Robustness Guarantees for Credal Bayesian Networks via Constraint Relaxation over Probabilistic Circuits

Hjalmar Wijk , Benjie Wang , Marta Kwiatkowska
University of Oxford
hjalmar.wijk@st-annes.ox.ac.uk, benjie.wang@keble.ox.ac.uk, marta.kwiatkowska@cs.ox.ac.uk

Abstract

In many domains, worst-case guarantees on the performance (e.g., prediction accuracy) of a decision function subject to distributional shifts and uncertainty about the environment are crucial. In this work we develop a method to quantify the robustness of decision functions with respect to credal Bayesian networks, formal parametric models of the environment where uncertainty is expressed through *credal sets* on the parameters. In particular, we address the maximum marginal probability ($\texttt{MAR}_{\max}$) problem, that is, determining the greatest probability of an event (such as misclassification) obtainable for parameters in the credal set. We develop a method to faithfully transfer the problem into a constrained optimization problem on a probabilistic circuit. By performing a simple constraint relaxation, we show how to obtain a guaranteed upper bound on $\texttt{MAR}_{\max}$ in linear time in the size of the circuit. We further theoretically characterize this constraint relaxation in terms of the original Bayesian network structure, which yields insight into the tightness of the bound. We implement the method and provide experimental evidence that the upper bound is often near tight and demonstrates improved scalability compared to other methods.

1 Introduction

Probabilistic models allow us to make quantitative inferences about the behaviour of complex systems, and are an important tool to guide their use and design. When such models are learnt from data, exposed to potential distribution shifts or are partially unknown, it is important to be able to verify the robustness of inferences on the model to these uncertainties. This is particularly relevant for decision functions taking action in the model, where much work has gone into verifying worst-case behaviour when exposed to various disturbances or changes in the environment (distribution shifts). Causal Bayesian networks (BNs) [Pearl, 1985] are compelling models for this purpose, since one can perform causal interventions on them, giving rise to families of distributions that share a common structure. However, performing useful inference on BNs is often intractable, and one way to address this is to compile them into more tractable representations such as arithmetic circuits [Darwiche, 2003]. Recent work has shown that such compilation methods can also efficiently compute bounds on a decision function's robustness to causal interventions [Wang *et al.*, 2021]. A limiting factor on the applicability of these methods is the need to have an exact model, where all non-intervened parameters are known precisely. This is difficult to achieve when learning parameters from data, since most settings will only allow reliable determination up to some error bound ϵ.

In this paper we study robustness of credal Bayesian networks (CrBNs) [Mauá and Cozman, 2020], a generalisation of Bayesian networks where parameters are only known to be within some credal sets (e.g., intervals). They can be used to model causal interventions, but are also very well suited to modelling parameters learned from data, as well as modelling of exogenous variables [Zaffalon *et al.*, 2020].

We consider the maximum marginal probability ($\texttt{MAR}_{\max}$) problem for CrBNs and develop a solution by encoding the network as a tractable probabilistic circuit (a credal extension of sum-product networks, called CSPNs). More specifically, this paper makes the following contributions: (i) a method for constructing a probabilistic circuit whose parameters semantically represent the conditional probability distributions of a BN, allowing the transfer of credal inference problems from a highly intractable setting (CrBNs) to a tractable one (CSPN) through constraint relaxation; (ii) algorithms which make use of this transfer to compute upper and lower bounds on probabilities of events under many forms of parameter uncertainty; (iii) a characterization of the tightness of the upper bound in terms of the network structure; and (iv) an evaluation on a set of benchmarks, demonstrating comparable precision and significantly improved scalability compared to state-of-the-art credal network inference, while also providing formal guarantees.

Due to space constraints some details and proofs can be found in the Appendix of the extended paper at http://www.fun2model.org/bibitem.php?key=WWK22

1.1 Related Work

The problem of robustness of inferences under imprecise knowledge of the distribution has been studied under many guises. In the machine learning community, there has been much work on robustness of classifiers to simple adversarial

attacks or distribution shifts [Quiñonero-Candela *et al.*, 2009; Zhang *et al.*, 2015; Lipton *et al.*, 2018]. Motivated by safety concerns, methods have been developed to compute formal guarantees of robustness through constraint solving [Katz *et al.*, 2017; Narodytska *et al.*, 2018] or output reachability analysis [Ruan *et al.*, 2018]. However, these methods do not model the environment, and are thus limited in the types of distributional shifts they can address.

