to another state selected with uniform probability amongst a set of possible states. Now consider solving a problem with a machine containing probabilistic transitions. It may be the case that for a given input the machine accepts or rejects it with probability one. But in general each input string will be accepted with some probability. And there will be a probability that the machine makes a mistake (that is, it accepts when it should not, or it does not accept when it should do so). The complexity class PP consists of those problems that can be solved by a Turing machine using a number of probabilistic transitions that is a polynomial on the size of the input, such that the probability of error is smaller than half for each input string. This is a difficult and convoluted definition. Note that to decide whether a problem is in PP or not it does not suffice to run a machine once for each possible input. What matters is the probability that the machine makes a mistake, for each possible input. So we must rather look at the prospective solving machine "from the outside", computing its probabilities so as to declare our problem to be in PP or not.

Here is an alternative way to understand the complexity class PP. A problem is in PP if and only if we can find a Turing machine such that, for each input that must be accepted, more than half of the computation paths end up accepting, and these computation paths (accepting or rejecting ones) contain a number of nondeterministic transitions that is polynomial on the size of the input. That is, we find a *nondeterministic polynomial-time* Turing machine and we again look at it "from the outside", not worrying whether a particular computation path accepts the input of interest, but rather by counting the computation paths that accept the input, and checking whether this number is larger than half of the total number of computation paths. Clearly PP has a different nature from NP, as it is related to counting solutions, rather than finding one solution.

For any given complexity class C, we say that a problem P is C-hard if any problem in C can be solved first by transforming the string input s with polynomially many steps into another string s', and then feeding s' to P (this process is often called a *many-one reduction*). A problem P is C-complete if and only if it is in C and it is C-hard. Intuitively, if a problem is C-complete, then it requires the full resources that are used to define C, because one can use the solution of the problem to solve any other problem in C.

An important result about PP is that deciding if a marginal inference exceeds a given threshold with Bayesian networks is PP-complete [88]. Intuitively, this means that "probabilistic reasoning means counting". Indeed, to compute the probability of a solution one must examine the set of possible solutions, rather than finding a single solution — hence one might conjecture that PP is "harder" than NP". While it is known that $\mathsf{NP} \subseteq \mathsf{PP}$, it is open whether $\mathsf{NP} = \mathsf{PP}$.

To state results about credal networks we need one more piece of complexity theory. An *oracle Turing machine* $M^{\mathcal{P}}$, where $\mathcal{P}$ is a problem (a set of strings), is a Turing machine that can at any time request an immediate solution to an instance of $\mathcal{P}$. If a class of problems C is defined by a set of Turing machines $\mathcal{M}$ (that is, the problems are solved by these machines) then $C^{\mathcal{P}}$ denotes the set of problems that are solved by $\{M^{\mathcal{P}} : M \in \mathcal{M}\}$; if B is another class of problems, then C^B is defined to be $\bigcup_{\mathcal{P} \in B} C^{\mathcal{P}}$. For instance, the class $\mathsf{NP}^{\mathsf{PP}}$ contains problems that can be solved by running a search procedure in parallel (the nondeterministic part) such that at any time the solution to a counting problem can be used (the probabilistic part).

One final note about complexity theory. The computational problems we entertain here, for example computing marginal inference with a given credal network, are of a functional form, meaning that they produce a number and not simply a decision. While such problems can be captured in complexity theory under the so-called functional classes (e.g. FP or #P), doing this introduces subtleties that are out of the scope of this work (e.g. approximating real values) [89]. These two classes of problems however are closely related. For example, we can compute a functional version of a problem (e.g. computing marginal inference) by running the bisection method where each iteration solves the decision version (e.g. whether marginal inference exceeds a threshold). However, a celebrated theorem by Toda [90] states that any problem that can be solved by a NP or PP machine with an arbitrary finite number of PP or NP oracles "stacked" can be solved with polynomially many calls to a PP machine. Thus, allowing polynomially many calls to a problem may blur the distinction between complexity classes. Yet, it is our understanding that studying the decision version of the inference problems provides a fair picture of the complexities of the respective functional problems without delving into more complicated issues. Thus in the following we consider only the decision version of problems.

6.2. Complexity results

To start, suppose we get a credal network, a query $Z = z$, evidence $\mathbf{Y} = \mathbf{y}$ and a rational value γ as input, and consider the following problem: Decide whether the upper probability of $Z = z$ given $\mathbf{Y} = \mathbf{y}$ is larger than γ. The basic result is that, if we impose no restriction on the input credal network, this problem is $\mathsf{NP}^{\mathsf{PP}}$-complete [86]. Intuitively: marginal inference with strong extensions consists of searching the "best" extreme point formed by extreme points of the local conditional credal sets, where each extreme point is ranked by its probability, computed using multiple calls to a problem in PP. The fact that marginal inference can be solved this way is clear: just select extreme points, thus obtaining a Bayesian network, and run inference on the latter. The proof of $\mathsf{NP}^{\mathsf{PP}}$-hardness is more involved; the most direct way is to show that any instance of E-MAJSAT, a complete problem for $\mathsf{NP}^{\mathsf{PP}}$, can be turned into a marginal inference for a strong extension [69].

The natural question is then whether inference gets less complex when we impose restrictions on the input credal networks. Trivially yes: if every credal set is a singleton, we return to the PP-completeness of deciding marginal inference with Bayesian networks. And if every credal set is a singleton and the DAG is a polytree, or even if the DAG merely has a bound on treewidth, then deciding marginal inference is in P. But we also know the 2U algorithm takes polynomial time to compute an inference for a credal network over binary variables whose DAG is a polytree [16]. Can we remove the restriction of binary variables and stay within P? No: when the DAG is already a tree, marginal inference is NP-hard when variables have more than two values [91,92]! And deciding marginal inference is in NP when the DAG is a polytree (irrespective of the number of values any variable can assume, provided that this number is larger than one) [69].

When the input credal network is assumed to have bounded treewidth, then deciding marginal inference is still NP-complete [69]. A more impressive result is this: as mentioned in Section 5, if treewidth is bounded, and the number of values of variables is also bounded, then there is fully-polynomial time approximation scheme that computes marginal inference with error within some ϵ in time that is polynomial on the size of the input and on $1/\epsilon$ [91].

Mauá et al. [91] provided another tractable case by showing that marginal inference in HMM-shaped networks is polynomial-time computable if the query variable is the "last" hidden node (this type of marginal inference is called filtering in the time-series statistical analysis literature). An alternative polynomial-time algorithm for the same problem was later developed by Mauá et al. [93], along with efficient algorithms for other types of inference in HMM-shaped networks.

Regarding decision making, Antonucci and de Campos [94] showed that both deciding if a given value of a single variable is E-admissible and deciding if a given value is maximal can be used to solve any marginal inference problem; hence verifying E-admissibility and maximality is as hard as marginal inference. As discussed in Section 5.3, when deciding for the values of a single variable, maximality can be efficiently reduced to marginal inference. Hence, the complexity of the former and the latter coincide.

An interesting exception in this sea of intractability is that of Forest-Augmented Naive Credal Classifiers, which consist of a credal network where there is a distinguished variable Z such that every other node has Z and at most one other variable as parent (i.e., the network obtained by removing Z is a forest). In this case, Zaffalon and Fagiuoli [95] developed a polynomial-time algorithm for computing the maximal configurations of Z given evidence on all the remaining variables.