In the Bayesian network literature, robustness has primarily been studied in terms of the effect of parameters on inference queries, such as marginal probabilities. For instance, sensitivity analysis [Coupé *et al.*, 2000; Chan and Darwiche, 2004] is concerned with the effect of small, local changes/perturbations to parameters. Closer to our work is the formalism of credal networks [Mauá and Cozman, 2020], which represent imprecise knowledge by positing sets of parameters for each conditional distribution, rather than precise values. Inference then corresponds to computing maximal (or minimal) probabilities over the possible parameter values. Unfortunately, exact methods for inference in credal networks do not perform well except for smaller networks with simple credal sets, or in special cases such as polytrees [Fagiuoli and Zaffalon, 1998; De Campos and Cozman, 2007]. On the other hand, approximate methods [Cano *et al.*, 2007; Antonucci *et al.*, 2010; Antonucci *et al.*, 2015] usually cannot provide theoretical guarantees (upper bounds), limiting their applicability in safety-critical scenarios.

This paper builds on work showing the tractability of credal inference for certain probabilistic circuits [Mauá *et al.*, 2017] [Mattei *et al.*, 2020]. Our key contribution is a method for mapping credal network problems into tractable credal inference problems on probabilistic circuits, which affords not only greater scalability compared to the state-of-the-art in credal network inference, but also provides formal guarantees.

Finally, methods for providing robustness guarantees for classifiers in combination with a Bayesian network environment model have recently been proposed [Wang *et al.*, 2021]. Our paper generalizes and extends their work, enabling efficient computation for broader and more realistic classes of parameter uncertainty.

2 Background

A Bayesian network (BN) $\mathcal{N} = (\mathcal{G}, \Theta)$ over discrete variables $\boldsymbol{V} = \{V_1, ..., V_n\}$ consists of a directed acyclic graph (DAG) $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ and a set of parameters Θ. It is a factoring of a joint probability distribution $p_{\mathcal{N}}$ into conditional distributions for each variable, such that

$$p_{\mathcal{N}}(V_1, ..., V_n) = \prod_{i=1}^{n} p_{\mathcal{N}}(V_i | \text{pa}(V_i)),$$

where the parents $\text{pa}(V_i)$ of V_i are the set of variables V_j such that $(V_j, V_i) \in \boldsymbol{E}$. Θ is the set of parameters of the form:

$$\theta_{v_i|\boldsymbol{u}_i} = p_{\mathcal{N}}(V_i = v_i | \boldsymbol{U}_i = \boldsymbol{u}_i),$$

for each instantiation $v_i, \boldsymbol{u}_i$ of a variable V_i and its parents $\boldsymbol{U}_i$.

Given a Bayesian network model, to obtain useful information about the distribution we will need to perform inference. For example, we might wish to obtain the probability $p_{\mathcal{N}}(\boldsymbol{W} = \boldsymbol{w})$ for some subset of variables $\boldsymbol{W} \subseteq \boldsymbol{V}$, a procedure known as marginalization. In the worst case, marginal inference in Bayesian networks is known to be #P-complete, though many practical inference methods exist.

Given a classifier, we can represent its input-output behaviour using a decision function $F : \boldsymbol{X} \to Y$, which observes some subset $\boldsymbol{X} \subseteq \boldsymbol{V}$ and tries to predict $Y \in \boldsymbol{V}$. To combine this with a Bayesian network environment model $\mathcal{N}$, we follow [Wang *et al.*, 2021] in the construction of an *augmented BN* $\mathcal{N}_F$, which is a Bayesian network based on $\mathcal{N}$ where an additional variable (node) $\hat{Y}$ is added with $\text{pa}(\hat{Y}) = \boldsymbol{X}$ and $p_{\mathcal{N}_F}(\hat{Y} = \hat{y} | \boldsymbol{X} = \boldsymbol{x}) = \mathbb{1}[\hat{y} = F(\boldsymbol{x})]$. $\mathcal{N}_F$ is thus a unified model of environment and decision maker, and inference on the model can answer questions such as the prediction accuracy $p_{\mathcal{N}_F}(\hat{Y} = Y)$.

3 Robust Inference on Bayesian Networks

It is rarely the case that we can specify the parameters of a Bayesian network with complete certainty before performing inference. Firstly, whether the parameters are learned from data or elicited from expert knowledge, the knowledge that we obtain regarding the parameters is typically imprecise, specified as sets or intervals. Secondly, when the Bayesian network is imbued with a causal interpretation, one is often concerned about potential distribution shift, modelled by causal interventions, and their effect on inference queries.