The complexity of decision making involving multiple variables has been much less studied. de Campos and Cozman [69] showed that computing maximin configurations is $\mathsf{NP}^{\mathsf{NP}}$-hard in polytree-shaped and bounded-treewidth credal networks. This complexity is reduced if every variable is either a decision variable (i.e., one of the variables whose value we want to decide) or part of evidence; in this case, complexity of maximin is PP-hard. Kwisthout [96] showed that deciding maximinimality is $\mathsf{NP}^{\mathsf{NP}^{\mathsf{PP}}}$-complete, climbing several steps in the counting hierarchy.

7. Eliciting, learning, and applying credal networks

A Bayesian network consists of two parts: a qualitative one that is encoded by the underlying directed acyclic graph, referred to as the *structure* of the network; and a quantitative one that is translated into conditional probabilities. Similarly, a separately specified credal network consists of a directed acyclic graph, its structure, and a set of local credal sets. The structure carries information about dependences and independences amongst relevant variables; the guiding principle is the Markov condition that connects the graph with assumed independence relations.

While the elicitation of a Bayesian network must necessarily interpret independence relations in a single way (stochastic independence), with credal networks one may choose among strong independence, epistemic irrelevance, and so on. In this paper we have focused on strong independence, so we assume here the Markov condition with respect to strong independence. Thus the elicitation of the structure is similar in principle to the elicitation of the structure for a Bayesian network. On the other hand, one has much more flexibility in eliciting local credal sets than in eliciting conditional distributions. For instance, one may work with assessments such as "the probability of victory is twice as large as the probability of defeat" instead of requiring a precise number as in "the probability of victory is exactly 0.85". Indeed by allowing probabilities to be imprecise one greatly facilitates the elicitation of knowledge-based systems; by demanding less from experts, one can build models whose recommendations have higher credibility [97]. The enormous variety of elicitation strategies that one can use, both in eliciting structure and local credal sets, has been discussed at length in a previous excellent tutorial [7]. Given that published material, we do not dwell further on the topic.

As an example of elicitation-based decision support system, Salvetti et al. [98] used credal networks to identify debris flow source areas. Debris flows are destructive natural hazards, which might lead to human losses, especially in mountain regions. Also, Antonucci et al. [99] designed a credal network to assess the operational risk of banks, which sets the capital requirements needed and has important implications. And Antonucci et al. [100,101] used expert knowledge to build credal networks that supported critical decision making for military purposes, where data is usually scarce.

Several other applications of credal networks can be found in the literature, particularly during the last ten years or so; many of them resort to data analysis and statistical inference to build credal networks, particularly for classification tasks. Before we visit some notable recent applications, we must comment on a few general techniques to learn credal networks from data.

There are in fact two tasks: one is to learn the structure; the other is to learn the local credal sets. We start with the latter. Clearly the estimation of local credal sets is no trivial matter; the most common strategy is to place sets of prior distributions over parameters and to use Bayes rule over them. One particularly popular set of distributions that is used in

this setting is the Imprecise Dirichlet Model (IDM); as suggested by the name, the idea there is to specify a set of Dirichlet distributions [102]. Other strategies, such as NonParametric Inference (NPI), also lead to credal sets when data is collected [103]; even frequentist and imputation methods can be adapted to handle robustness and missing data concerns [102,104]. The literature is extensive on these matters, but they are general issues regarding credal sets and statistical inference, not challenges that depend on properties of credal networks.

Building the structure of a credal network from data using methods that take into account the possible imprecision in probability values is a challenging topic. Should we stay with a single graph? Or should we look into "imprecise" structures — perhaps a set of related graphs? The literature has not converged to a consensus on these matters, with few representative proposals. One notable effort is due to Masegosa and Moral [105], where the authors consider a variety of approaches. They start with the Imprecise Dirichlet Model so as to learn credal sets from data. First they identify drawbacks of the lmprecise Dirichlet Model and propose suitable fixes. They then consider various ways to build the structure of a credal network, contemplating methods that build *sets* of credal networks, and also methods that produce extensively (not separately) specified credal networks. Recently an even more general framework has been proposed by Moral [106], where several paradigms used to learn credal sets are combined and applied to credal networks.

Apart from these general learning methods, there are several techniques that aim at specific types of credal networks. As already noted, most of them focus on classification tasks. Indeed, a typical applied theme for credal networks has been the construction of reliable classifiers, that is, systems that issue a prediction (an assignment of an object to a class) only if enough statistical evidence is available. Work on *credal classification* started in 2001 [107] with the *Naive Credal Classifier (NCC)*, a version of the popular Naive Bayes classifier that accommodates imprecision in probability values. The idea is that a NCC is learned by collecting training data and computing posterior probabilities from a family of prior distributions; the NCC resorts to the Imprecise Dirichlet Model (IDM) [70] to encode sets of prior distributions. The IDM can capture a set of Dirichlet distributions and leads to closed-form expressions for decisions of interval dominance in the NCC. The NCC has since been extended in a variety of directions, for instance by allowing missing data to be treated with conservative updating [108], as well as model averaging [109] and tree-like structures [95].

We refrain from providing technical details on credal classifiers, and indeed from further surveying the many publications on this topic, because an excellent review of the literature has appeared in 2014 [110]. Since then there has been significant extensions and applications [111–113]. In the following paragraphs we describe a few applications of credal classification, from its initial years until rather recent work.

Zaffalon [114] employed credal classifiers to reliably predict grass grub quantities based on paddock characteristics and farming practices. Grass grubs are one of the major insect pests in pastures in New Zealand; data on grass grub occurrence and related events is however scarce. Corani et al. [115] used credal classifiers to perform robust texture recognition based on images. The classifiers were able to suspend classification on degraded images, on which standard classifiers usually perform poorly. Missing data is another source of unreliability in classification. de Campos and Ji [116] and Antonucci et al. [117] modeled the missing process as credal sets and employed the resulting models in action recognition tasks, that is, the task of predicting which action is being carried out from a video sequence. Notably, these procedures obtain credal networks even when the underlying model is a Bayesian network.

Soullard et al. [118] used HMM-shaped credal networks to perform technical gesture recognition in workers of an aluminium foundry. They argue that class imbalance hampers the classification accuracy, and that, due to scarcity of data, class distributions obtained from data are unreliable. They use the Imprecise Dirichlet Model to learn a credal network, which is used to compute the maximal classes. They report an increase in predictive accuracy with respect to the precise classifier when a single maximal class is obtained, which occurs in more than 90% of instances.

Antonucci et al. [119] tackled the problem of multivariate time series classification by comparing the limiting upper and lower bounds of the probability of observations of different HMM-shaped credal networks learned for each class. They evaluate their approach on benchmark datasets of action recognition and gesture recognition (e.g., sign language recognition). They showed that their approach finds hard-to-classify instances more accurately than previous approaches based on credal HMMs, and are competitive with standard methods such as Dynamic Time Warping.

The classification examples we have discussed consider that each instance is assigned exactly one class label. In some tasks such as annotating multimedia content (photos, videos, texts), an object can be assigned many or even no labels. This is known as multilabel classification. Antonucci and Corani [111] extended the multilabel classifier based on Bayesian networks of Antonucci et al. [120] to accommodate imprecision in the model parameters. The resulting model, named Multilabel Naive Credal Classifier (MNCC), was evaluated on a collection of benchmark datasets for multilabel classification. The results showed that MNCC was able to determine with high accuracy instances where the precise counterpart performed poorly, thus supporting reliable predictions.

Mauá et al. [93] shows applications for HMM-like credal networks such as reliable classification of phonemes of Japanese, human action recognition, text completion for smart TV user interfaces, and part-of-speech tagging. In all applications they show that a credal classification achieves better results in terms of discounted accuracy [121] and custom losses, and is able to discriminate between hard-to-classify and easy-to-classify instances.