As a running example, consider a fictional scenario depicted in Figure 1, where patients are infected by some unobservable strain S of a disease, with some strains much more severe than others, and a decision rule F must be created based on observable symptoms V and test results R that decides whether to administer an expensive treatment option. While it is desirable to save resources by only administering treatment for the more severe strains, the result of denying treatment to a patient with a severe strain would be disastrous, so a guarantee is needed that the decision rule has a robustly low probability of an erroneous decision. To provide such a guarantee we model the system as an augmented BN $\mathcal{N}_F$ over variables $\{S, V, R, T\}$, where T is binary and deterministic, given by $F(v, r)$. However, we do not have precise knowledge over the parameters of the BN, so we instead design intervals which specify the range of values a parameter could take.

We start with θ_S, the distribution over the strains. We expect that the decision rule will be deployed across a variety of areas and times, and as such we are concerned about distributional shifts in θ_S. We could thus decide to allow any probability distribution across the strains, i.e. $\theta_{S=s} \in [0, 1]$. We imagine the tests used are very well understood, and we know $\theta_{R|s}$ exactly. Moving onto the parameters $\theta_{V|s}$, describing the symptoms of a particular strain, we might expect that these, unlike θ_S, are relatively fixed across different settings. However, gathering enough data on each strain and symptom combination to be certain of this (fixed) parameter value might turn out to be challenging. In this case it might be suitable to

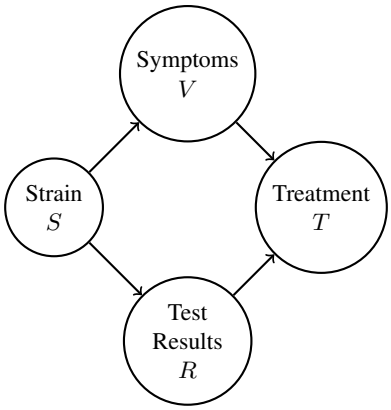


Figure 1: The DAG of an augmented BN modelling a simple fictional medical treatment scenario.

take mean estimates of the parameter values $\theta^*_{V=v|s}$, and then select some confidence interval $[\theta^*_{V=v|s} - \epsilon, \theta^*_{V=v|s} + \epsilon]$.

3.1 Credal Bayesian Networks

To formalize the prior discussion, we use credal Bayesian networks [Mauá and Cozman, 2020], a framework that encompasses both causal interventions and imprecise knowledge of parameters.

Definition 1. *Let $p_\Theta, \Theta \in \boldsymbol{\Theta}$, be any parameterised probability distribution, where $\boldsymbol{\Theta}$ is the set of allowed parameter values. Then we call $\boldsymbol{\mathcal{C}} \subseteq \boldsymbol{\Theta}$ a* credal set *for this parameterisation, and the* credal family $\mathbb{C}_p[\boldsymbol{\mathcal{C}}] = \{p_{\Theta'} | \Theta' \in \boldsymbol{\mathcal{C}}\}$ *is the family of distributions where the parameters are in $\boldsymbol{\mathcal{C}}$.*

This is a maximally expressive formalism for credal uncertainty. However, an independence assumption between the uncertainty of different conditional distributions in a BN (sometimes known as the strong extension [Cozman, 2000]) is usually assumed:

Definition 2. *[Cozman, 2000] A Credal BN (CrBN) $\mathbb{C}_\mathcal{N}[\boldsymbol{\mathcal{C}}] = \{\mathcal{N}_\Theta | \Theta \in \boldsymbol{\mathcal{C}}\}$ over a BN $\mathcal{N}_\Theta = (\mathcal{G}, \Theta)$ is a credal family satisfying*

$$\boldsymbol{\mathcal{C}} = \prod_{V_i, \boldsymbol{u_i}} \mathcal{C}_{V_i|\boldsymbol{u_i}},$$

i.e. the credal set decomposes as a cartesian product of separate credal sets for each variable V_i and instantiation of its parent variables $\boldsymbol{u_i}$.

Since augmented Bayesian networks are simply Bayesian networks with an additional deterministic node (the decision function), we can convert any credal set over a Bayesian network model $\mathcal{N}$ to a credal set over $\mathcal{N}_F$ by maintaining the credal sets for all variables, while assuming the conditional distribution for the new variable $\hat{Y}$ is known exactly. This framework then fully generalizes the "interventional robustness problem" introduced by [Wang *et al.*, 2021] to allow arbitrary credal sets for parameters; see Appendix for details.

3.2 Problem Definition

In the treatment example we wished to guarantee the worst-case probability of an event occurring over a CrBN. We will now formalise this problem.

	s_1	s_2	s_3
θ_R	0.95	0.05	0.5
θ_V	0.2 ± 0.1	0.8 ± 0.1	0.6 ± 0.2

Table 1: Credal sets for symptom and test result parameters.