8. Conclusions

After some thirty years of development, strong extensions lie at the center of a relatively large web of results and algorithms. One characteristic of strong extensions of separately specified credal networks (with closed convex local sets) that is worth emphasizing is that they mimic many properties of Bayesian networks, given that lower and upper bounds on probabilities are attained at their extreme points — and each extreme point is basically a Bayesian network. For instance, the key property of d-separation is preserved. Another characteristic of strong extensions that is worth mentioning is that they are closely tied to convexity; indeed strong independence is just complete independence spiced with convexity. Thus strong extensions are quite appealing within the context of theories of imprecise and indeterminate probabilities that rely on convex (and closed) credal sets. However it is fair to ask whether the conceptual benefits of convexity pay off. On the one hand, most algorithms and complexity results presented in this paper are not affected in any essential way if strong independence is replaced by complete independence. Some calculations, for example related to decision making or to entropy and mutual information (topics we have not stressed here as there is relatively little to report on them) may actually be affected by whether credal sets are convex or not. Still, strong extensions live on offering the most popular semantics for credal networks, one that is adopted in almost all the literature.

In this review we have concentrated on the main ideas behind algorithms and complexity results. The reader can benefit from previous gentler tutorials on credal networks [6,7], and from surveys that have covered historical and computational aspects of credal networks up to 2005 [5,18]. Certainly we could not do justice to many aspects of credal networks that should deserve more space. We did not present in detail several algorithms that are quite ingenious but that work within very special conditions. We have also downplayed several algorithms that contain clever and potentially useful ideas but that have been surpassed by more recent schemes. Also, we have tried to paint an intuitive picture of complexity results, as their proofs can be very technical and dry. Finally, we have summarized only the main ideas behind elicitation, learning and classification; there are other tutorials that cover several issues there, and a comprehensive picture is yet to emerge.

From a computational point of view, we now have the basic infrastructure for strong extensions in place: we know the complexity of marginal inference for a variety of cases, and we have algorithms that can handle practical problems. However, there is quite some room for future exploration. There are many criteria for decision-making, and few have received adequate investigation; their complexity is mostly unknown, and few algorithms can be run. And marginal inference offers still plenty of challenges before large credal networks can be applied in practice. Finally, we note that there has been little work on specification languages that specify credal networks in flexible ways.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

We are very grateful to Jasper De Bock for his many contributions to this paper; in particular for sending us many corrections and suggestions that really improved the text. We are also grateful to the anonymous referees, for their very detailed reviews and insightful comments.

The first author received financial support by the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), grants 304012/2019-0 (PQ) and 420669/2016-7 (Universal). The second author is partially supported by the CNPq grant 312180/2018-7 (PQ). This work has been supported in part by the Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), grants 2015/21880-4, 2019/07665-4, and 2016/18841-0, and in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) - finance code 001.

Appendix A. Proofs

In this section we present proofs for the main results in the text. To facilitate reading, we replicate the statements.

Theorem 1. *The convex hull of the complete extension of a credal network is the strong extension of the credal network: it is the largest credal set that satisfies the Markov condition with respect to strong independence. This is also true when we replace each credal set* $\mathbb{K}(X_i|Pa(X_i) = \pi_i)$ *in Expression (4) by the set of its extreme points.*

Proof. To show that $\text{co}\,\mathbb{K}_C(X_1, \ldots, X_n)$ is indeed the strong extension of the credal network, reason by finite induction. Assume that $X_1, \ldots, X_n$ are topologically sorted, and associate with each variable X_i a subgraph G_i consisting of $X_1, \ldots, X_i$ and the edges amongst those variables. Clearly the strong extension for G_1 is $\mathbb{K}_C(X_1) = \mathbb{K}(X_1) = \text{co}\,\text{ext}\,\mathbb{K}(X_1)$. Consider a subgraph G_i and the associated local credal sets; assume their strong extension is $\text{co}\,\mathbb{K}_C(X_1, \ldots, X_i)$. Now consider moving to the subgraph G_{i+1}, that is, consider adding the variable X_{i+1}. Note that the marginal credal set of $\mathbb{K}_C(X_1, \ldots, X_{i+1})$ with respect to $X_1, \ldots, X_i$ is $\mathbb{K}_C(X_1, \ldots, X_i)$, as

$$\sum_{X_{i+1}} \prod_{j=1}^{i+1} \mathbb{P}(X_j|\text{Pa}(X_j)) = \prod_{j=1}^{j} \mathbb{P}(X_j|\text{Pa}(X_j))\,,$$

for any choice of $\mathbb{P}(X_j|\text{Pa}(X_j)) \in \mathbb{K}(X_j|\text{Pa}(X_j))$. Now consider extending $\text{co}\,\mathbb{K}_C(X_1,\ldots,X_i)$ to the largest set that satisfies the Markov condition "X_{i+1} is strongly independent from its nondescendant nonparents given $\text{Pa}(X_{i+1})$" and the local constraints $\mathbb{P}(X_{i+1}|\text{Pa}(X_{i+1})) \in \mathbb{K}(X_{i+1}|\text{Pa}(X_{i+1}))$. That means that the extreme distributions of that (closed and convex) largest set must factorize as

$$\mathbb{P}(X_1,\ldots,X_i)\mathbb{P}(X_{i+1}|\text{Pa}(X_{i+1}))\,,$$

where $\mathbb{P}(X_1,\ldots,X_i)$ is an extreme distribution of $\text{co}\,\mathbb{K}_C(X_1,\ldots,X_i)$ and $\mathbb{P}(X_{i+1}|\text{Pa}(X_{i+1})) \in \mathbb{K}(X_{i+1}|\text{Pa}(X_{i+1}))$. The convex hull of these distributions is exactly $\text{co}\,\mathbb{K}_C(X_1,\ldots,X_{i+1})$. Hence the strong extension of the whole credal network is the credal set $\text{co}\,\mathbb{K}_C(X_1,\ldots,X_n)$.

Now we argue that each local credal set $\mathbb{K}(X_i|\text{Pa}(X_i)=\pi_i)$ can be replaced by $\text{ext}\,\mathbb{K}(X_i|\text{Pa}(X_i)=\pi_i)$ in Expression (4). Note first that each element of each $\mathbb{K}(X_i|\text{Pa}(X_i)=\pi_i)$ can be written as a convex combination $\sum_{j_i=1}^{m_i} \alpha_{j_i}\mathbb{P}_{j_i}(X_i|\text{Pa}(X_i)=\pi_i)$ where $\alpha_{j_i} \geq 0$, $\sum_{j_i}\alpha_{j_i}=1$, and $\mathbb{P}_{j_i}(X_i|\text{Pa}(X_i)=\pi_i) \in \text{ext}\,\mathbb{K}(X_i|\text{Pa}(X_i)=\pi_i)$. Now suppose we build a joint distribution $\mathbb{P}(X_1,\ldots,X_n)$ in the product form of Expression (4), where each term is written as a convex combination. Then:

$$\begin{aligned}\mathbb{P}(X_1=x_1,\ldots,X_n=x_n) &= \prod_i \left(\sum_{j_i=1}^{m_i} \alpha_{j_i}\mathbb{P}_{j_i}(X_i=x_i|\text{Pa}(X_i)=\pi_i)\right)\\ &= \sum_{j_1=1}^{m_1}\cdots\sum_{j_n=1}^{m_n}\left(\prod_i \alpha_{j_i}\mathbb{P}_{j_i}(X_i=x_i|\text{Pa}(X_i)=\pi_i)\right).\end{aligned}$$