Definition 3. *Given an (augmented) CrBN $\mathbb{C}_\mathcal{N}[\boldsymbol{\mathcal{C}}]$ and an event e (an instantiation of a subset of the variables), the maximum marginal probability ($\texttt{MAR}_{\max}$) problem is that of determining*

$$\texttt{MAR}_{\max}(\mathcal{N}, \boldsymbol{\mathcal{C}}, e) = \max_{\Theta \in \boldsymbol{\mathcal{C}}} p_{\mathcal{N}_\Theta}(e).$$

This generalization of causal interventions enables many new problems to be considered, as causal interventions require parameters to be known exactly or be entirely unknown. Crucially, it allows us to model parameters which are estimated from data to be within some interval. It also allows the degree of uncertainty to depend on the value of the parents, as it might if some parent values are rare and lack data points.

As an illustration we now define a CrBN over $\mathcal{N}_F$ to formalize the treatment scenario in Figure 1. We imagine there are three strains s_1, s_2, s_3, of which only s_3 is severe and requires treatment. We take V and R to be binary variables (symptomatic/asymptomatic and positive/negative test). We wish to be able to apply the decision rule in any situation where the prevalence of s_3 is at most 0.1, so we assign $\mathcal{C}_S = \{\boldsymbol{\theta} \in Z_3 | \theta_{S=s_3} < 0.1\}$, where Z_3 is the three-dimensional probability simplex. We use singleton credal sets for R and confidence intervals for V, with the values given in Table 1. The decision rule to be analysed gives treatment when $R = V$, since this is unlikely for s_1 and s_2.

This is an instance of the maximum marginal probability $\texttt{MAR}_{\max}$ problem, where the CrBN $\mathbb{C}_\mathcal{N}[\boldsymbol{\mathcal{C}}]$ is as specified above, and the event of interest is $e = (T = 0) \wedge (S = s_3)$.

4 Credal Robustness via Probabilistic Circuits

In this section, we present an efficient method for bounding $\texttt{MAR}_{\max}$ credal robustness for Bayesian networks with guarantees. In particular, the method returns an upper bound on $\texttt{MAR}_{\max}$. In the treatment example, this would mean that we can be certain that the probability of denying treatment to a patient with the severe strain does not exceed the computed value, assuming all parameters lie within the credal sets. Our method is based upon establishing a correspondence between credal BNs and credal sum-product networks (CSPN) [Mauá *et al.*, 2017], a recently proposed model which introduces uncertainty sets over the weights of a sum-product network. In particular, we develop an algorithm for *compiling* CrBNs into equivalent CSPNs. By efficiently solving a similar credal maximization problem on the CSPN, we can derive upper bounds on $\texttt{MAR}_{\max}$ for the original CrBN.

4.1 Compilation to Arithmetic Circuits

The first step of our method is to compile the credal Bayesian network to an arithmetic circuit. To describe this, we first consider an alternative representation of a Bayesian network.

Definition 4. *[Darwiche, 2003] The* network polynomial *of a BN $\mathcal{N}$ is defined as*

$$l_N[\boldsymbol{\lambda}, \Theta] = \sum_{v_1,...,v_n} \prod_{i=1}^{n} \theta_{v_i|u_i} \lambda_{v_i},$$

where λ_{v_i} are indicator variables for variable V_i, which take the value 1 if $V_i = v_i$ and 0 otherwise.

The network polynomial is a multilinear function which unambigously encodes the graphical structure of the Bayesian network, for any value of the parameters Θ. In particular, one can obtain the joint probability $p_{\mathcal{N}}(v_1, ..., v_n)$ for any instantiation $v_1, ..., v_n$ by setting the indicator variables and evaluating the network polynomial. Unfortunately, it has an exponential number of terms in the number of variables of the BN, which means we cannot use it directly. The goal of compilation is to represent the network polynomial more efficiently, by exchanging sums and products where possible. The result of such a procedure can be interpreted as a rooted directed acyclic graph (DAG) called an arithmetic circuit.

Definition 5. *[Darwiche, 2003] An arithmetic circuit (AC) $\mathcal{T}$ over variables $\boldsymbol{V}$ and parameters Θ is a rooted DAG, whose internal nodes are labelled with $+$ or $\times$ and whose leaf nodes are labelled with indicator variables λ_v*

or non-negative parameters. For an internal node t we will write $\mathcal{T}_t$ for the arithmetic circuit containing t and all its descendants.

Definition 6. *[Chan and Darwiche, 2006] A complete subcircuit α of an AC is obtained by traversing the circuit top-down, choosing one child of every visited $+$-node and all children of every visited $\times$-node. The term* $\text{term}(\alpha)$ *of α is the product of all leaf nodes visited (i.e. all indicator and parameter variables). The AC polynomial $l_{\mathcal{T}}[\boldsymbol{\lambda}, \Theta]$ is the sum of the terms of all complete subcircuits.*

Compilation will produce an AC $\mathcal{T}$ which has the same polynomial as the BN, i.e. $l_N[\boldsymbol{\lambda}, \Theta] = l_{\mathcal{T}}[\boldsymbol{\lambda}, \Theta]$. In addition, it will satisfy technical conditions called *decomposability*, *determinism* and *smoothness*, which allow us to perform many inference queries in linear time in the size of the circuit.