We have that $\alpha_j = \prod_i \alpha_{j_i} \geq 0$ and

$$\sum_{j=1}^{m_1\cdots m_n}\alpha_j = \sum_{j_1=1}^{m_1}\cdots\sum_{j_n=1}^{m_n}\prod_i \alpha_{j_i} = \prod_i\left(\sum_{j_i=1}^{m_i}\alpha_{j_i}\right) = 1\,.$$

Hence any joint distribution $\mathbb{P}(X_1,\ldots,X_n)$ produced this way with at least a nontrivial convex combination (i.e., with $m_i > 1$ and all $\alpha_{j_i} > 0$) is itself a nontrivial convex combination of joint distributions built as the product of extreme points of local credal sets. But any such convex combination is already generated in the convex hull of the products involving only extreme points of local credal sets; hence these latter sets of "local" extreme points are all we need to consider. □

Theorem 2. *Any combination of extreme points from the local credal sets $\mathbb{K}(X|Pa(X))$ is an extreme point of the strong extension.*

Proof. If the strong extension has a single point (i.e., it is a Bayesian network), then the result follows trivially. Otherwise, reason by contradiction as follows. Take $\mathbb{P}$ that is the product of extreme points of local credal sets (as specified by Theorem 1) and suppose that $\mathbb{P}$ is a nontrivial convex combination $\sum_i \alpha_i\mathbb{P}_i$ where each $\mathbb{P}_i$ is a distinct extreme point of the strong extension, $\alpha_i > 0$ for all i, and $\sum_i \alpha_i = 1$. Suppose the variables $X_1,\ldots,X_n$ are topologically ordered, so that X_1 has no parents and any other X_j has parents in $\{X_1,\ldots,X_{j-1}\}$, if any.

Consider the following property R_j: either the marginal of $\mathbb{P}$ with respect to a nonempty subset of $X_1,\ldots,X_j$ contradicts the hypothesis that $\mathbb{P}$ is a nontrivial convex combination of extreme points of the strong extension, or the marginals of $\mathbb{P}_i$ with respect to $X_1,\ldots,X_j$ are identical for all i, that is, $\mathbb{P}_i(X_1,\ldots,X_j) = \mathbb{P}(X_1,\ldots,X_j)$.

We start by showing that R_1 holds. Suppose there are two marginals $\mathbb{P}_{i_1}(X_1)$ and $\mathbb{P}_{i_2}(X_1)$ that are different; then the marginal of $\mathbb{P}$ with respect to X_1 is a nontrivial convex combination of distinct points of $\mathbb{K}(X_1)$, contradicting the fact that $\mathbb{P}$ was built by multiplying extreme points of local credal sets. Thus we either find that marginals $\mathbb{P}_i(X_1) = \mathbb{P}(X_1)$ for all i, or we reach a contradiction (and the proof is finished).

Now take some $j > 1$ and assume that R_{j-1} holds. If property R_{j-1} holds because a contradiction is produced, then property R_k holds for all $k \geq j$ as well (and in fact the proof of the theorem is finished). Thus suppose that $\mathbb{P}_i(X_1,\ldots,X_{j-1}) = \mathbb{P}(X_1,\ldots,X_{j-1})$ for all i. Consider the case where X_j has a nonempty set of parents $\text{Pa}(X_j)$. Hence, the marginals of all $\mathbb{P}_i$ with respect to $\text{Pa}(X_j) \subseteq \{X_1,\ldots,X_{j-1}\}$ are identical, that is, $\mathbb{P}(\text{Pa}(X_j)) = \mathbb{P}_i(\text{Pa}(X_j))$. Select a configuration π of $\text{Pa}(X_j)$ such that $\mathbb{P}(\text{Pa}(X_j)=\pi) > 0$. It may happen that all $\mathbb{P}_i(X_j|\text{Pa}(X_j)=\pi)$ for every such π are identical; then it follows from the Markov property that the marginals of $\mathbb{P}_i$ with respect to $X_1,\ldots,X_j$ are all identical, so property R_j holds as well. If however there are $\mathbb{P}_{i_1}(X_j|\text{Pa}(X_j)=\pi) \neq \mathbb{P}_{i_2}(X_j|\text{Pa}(X_j)=\pi)$, then

$$\begin{aligned}\mathbb{P}(X_j|\text{Pa}(X_j)=\pi) &= \sum_i \alpha_i \frac{\mathbb{P}_i(\text{Pa}(X_j)=\pi)}{\sum_{i'}\alpha_{i'}\mathbb{P}_{i'}(\text{Pa}(X_j)=\pi)}\mathbb{P}_i(X_j|\text{Pa}(X_j)=\pi)\\ &= \sum_i \alpha_i\mathbb{P}_i(X_j|\text{Pa}(X_j)=\pi),\end{aligned}$$

where the last equality follows from the fact that $\mathbb{P}_i(\text{Pa}(X_j) = \pi)$ are all identical. Hence we reach a contradiction with the fact that $\mathbb{P}$ is the product of extreme points of local credal sets, consequently showing that R_j holds. If X_j has no parents, then simply repeat the argument above with $\mathbb{P}_i(X_j|\text{Pa}(X_j) = \pi) = \mathbb{P}_i(X_j)$ and $\mathbb{P}_i(\text{Pa}(X_j) = \pi) = 1$ to conclude that R_j must hold.

To conclude, as we reach R_n we find that there is a contradiction to the fact that $\mathbb{P}$ is a nontrivial convex combination of distinct extreme points of the strong extension. □

Theorem 3. *The lower/upper probability in Expression (5) are obtained, respectively, by*

$$\inf / \sup \ \frac{\sum_{\mathbf{x}'} \prod_{i=1}^{n} \mathbb{P}(X_i = x_i | Pa(X_i) = \pi)}{\sum_z \sum_{\mathbf{x}'} \prod_{i=1}^{n} \mathbb{P}(X_i = x_i | Pa(X_i) = \pi)},$$

where the sum in the numerator and the inner sum in the denominator are over the values of the set of marginalized *variables* $\mathbf{X}' = \mathbf{X} \setminus (\{Z\} \cup \mathbf{Y})$*, the outer sum in the denominator is over the values of the query variable* Z*, and the optimization is over the distributions from the complete extension such that* $\mathbb{P}(\mathbf{Y} = \mathbf{y}) > 0$*.*

Proof. Any $\mathbb{P} \in \mathbb{K}(Z|\mathbf{Y} = \mathbf{y})$ is produced by taking $\mathbb{P} = \sum_i \alpha_i \mathbb{P}_i$ for $\alpha_i \in [0, 1]$ and $\mathbb{P}_i \in \mathbb{K}_C(X_1, \ldots, X_n)$, marginalizing out $\mathbf{X}'$ and conditioning on $\mathbf{Y} = \mathbf{y}$. Thus we have that $\mathbb{P}(z|\mathbf{y}) = \sum_{i:\beta_i>0} \beta_i \mathbb{P}_i(z|\mathbf{y})$, where $\beta_i = \alpha_i \mathbb{P}_i(\mathbf{y}) / (\sum_j \alpha_j \mathbb{P}_j(\mathbf{y}))$. Note that due to regular conditioning there must be $\beta_i > 0$. Since any convex combination of real numbers lies between its extrema, we have that $\mathbb{P}(z|\mathbf{y}) \in [\min_{i:\beta_i>0} \mathbb{P}_i(z|\mathbf{y}), \max_{i:\beta_i>0} \mathbb{P}_i(z|\mathbf{y})]$. Therefore the optimization reaches the same result if we discard every $\mathbb{P}$ with $\alpha_i \in (0, 1)$ for some i. □