In [Wang *et al.*, 2021] a method is described for compiling an augmented BN to a smooth, decomposable and deterministic AC, which allows one to tractably compute marginal probabilities involving both the decision function and Bayesian network variables. In order to support further queries, they additionally impose ordering constraints on the AC.

Definition 7. *A $+$-node t with children $t_1, ..., t_n$ in an arithmetic circuit $\mathcal{T}$* splits *on variable V_i if there exists an ordering of the domain $v_i^1, ..., v_i^n$ of V_i such that all complete subcircuits of $\mathcal{T}_{t_i}$ contain the indicator λ_{v_i}.*

Definition 8. *Let $\sigma = (V_1, ..., V_n)$ be a topological ordering of the variables in BN $\mathcal{N}$. We say that an (smooth, decomposable, deterministic) arithmetic circuit $\mathcal{T}$ computes the BN $\mathcal{N}$ respecting σ if:*

1. *$l_{\mathcal{T}}[\boldsymbol{\lambda}, \Theta] = l_{\mathcal{N}}[\boldsymbol{\lambda}, \Theta]$*
2. *Each $+$-node in $\mathcal{T}$ splits on some variable V_i. We define split(t) for a $+$-node t to be the variable it splits on.*
3. *The variables are split respecting the topological order. That is, if $V_i = split(t), V_j = split(t')$, then*

$$t' \text{ is a descendant of } t \implies j > i.$$

In other words, it is required that the AC represents the same polynomial as the BN, and further that the AC satisfies particular structural constraints that mean that the AC must split on parents before children. This leads to the following new result, which intuitively means that, when an AC splits on variable V, the values of its parents are already known.

Lemma 1. *Suppose that $\mathcal{T}$ computes $\mathcal{N}$ respecting some topological order. Let t be a $+$-node in $\mathcal{T}$ splitting on some variable V. Then all complete subcircuits α which include t must agree on the value of its parents $pa_{\mathcal{N}}(V)$.*

The AC compiled from the treatment example (Figure 1) is too large to include in its entirety, but Figure 2a shows one branch from the root sum node, with $+$-nodes labelled with the variable they split on. Notice that the topological order (S, R, V, T) is respected, and that, at every $+$-node, the value of the parents of the splitting variable are already "known".

4.2 Compiling to Credal SPNs

While this compiled AC allows us to efficiently compute marginals for given parameter values Θ, it does not effectively represent credal sets, and thus finding maximizing parameter values is challenging (one would need to solve constraints potentially spread out across the whole circuit).

In the next step of our method, we further compile the AC to a sum-product network (SPN). SPNs differ from ACs in that they lack parameter nodes and instead have parameters (i.e. weights) associated with branches from sum nodes.

Definition 9. *[Poon and Domingos, 2011] A sum-product network (SPN) over variables $\boldsymbol{V}$ and with weights W is a rooted DAG whose internal nodes are labelled with either $+$ or $\times$, and whose leaf nodes are labelled with indicator variables λ_v. The branches of a sum node t_i with k branches are labelled with weights $w_{i,1}, .., w_{i,k}$.*

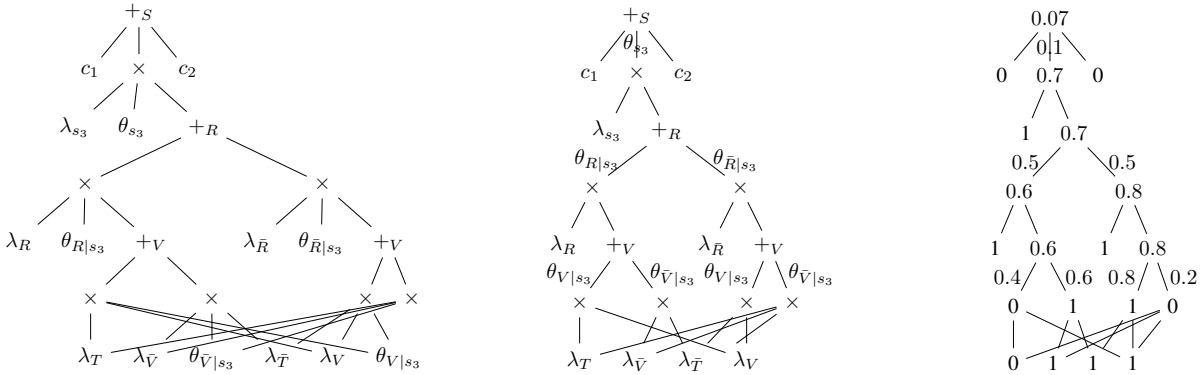
Definition 10. *A complete subcircuit α of an SPN S is obtained by traversing the circuit top-down, choosing one child of every visited $+$-node and all children of every visited $\times$-node. The term* $\text{term}(\alpha)$ *of α is the product of all leaf nodes visited (i.e. all indicators) and all weights w_{ij} along branches chosen by the subcircuit.*

The SPN-polynomial $l_{\mathcal{S}}[\boldsymbol{\lambda}, W]$ is the sum of the terms of all complete subcircuits.

Our compilation differs from that presented in [Rooshenas and Lowd, 2014] in that we make use of the particular structure of the AC, shown in Lemma 1, to make sure the weights on the sum nodes directly correspond to the parameters in the BN (which would not be the case under standard compilation).

At a high level, the compilation only involves two steps:

1. For each sum node t splitting on V_i, assign weights over branches according to $\boldsymbol{\theta}_{V_i|u_i}$, where all variables in u_i are known due to Lemma 1.
2. Remove all parameter nodes.



(a) AC compiled from the augmented BN

(b) SPN compiled from the AC

(c) Execution of $\texttt{MAR}_{\max}(\boldsymbol{\mathcal{S}}, \bar{T} \wedge s_3)$ on the constructed CSPN $\boldsymbol{\mathcal{S}}$ over the SPN.

Figure 2: Illustration of Algorithm 2 for the treatment example. Due to space constraints we show only the $S = s_3$ branch of the AC/SPN.

To algorithmically decide which parameters correspond to a particular sum node we construct a notion of 'possible values' for variables at nodes in the SPN. We first say that a node 'conditions' on $V = v$ if the node is a parameter node $\theta_{W=w|V=v,\boldsymbol{U}=\boldsymbol{u}}$, and define

$$\boldsymbol{P}_t(V) = \{v : \exists t' \in \text{descendants}(t), t' \text{ conditions on } V = v\}.$$

Corollary 1. *For any sum node t splitting on V, if W is a parent of V then $\boldsymbol{P}_t(W)$ must contain exactly one possible value.*

Proof. Any complete subcircuit corresponds to a term of the network polynomial, and must contain a parameter $\theta_{V|w_i,\boldsymbol{u_i}}$ for some $w_i, \boldsymbol{u_i}$. This parameter cannot occur as an ancestor of t, since it would then be impossible to satisfy Definition 7, and so it must be a descendant. Thus $\boldsymbol{P}_t(V)$ is non-empty. By Lemma 1, it cannot contain more than 1 element. □

These sets uniquely determine the values of all parents, and thus which parameters to use. The sets can be efficiently computed , as described in Algorithm 1.

Figure 2b shows the result for the AC in Figure 2a.

Algorithm 1: SPN compilation from AC

Input: AC $\mathcal{T}$ computing $\mathcal{N}$ and satisfying Definition 8.
Result: An SPN computing $\mathcal{N}$ where all sum node weights correspond to a CPT $\boldsymbol{\theta}_{V_i|u_i}$

begin
- For indicator nodes t, assign $\boldsymbol{P}_t(V) = \{\}$;
- For parameter nodes $\theta_{v|u_1,u_2,\ldots}$, assign $\boldsymbol{P}_t(U_i) = \{u_i\}$ if $U_i \in \text{pa}(V)$ and $\boldsymbol{P}_t(U_i) = \{\}$ otherwise;
- For inner nodes t compute $\boldsymbol{P}_t(V) = \bigcup_{c \in \text{children}(t)} \boldsymbol{P}_c(V)$;
- For sum nodes t, label the edges according to $\boldsymbol{\theta}_{V|u_1,u_2,\ldots}$, where t splits on V and u_i is the unique value in $\boldsymbol{P}_t(U_i)$;
- Remove all parameter nodes.