Theorem 4. *Suppose a credal network is separately specified with closed convex local credal sets, where each local credal set has finitely many extreme points. Then the solution to Equation (6) is attained at some extreme point* $\mathbb{P}$ *of the strong extension such that* $\mathbb{P}(\mathbf{Y} = \mathbf{y}) > 0$*; hence it is attained at some combination of extreme points of the local credal sets* $\mathbb{K}(X|Pa(X) = \pi)$*.*

Proof. To simplify the notation, denote by $\mathbb{K}_S$ the credal set $\text{co}\,\mathbb{K}_C(\mathbf{X})$, by $\mathbb{K}_S^0$ the subset of $\mathbb{K}_S$ containing distributions such that $\mathbb{P}(\mathbf{Y} = \mathbf{y}) = 0$, and by $\mathbb{K}_C^+$ the set of extreme points of $\mathbb{K}_S$ such that $\mathbb{P}(\mathbf{Y} = \mathbf{y}) > 0$. Distributions in $\mathbb{K}_S^0$ are not to be taken into account in the optimization. Any distribution $\mathbb{P}$ in $\mathbb{K}_S \setminus \{\mathbb{K}_S^0 \cup \text{co}\,\mathbb{K}_C^+\}$ is the convex combination of a distribution $\mathbb{P}' \in \text{co}\,\mathbb{K}_C^+$ and a distribution $\mathbb{P}'' \in \mathbb{K}_S^0$. For $\mathbb{P}$ must be a convex combination $\sum_i \alpha_i \mathbb{P}_i$ of extreme points of $\mathbb{K}_S$, then $\mathbb{P}' = \sum_{i:\mathbb{P}_i(\mathbf{Y}=\mathbf{y})>0} (\alpha_i/\alpha) \mathbb{P}_i$ and $\mathbb{P}'' = \sum_{i:\mathbb{P}_i(\mathbf{Y}=\mathbf{y})=0} (\alpha_i/(1-\alpha)) \mathbb{P}_i$ for $\alpha = \sum_{i:\mathbb{P}_i(\mathbf{Y}=\mathbf{y})>0} \alpha_i$. We cannot have $\alpha = 1$ for otherwise $\mathbb{P} \in \text{co}\,\mathbb{K}_C^+$ and we cannot have $\alpha = 0$ for otherwise $\mathbb{P} \in \mathbb{K}_S^0$. Now for any distribution $\mathbb{P} = \alpha\mathbb{P}' + (1-\alpha)\mathbb{P}''$ such that $\mathbb{P}' \in \text{co}\,\mathbb{K}_C^+$ and $\mathbb{P}'' \in \mathbb{K}_S^0$, with $\alpha \in (0, 1)$, we have

$$\begin{aligned}
\mathbb{P}(Z = z|\mathbf{Y} = \mathbf{y}) &= \frac{\alpha\mathbb{P}'(Z = z, \mathbf{Y} = \mathbf{y}) + (1-\alpha)\mathbb{P}''(Z = z, \mathbf{Y} = \mathbf{y})}{\alpha\mathbb{P}'(\mathbf{Y} = \mathbf{y}) + (1-\alpha)\mathbb{P}''(\mathbf{Y} = \mathbf{y})} \\
&= \frac{\alpha\mathbb{P}'(Z = z, \mathbf{Y} = \mathbf{y})}{\alpha\mathbb{P}'(\mathbf{Y} = \mathbf{y})} \\
&= \mathbb{P}'(Z = z|\mathbf{Y} = \mathbf{y}).
\end{aligned}$$

Thus we do not need to look at any distribution in $\mathbb{K}_S \setminus \{\mathbb{K}_S^0 \cup \text{co}\,\mathbb{K}_C^+\}$; hence we can restrict ourselves to $\text{co}\,\mathbb{K}_C^+$ in the optimization (and the infimum/supremum is indeed a minimum/maximum over this closed set). In fact we can produce $\mathbb{K}(Z|\mathbf{Y} = \mathbf{y})$ by taking the finitely many points in $\mathbb{K}_C^+$, conditioning them, and taking the convex hull of the resulting points. This shows that $\mathbb{K}(Z|\mathbf{Y} = \mathbf{y})$ is itself a finitely generated credal set.

Finally, there is no need to look at elements of $\text{co}\,\mathbb{K}_C^+$ that are nontrivial (that is, $0 \neq \alpha \neq 1$) convex combinations of distributions in $\text{co}\,\mathbb{K}_C^+$: suppose we had the minimum/maximum at some distribution $\mathbb{P} \in \text{co}\,\mathbb{K}_C^+$ that is a convex combination $\alpha\mathbb{P}' + (1-\alpha)\mathbb{P}''$ for some $\alpha \in (0, 1)$ and for $\mathbb{P}'$ and $\mathbb{P}''$ elements of $\text{co}\,\mathbb{K}_C^+$. Then we would have $\mathbb{P}(Z = z|\mathbf{Y} = \mathbf{y}) = \beta\mathbb{P}'(Z = z|\mathbf{Y} = \mathbf{y}) + (1-\beta)\mathbb{P}''(Z = z|\mathbf{Y} = \mathbf{y})$ where $\beta = \alpha k_1/(\alpha k_1 + (1-\alpha)k_2)$ for $k_1 = \mathbb{P}'(Z = z|\mathbf{Y} = \mathbf{y})$ and $k_2 = \mathbb{P}''(Z = z|\mathbf{Y} = \mathbf{y})$, and it would be at least as good to have $\mathbb{P}'$ or $\mathbb{P}''$ as $\mathbb{P}$. Thus we can examine only $\mathbb{K}_C^+$ in the optimization. □

Theorem 5. *Given a separately specified credal network with finitely generated local credal sets, a configuration* $\mathbf{z}^*$ *is maximal iff:*

$$\max_{\mathbf{z}} \min_{\mathbb{P} \in \text{ext}\,\mathbb{K}(\mathbf{Z}|\mathbf{y})} \left[\mathbb{P}(\mathbf{Z} = \mathbf{z}|\mathbf{Y} = \mathbf{y}) - \mathbb{P}(\mathbf{Z} = \mathbf{z}^*|\mathbf{Y} = \mathbf{y}) \right] \leq 0.$$

Proof. As in the proof of Theorem 4, we use $\mathbb{K}_S$, $\mathbb{K}_S^0$ and $\mathbb{K}_C^+$ with the same meaning. Recall that any distribution $\mathbb{P}$ in $\mathbb{K}_S \setminus \{\mathbb{K}_S^0 \cup \text{co}\,\mathbb{K}_C^+\}$ is the convex combination $\alpha\mathbb{P}' + (1-\alpha)\mathbb{P}''$ of a distribution $\mathbb{P}' \in \text{co}\,\mathbb{K}_C^+$ and a distribution $\mathbb{P}'' \in \mathbb{K}_S^0$, with $\alpha \in (0, 1)$. That implies that $\mathbb{P}(\mathbf{Z} = \mathbf{z}|\mathbf{Y} = \mathbf{y}) = \mathbb{P}'(\mathbf{Z} = \mathbf{z}|\mathbf{Y} = \mathbf{y})$ for any $\mathbf{z}$. Thus we can restrict ourselves to $\text{co}\,\mathbb{K}_C^+$ when verifying maximality. Again, following the same reasoning in the end of the proof of Theorem 4, we have that the minium of the difference $\mathbb{P}(\mathbf{Z} = \mathbf{z}|\mathbf{Y} = \mathbf{y}) - \mathbb{P}(\mathbf{Z} = \mathbf{z}^*|\mathbf{Y} = \mathbf{y})$ is attained at an extreme point of $\mathbb{K}(\mathbf{Z}|\mathbf{Y} = \mathbf{y})$. □

References

[1] S. Kent, Words of estimative probability, in: Sherman Kent and the Board of National Estimates: Collected Essays, 2014, Historical Document (online).
[2] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan Kaufmann, 1988.
[3] A. Darwiche, Modeling and Reasoning with Bayesian Networks, Cambridge University Press, 2009.
[4] D. Koller, N. Friedman, Probabilistic Graphical Models, MIT Press, 2009.
[5] F.G. Cozman, Graphical models for imprecise probabilities, Int. J. Approx. Reason. 39 (2005) 167–184.
[6] A. Antonucci, C. de Campos, M. Zaffalon, Probabilistic graphical models, in: T. Augustin, F.P.A. Coolen, G. de Cooman, M.C.M. Troffaes (Eds.), Introduction to Imprecise Probabilities, Wiley, 2014, pp. 207–229.
[7] A. Piatti, A. Antonucci, M. Zaffalon, Building Knowledge-Based Systems by Credal Networks: A Tutorial, Nova Science, 2010, pp. 227–279.
[8] J. De Bock, Credal networks under epistemic irrelevance, Int. J. Approx. Reason. 85 (2017) 107–138.
[9] N.J. Nilsson, Probabilistic logic, Artif. Intell. 28 (1986) 71–87.
[10] J.R. Quinlan, INFERNO: A Cautious Approach to Uncertain Inference, Technical Report, RAND Corporation, 1982, N-1898-RC.
[11] E.H. Shortliffe, B.G. Buchanan, A model of inexact reasoning in medicine, Math. Biosci. 23 (1975) 351–379.
[12] V.M.H. Coupé, L.L.C. van der Gaag, J.D.F. Habbema, Sensitivity analysis: an aid for belief-network quantification, Knowl. Eng. Rev. 15 (2000) 1–18.
[13] J. Kwisthout, L.C. van der Gaag, The computational complexity of sensitivity analysis and parameter tuning, in: Proceedings of the 24th Conference in Uncertainty in Artificial Intelligence (UAI), 2008, pp. 349–356.
[14] M.T. Lamata, S. Moral, Dependence graphs: upper and lower probabilities, in: System Analysis and Computer Science, 1990, pp. 113–122.
[15] J. Cano, M. Delgado, S. Moral, An axiomatic framework for propagating uncertainty in directed acyclic networks, Int. J. Approx. Reason. 8 (1993) 253–280.
[16] E. Fagiuoli, M. Zaffalon, 2U: an exact interval propagation algorithm for polytrees with binary variables, Artif. Intell. 106 (1998) 77–107.
[17] B. Tessem, Interval probability propagation, Int. J. Approx. Reason. 7 (1992) 95–120.
[18] A. Cano, S. Moral, Algorithms for imprecise probabilities, in: J. Kohlas, S. Moral (Eds.), Handbook of Defeasible and Uncertainty Management Systems, Kluwer Academic Publishers, 2000, pp. 369–420.
[19] I. Levi, The Enterprise of Knowledge, MIT Press, London, 1980.
[20] F.G. Cozman, Imprecise and indeterminate probabilities, in: A. Hájek, C. Hitchcock (Eds.), The Oxford Handbook of Probability and Philosophy, Oxford University Press, 2016, pp. 296–311.
[21] P. Walley, Statistical Reasoning with Imprecise Probabilities, Chapman and Hall, 1991.
[22] F.G. Cozman, Credal networks, Artif. Intell. 120 (2000) 199–233.
[23] S.L. Lauritzen, A.P. Dawid, B.N. Larsen, H.-G. Leimer, Independence properties of directed Markov fields, Networks 20 (1990) 491–505.
[24] R.D. Shachter, Bayes-ball: the rational pastime (for determining irrelevance and requisite information in belief networks and influence diagrams), in: Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence, 1998, pp. 480–487.
[25] K. Weichselberger, The theory of interval-probability as a unifying concept for uncertainty, Int. J. Approx. Reason. 24 (2000) 149–170.
[26] J. De Bock, G. de Cooman, Allowing for probability zero in credal networks under epistemic irrelevance, in: Proceedings of the 8th International Symposium on Imprecise Probability: Theories and Applications (ISIPTA), 2013, pp. 109–118.
[27] G. Coletti, R. Scozzafava, Probabilistic Logic in a Coherent Setting, Trends in Logic, vol. 15, Kluwer, Dordrecht, 2002.
[28] F.G. Cozman, Sets of probability distributions, independence, and convexity, Synthese 186 (2012) 577–600.
[29] D. Avis, A revised implementation of the reverse search vertex enumeration algorithm, in: G. Kalai, G.M. Ziegler (Eds.), Polytopes: Combinatorics and Computation, in: DMV Seminar, vol. 29, Birkhäuser, Basel, 2000, pp. 177–198.
[30] I. Couso, S. Moral, P. Walley, A survey of concepts of independence for imprecise probabilities, Risk Decis. Policy 5 (2000) 165–181.
[31] F.J. Giron, S. Rios, Quasi-Bayesian behaviour: a more realistic approach to decision making?, in: J.M. Bernardo, J.H. DeGroot, D.V. Lindley, A.F.M. Smith (Eds.), Bayesian Statistics, University Press, Valencia, Spain, 1980, pp. 17–38.
[32] V.P. Kuznetsov, Interval Statistical Methods, Radio i Svyaz Publ., 1991 (in Russian).
[33] F. Cozman, C. de Campos, Kuznetsov independence for interval-valued expectations and sets of probability distributions: properties and algorithms, Int. J. Approx. Reason. 55 (2014) 666–682.
[34] C.P. de Campos, F.G. Cozman, Computing lower and upper expectations under epistemic independence, Int. J. Approx. Reason. 44 (2007) 244–260.
[35] M. Wellman, Fundamental concepts of qualitative probabilistic networks, Artif. Intell. 44 (1990) 257–303.
[36] S. Renooij, L.C. van der Gaag, From qualitative to quantitative probabilistic networks, in: Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence, 2002, pp. 422–429.
[37] S. Renooij, L.C. Van der Gaag, Enhanced qualitative probabilistic networks for resolving trade-offs, Artif. Intell. 172 (2008) 1470–1494.
[38] C. de Campos, L. Zhang, Y. Tong, Q. Ji, Semi-qualitative probabilistic networks in computer vision problems, J. Stat. Theory Pract. 3 (2009) 197–210.
[39] K.A. Andersen, J.N. Hooker, Bayesian logic, Decis. Support Syst. 11 (1994) 191–210.
[40] G. de Cooman, M. Zaffalon, Updating beliefs with incomplete observations, Artif. Intell. 159 (2004) 75–125.
[41] A. Antonucci, M. Zaffalon, Equivalence between Bayesian and credal nets on an updating problem, in: Proceedings of the Third International Conference on Soft Methods in Probability and Statistics, 2006, pp. 223–230.
[42] A. Antonucci, M. Zaffalon, Decision-theoretic specification of credal networks: a unified language for uncertain modeling with sets of Bayesian networks, Int. J. Approx. Reason. 49 (2008) 345–361.
[43] F.G. Cozman, Irrelevance and independence axioms in quasi-Bayesian theory, in: A. Hunter, S. Parsons (Eds.), European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU), Springer, 1999, pp. 128–136.
[44] J. De Bock, G. de Cooman, Credal networks under epistemic irrelevance: the sets of desirable gambles approach, Int. J. Approx. Reason. 56 (2015) 178–207.
[45] S. Moral, Epistemic irrelevance on sets of desirable gambles, Ann. Math. Artif. Intell. 45 (2005) 197–214.
[46] B. Vantaggi, Graphical models for conditional independence structures, in: Second International Symposium on Imprecise Probabilities and Their Applications, Shaker, 2001, pp. 332–341.
[47] W.R. Gilks, A. Thomas, D. Spiegelhalter, A language and program for complex Bayesian modelling, Statistician 43 (1993) 169–178.
[48] D. Heckerman, C. Meek, D. Koller, Probabilistic entity-relationship models, PRMs, and plate models, in: L. Getoor, B. Taskar (Eds.), Introduction to Statistical Relational Learning, MIT Press, 2007, pp. 201–238.
[49] B. Milch, B. Marthi, S. Russell, D. Sontag, D.L. Ong, A. Kolobov, BLOG: probabilistic models with unknown objects, in: Proceedings of the 19th International Joint Conference on Artificial Intelligence, 2005, pp. 1352–1359.
[50] D. Fierens, G. van den Broeck, J. Renkens, D. Shrerionov, B. Gutmann, G. Janssens, L. de Raedt, Inference and learning in probabilistic logic programs using weighted Boolean formulas, Theory Pract. Log. Program. 15 (2014) 358–401.
[51] F.G. Cozman, Languages for probabilistic modeling over structured and relational domains, in: P. Marquis, O. Papini, H. Prade (Eds.), A Guided Tour of Artificial Intelligence Research: Artificial Intelligence Algorithms – Volume II: AI Algorithms, Springer, 2020.
[52] F.G. Cozman, D.D. Mauá, On the semantics and complexity of probabilistic logic programs, J. Artif. Intell. Res. 60 (2017) 221–262.