Proposition 1. *Given an arithmetic circuit $\mathcal{T}$ which computes $\mathcal{N}$ satsifying some topological order, the SPN $\mathcal{S}$ compiled as above satisfies*

$$l_{\mathcal{T}}[\boldsymbol{\lambda}, \Theta] = l_{\mathcal{S}}[\boldsymbol{\lambda}, \Theta].$$

Proof. We can put the complete subcircuits of $\mathcal{T}$ and $\mathcal{S}$ in a one-to-one correspondence by the choice of branch at each sum node (since only the parameter nodes/weights have changed). Let α_T be a subcircuit of $\mathcal{T}$, and α_S the corresponding subcircuit of $\mathcal{S}$. For every variable V, α_T contains exactly one $+$-node splitting on V, and exactly one parameter of the form $\theta_{V|pa(V)}$. The compilation procedure moves this parameter to be a weight of the $+$-node splitting on V, so that the overall term is unchanged. Applying this to all variables, we have that $\text{term}(\alpha_T) = \text{term}(\alpha_S)$, and thus the result. □

For credal families over SPNs satisfying an independence requirement between all sum nodes (such that knowing the weights of one sum node does not affect your uncertainty over other weights), $\texttt{MAR}_{\max}$ can be computed efficiently. These are exactly the Credal SPNs introduced in [Mauá *et al.*, 2017].

Definition 11. *[Mauá* et al.*, 2017] A Credal SPN (CSPN) is a credal family $\mathbb{C}_S[\boldsymbol{\mathcal{C}}]$ over an SPN S satisfying*

$$\boldsymbol{\mathcal{C}} = \prod_{i=1}^{n} \mathcal{C}_i,$$

where $\mathcal{C}_i$ is a subset of a probability simplex on the weights of sum node i.

We can construct a credal family over the compiled SPN, which is equivalent to our CrBN, by requiring all sum nodes that split on a variable V_i and have parents $\boldsymbol{u_i}$ to *(i)* all have the same weights and *(ii)* have that weight be in $\mathcal{C}_{V_i|u_i}$. However, this will not in general be a CSPN, since (i) breaks the independence requirement (observing the weights of one sum node will change the credal set for a different sum node if they both split on the same variable with the same values of the parents).

We can, however, construct a CSPN by removing requirement (i), which creates a strictly larger credal family. This relaxation is the final step of our compilation process.

Lemma 2. *For a CrBN $\mathbb{C}_{\mathcal{N}}[\boldsymbol{\mathcal{C}}_{\mathcal{N}}]$ and its compiled CSPN $\mathbb{C}_{\mathcal{S}}[\boldsymbol{\mathcal{C}}_{\mathcal{S}}]$,*

$$\max_{\Theta \in \boldsymbol{\mathcal{C}}_{\mathcal{S}}} l_S[\boldsymbol{\lambda}, \Theta] \geq \max_{\Theta' \in \boldsymbol{\mathcal{C}}_{\mathcal{N}}} l_{\mathcal{N}}[\boldsymbol{\lambda}, \Theta'].$$

Proof. For any given $\Theta \in \boldsymbol{\mathcal{C}}_{\mathcal{N}}$ we have $l_S[\boldsymbol{\lambda}, \Theta] = l_{\mathcal{N}}[\Theta, \boldsymbol{\lambda}]$, by Proposition 1 and the fact that $\mathcal{S}$ computes $\mathcal{N}$. The only way to violate the inequality is if $\Theta \notin \boldsymbol{\mathcal{C}}_S$. But $\boldsymbol{\mathcal{C}}_S$ only demands that at each sum node t splitting on V_i the parameters are in $\mathcal{C}_{V_i|u_i}$, which will certainly be true if $\Theta \in \boldsymbol{\mathcal{C}}_{\mathcal{N}}$. □

If we apply this to construct a CSPN for our treatment example, it will be a CSPN over the SPN in Figure 2b, where the weights of the sum nodes are constrained by the credal sets of the CrBN defined in Section 3.2.

4.3 Solving $\texttt{MAR}_{\max}$

Analogously to CrBNs, we can define the maximum marginal probability problem for CSPNs as $\texttt{MAR}_{\max}(\mathcal{S}, \boldsymbol{\mathcal{C}}, e) = \max_{\Theta \in \boldsymbol{\mathcal{C}}} l_{\mathcal{S}}[\boldsymbol{\lambda}_e, \Theta]$, where $\boldsymbol{\lambda}_e$ refers to the appropriate instantiation of the indicators for the event e.

While $\texttt{MAR}_{\max}$ for the AC (and BN) is intractable, we can compute it efficiently for a CSPN [Mauá *et al.*, 2017].