[53] F.G. Cozman, D.D. Mauá, On the complexity of propositional and relational credal networks, Int. J. Approx. Reason. 83 (2017) 298–319.
[54] C.P. de Campos, F.G. Cozman, Inference in credal networks through integer programming, in: Proceeding of the Fifth International Symposium on Imprecise Probability: Theories and Applications (ISIPTA), 2007, pp. 145–154.
[55] D.D. Mauá, C.P. de Campos, M. Zaffalon, Updating credal networks is approximable in polynomial time, Int. J. Approx. Reason. 53 (2012) 1183–1199.
[56] A. Cano, M. Gómez, S. Moral, J. Abellán, Hill-climbing and branch-and-bound algorithms for exact and approximate inference in credal networks, Int. J. Approx. Reason. 44 (2007) 261–280.
[57] C.P. de Campos, F.G. Cozman, Inference in credal networks using multilinear programming, in: Second Starting AI Researcher Symposium, 2004, pp. 50–61.
[58] A. Antonucci, C.P. de Campos, D. Huber, M. Zaffalon, Approximating credal network inferences by linear programming, in: Proceedings of the 12th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU), vol. 7958, 2013, pp. 13–25.
[59] J.C.F. da Rocha, F.G. Cozman, C.P. de Campos, Inference with separately specified sets of probabilities in credal networks, in: Conference on Uncertainty in Artificial Intelligence, 2003, pp. 430–437.
[60] J.S. Ide, F.G. Cozman, Approximate algorithms for credal networks with binary variables, Int. J. Approx. Reason. 48 (2008) 275–296.
[61] K.P. Murphy, Y. Weiss, M.I. Jordan, Loopy belief propagation for approximate inference: an empirical study, in: Conference on Uncertainty in Artificial Intelligence, 1999, pp. 467–475.
[62] A. Antonucci, Y. Sun, C.P. de Campos, M. Zaffalon, Generalized loopy 2U: a new algorithm for approximate inference in credal networks, Int. J. Approx. Reason. 55 (2010).
[63] A. Cano, S. Moral, A genetic algorithm to approximate convex sets of probabilities, in: Proceedings of the Fifth International Conference on Information Processing and Management of Uncertainty in Knowledge Based Systems (IPMU), 1996, pp. 859–864.
[64] A. Cano, J.E. Cano, S. Moral, Convex sets of probabilities propagation by simulated annealing, in: Proceedings of the Fifth International Conference on Information Processing and Management of Uncertainty in Knowledge Based Systems (IPMU), 1994, pp. 4–8.
[65] R. Dechter, Bucket elimination: a unifying framework for reasoning, Artif. Intell. 113 (1999) 41–85.
[66] S.L. Lauritzen, D.J. Spiegelhalter, Local computations with probabilities on graphical structures and their application to expert systems, J. R. Stat. Soc., Ser. B (1988) 157–225.
[67] P.P. Shenoy, G. Shafer, Axioms for probability and belief-function propagation, in: Proceedings of the Fourth Annual Conference on Uncertainty in Artificial Intelligence (UAI), 1988, pp. 169–198.
[68] J. Kohlas, Information Algebras: Generic Structures for Inference, Springer-Verlag, 2003.
[69] C.P. de Campos, F.G. Cozman, The inferential complexity of Bayesian and credal networks, in: Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI), 2005, pp. 1313–1318.
[70] P. Walley, Inferences from multinomial data: learning about a bag of marbles, J. R. Stat. Soc., Ser. B (1996) 3–57.
[71] J.O. Berger, Statistical Decision Theory and Bayesian Analysis, 2nd ed., Springer Series in Statistics, Springer, 1993.
[72] L. de Campos, J. Huete, S. Moral, Probability intervals: a tool for uncertain reasoning, Int. J. Uncertain. Fuzziness Knowl.-Based Syst. 2 (1994) 167–196.
[73] E. Miranda, S. Destercke, Extreme points of the credal sets generated by elementary comparative probabilities, in: Proceedings of the 12th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU), vol. 7958, 2013, pp. 424–435.
[74] A. Charnes, W.W. Cooper, Programming with linear fractional functionals, Nav. Res. Logist. Q. 9 (1962) 181–186.
[75] J.C.F. da Rocha, F.G. Cozman, Inference in credal networks: branch-and-bound methods and the A/R+ algorithm, Int. J. Approx. Reason. 39 (2005) 279–296.
[76] A.M. Lukatskii, D.V. Shapot, Problems in multilinear programming, Comput. Math. Math. Phys. 41 (2000) 638–648.
[77] C. De Boom, J. De Bock, A. Van Camp, G. de Cooman, Robustifying the Viterbi algorithm, in: Proceedings of the Seventh European Workshop on Probabilistic Graphical Models, in: Lecture Notes in Computer Science, vol. 8754, 2014, pp. 160–175.
[78] J. De Bock, C.P. de Campos, A. Antonucci, Global sensitivity analysis for map inference in graphical models, in: Advances in Neural Information Processing Systems, vol. 27, 2014, pp. 2690–2698.
[79] G. Cooper, A method for using belief networks as influence diagrams, in: Fourth Workshop on Uncertainty in Artificial Intelligence, 1988.
[80] D.D. Mauá, C.P. de Campos, M. Zaffalon, The complexity of approximately solving influence diagrams, in: Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence, 2012, pp. 604–613.
[81] K. Fertig, J.S. Breese, Interval influence diagrams, in: Fifth Workshop on Uncertainty in Artificial Intelligence, 1989.
[82] K. Fertig, J.S. Breese, Decision making with interval influence diagrams, in: Uncertainty in Artificial Intelligence, 1990.
[83] R. Cabañas, A. Antonucci, A. Cano, M. Gómez-Olmedo, Evaluating interval-valued influence diagrams, Int. J. Approx. Reason. 80 (2016) 393–411.
[84] M.C.M. Troffaes, Decision making under uncertainty using imprecise probabilities, Int. J. Approx. Reason. 45 (2007) 17–29.
[85] J.C.F. da Rocha, F.G. Cozman, C.P. de Campos, Inference in polytrees with sets of probabilities, in: Proceedings of the 19th Conference on Uncertainty in Artificial Intelligence, 2003, pp. 217–224.
[86] F.G. Cozman, C.P. de Campos, J.S. Ide, J.C.F. da Rocha, Propositional and relational Bayesian networks associated with imprecise and qualitative probabilistic assessments, in: Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence, 2004, pp. 104–111.
[87] C.H. Papadimitriou, Computational Complexity, Addison-Wesley Publishing, 1994.
[88] D. Roth, On the hardness of approximate reasoning, Artif. Intell. 82 (1996) 273–302.
[89] C.P. de Campos, G. Stamoulis, D. Weyland, A structured view on weighted counting with relations to counting, quantum computation and applications, Electron. Colloq. Comput. Complex. 133 (2013), https://eccc.weizmann.ac.il/report/2013/133/, revision 2.
[90] S. Toda, PP is as hard as the polynomial-time hierarchy, SIAM J. Comput. 20 (1991) 865–877.
[91] D.D. Mauá, C.P. de Campos, A. Benavoli, A. Antonucci, On the complexity of strong and epistemic credal networks, in: Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence (UAI), 2013, pp. 391–400.
[92] D.D. Mauá, C.P. de Campos, A. Benavoli, A. Antonucci, Probabilistic inference in credal networks: new complexity results, J. Artif. Intell. Res. 50 (2014) 603–637.
[93] D.D. Mauá, A. Antonucci, C.P. de Campos, Hidden Markov models with set-valued parameters, Neurocomputing 180 (2016) 94–107.
[94] A. Antonucci, C. de Campos, Decision making by credal nets, in: Proceedings of the International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), 2011, pp. 201–204.
[95] M. Zaffalon, E. Fagiuoli, Tree-based credal networks for classification, Reliab. Comput. 9 (2003) 487–509.
[96] J. Kwisthout, Most probable explanations in Bayesian networks: complexity and tractability, Int. J. Approx. Reason. 52 (2011) 1452–1469.
[97] C. Manski, Identification and Prediction of Probability Distributions, Springer, 2003.
[98] A. Salvetti, A. Antonucci, M. Zaffalon, Spatially distributed identification of debris flow source areas by credal networks, in: Transactions of the 4th Bieenial Meeting of the International Congress on Environmental Modelling and Software Integrating Sciences and Information Technology for Environmental Assessment and Decision Making (iEMSs), iEMSs, 2008, pp. 380–387.
[99] A. Antonucci, A. Piatti, M. Zaffalon, Credal networks for operational risk measurement and management, in: International Conference on Knowledge-Based and Intelligent Information & Engineering Systems (KES), in: LNCS, vol. 4693, 2007, pp. 604–611.
[100] A. Antonucci, R. Brühlmann, A. Piatti, M. Zaffalon, Credal networks for military identification problems, Int. J. Approx. Reason. 50 (2009) 666–679.

[101] A. Antonucci, D. Huber, M. Zaffalon, P. Luginbühl, I. Chapman, R. Ladouceur, CREDO: a military decision-support system based on credal networks, in: Proceedings of the 16th Conference on Information Fusion (FUSION 2013), 2013, pp. 1942–1949.
[102] T. Augustin, G. Walter, F.A.P. Coolen, Statistical Inference, 2014.
[103] F. Coolen, On nonparametric predictive inference and objective bayesianism, J. Log. Lang. Inf. 15 (2006) 21–47.
[104] S. Marchetti, A. Antonucci, Reliable uncertain evidence modeling in Bayesian networks by credal networks, in: Proceedings of the 31st International Flairs Conference, 2018, pp. 513–518.
[105] A.R. Masegosa, S. Moral, Imprecise probability models for learning multinomial distributions from data: applications to learning credal networks, Int. J. Approx. Reason. 55 (2014) 1548–1569.
[106] S. Moral, Learning with imprecise probabilities as model selection and averaging, Int. J. Approx. Reason. 109 (2019) 111–124.
[107] M. Zaffalon, Statistical inference for the naive credal classifier, in: Second International Symposium on Imprecise Probabilities and Their Applications, Shaker, Netherlands, 2001, pp. 384–393.
[108] G. Corani, M. Zaffalon, Learning reliable classifiers from small and incomplete data sets: the naive credal classifier 2, J. Mach. Learn. Res. 9 (2008) 581–621.
[109] G. Corani, M. Zaffalon, Credal model averaging: an extension of Bayesian to imprecise probabilities, in: European Conference on Machine Learning and Principles and Practice of Knowledge Discover in Databases, 2008, pp. 257–271.
[110] G. Corani, J. Abellán, A. Masegosa, S. Moral, M. Zaffalon, Classification, in: T. Augustin, F.P.A. Coolen, G. de Cooman, M.C.M. Troffaes (Eds.), Introduction to Imprecise Probabilities, Wiley, 2014, pp. 230–257.
[111] A. Antonucci, G. Corani, The multilabel naive credal classifier, Int. J. Approx. Reason. 83 (2016) 320–336.
[112] G. Corani, A. Mignatti, Robust Bayesian model averaging for the analysis of presence–absence data, Environ. Ecol. Stat. 22 (2015) 513–534.
[113] G. Corani, A. Antonucci, Credal ensembles of classifiers, Comput. Stat. Data Anal. 71 (2014) 818–831.
[114] M. Zaffalon, Credible classification for environmental problems, Environ. Model. Softw. 20 (2005) 1003–1012.
[115] G. Corani, A. Giusti, D. Migliore, J. Schmidhuber, Robust texture recognition using credal classifiers, in: F. Labrosse, R. Zwiggelaar, Y. Liu, B. Tiddeman (Eds.), Proceedings of the British Machine Vision Conference, BMVA Press, 2010, pp. 78.1–78.10.
[116] C.P. de Campos, Q. Ji, Bayesian networks and the imprecise Dirichlet model applied to recognition problems, in: W. Liu (Ed.), Symbolic and Quantitative Approaches to Reasoning with Uncertainty, in: Lecture Notes in Computer Science, vol. 6717, Springer, Berlin/Heidelberg, 2011, pp. 158–169.
[117] A. Antonucci, R. de Rosa, A. Giusti, Action recognition by imprecise hidden Markov models, in: Proceedings of the 2011 International Conference on Image Processing, Computer Vision and Pattern Recognition (IPCV), 2011, pp. 474–478.
[118] Y. Soullard, A. Antonucci, S. Destercke, Technical gestures recognition by set-valued hidden Markov models with prior knowledge, in: Soft Methods for Data Science, in: Advances in Intelligent Systems and Computing, vol. 456, 2017, pp. 455–462.
[119] A. Antonucci, R. de Rosa, A. Giusti, F. Cuzzolin, Robust classification of multivariate time series by imprecise hidden Markov models, Int. J. Approx. Reason. 56 (2015) 249–263.
[120] A. Antonucci, G. Corani, D.D. Mauá, S. Gabaglio, An ensemble of Bayesian networks for multilabel classification, in: Proceedings of the 23rd International Joint Conference on Artificial Intelligence, 2013, pp. 1220–1225.
[121] M. Zaffalon, G. Corani, D.D. Mauá, Evaluating credal classifiers by utility-discounted predictive accuracy, Int. J. Approx. Reason. 53 (2012) 1282–1301.