Proposition 2. *Given a Credal SPN $\mathbb{C}_{\mathcal{S}}[\prod_{i=1}^{n} \mathcal{C}_i]$, we can solve $\texttt{MAR}_{\max}$ for this family of distributions in $\mathcal{O}(|\mathcal{S}|L)$, where L is an upper bound on solving $\max_{w_i} \sum_j w_{ij} c_j$ subject to $w_i \in C_i$.*

Proof. If we assume the maximum possible value of the children $c_1, ..., c_j$ of a sum node t_i are known, finding the maximum possible value of t_i can be done by solving $\max_{w_i} \sum_j w_{ij} c_j$ subject to $w_i \in C_i$. The same is true for product nodes, with the maximum value being the product of the maximum values of its children. By induction we can find the maximum possible value of the root node through bottom-up evaluation. For details see [Mauá *et al.*, 2017]. □

Figure 2c illustrates the computation on the CSPN compiled from the treatment example. The algorithm evaluates nodes bottom up in the graph, with the indicators set to their appropriate value ($\lambda_T = \lambda_{s_1} = \lambda_{s_2} = 0$, the rest 1). The s_1, s_2 branches always lead to an indicator with the value 0. When a sum node is reached, the maximizing weights allowed by the credal set at that sum node are picked. For the left $+_V$ node this means assigning $\theta_{V|s_3}$ the lowest weight allowed (0.4), while the right $+_V$ is instead maximized with the highest weight allowed (0.8). No choice is made at $+_R$ since it is a singleton, and at $+_S$ the maximum weight for s_3 (0.1) is chosen. This demonstrates that $\texttt{MAR}_{\max}(S, \boldsymbol{\mathcal{C}}, \bar{T} \wedge s_3) = 0.07$.

Our overall method **CUB** is summarized in Algorithm 2. The following theorem, which follows directly from Lemma 2, shows that we do indeed return an upper bound:

Theorem 2. *The output $\texttt{MAR}_{\max}(\mathcal{S}, \boldsymbol{\mathcal{C}}_{\mathcal{S}}, e)$ returned by Algorithm 2 satisfies*

$$\texttt{MAR}_{\max}(S, \boldsymbol{\mathcal{C}}_{\mathcal{S}}, e) \geq \texttt{MAR}_{\max}(\mathcal{N}, \boldsymbol{\mathcal{C}}_{\mathcal{N}}, e)$$

Thus, we find that the probability of not assigning treatment to a patient with the severe strain in the treatment example can be no greater than $\texttt{MAR}_{\max}(\mathcal{S}, \boldsymbol{\mathcal{C}}, \bar{T} \wedge s_3) = 0.07$.

Algorithm 2: Upper Bounding for $\texttt{MAR}_{\max}$

Input: Credal Bayesian Network $\mathbb{C}_{\mathcal{N}}[\boldsymbol{\mathcal{C}}_{\mathcal{N}}]$, event e, order σ
Result: Upper bound on $\texttt{MAR}_{\max}(\mathcal{N}, \boldsymbol{\mathcal{C}}_{\mathcal{N}}, e)$
begin
  Compile $\mathcal{N}$ to AC obeying topological order σ;
  Construct a credal SPN $\mathbb{C}_{\mathcal{S}}[\boldsymbol{\mathcal{C}}_{\mathcal{S}}]$ from the AC;
  Compute $\texttt{MAR}_{\max}(S, \boldsymbol{\mathcal{C}}_{\mathcal{S}}, e)$ for this credal SPN;
Return $\texttt{MAR}_{\max}(S, \boldsymbol{\mathcal{C}}_{\mathcal{S}}, e)$

4.4 Tightness of Upper Bound

Though our algorithm provides an upper bound on $\texttt{MAR}_{\max}$ for the Bayesian network, it will not typically be tight. This is illustrated in Figure 2c, where the two different sum nodes representing $\boldsymbol{\theta}_{V|s_3}$ are assigned different weights by the maximization in the CSPN, while this is not possible for the original CrBN. We will now provide a precise characterization of the looseness of the upper bound, using the concept of structural enrichment to find an enriched CrBN which can be put in a 1-1 correspondence with the CSPN.

Definition 12. *A structural enrichment of a CrBN $\mathbb{C}_{\mathcal{N}}[\boldsymbol{\mathcal{C}}]$ is a new CrBN $\mathbb{C}_{\mathcal{N}'}[\boldsymbol{\mathcal{C}}']$ with a new underlying graph $(\boldsymbol{V}, \boldsymbol{E}')$ such that $\boldsymbol{E} \subseteq \boldsymbol{E}'$, and a new credal set given by*

$$(\forall \boldsymbol{w}_i \in \boldsymbol{W}_i) \mathcal{C}'_{V_i|\boldsymbol{u}_i, \boldsymbol{w}_i} = \mathcal{C}_{V_i|\boldsymbol{u}_i},$$

where $\boldsymbol{U}_i$ are the parents of V_i in $\mathcal{N}$, while $\boldsymbol{W}_i$ are the newly added parents in $\mathcal{N}'$ which were not parents in $\mathcal{N}$.

To illustrate this, suppose that we had a BN with 3 variables A, B, C, where A is the only parent of C and we have the credal set $\theta_{C=0|A=0} \in [0.3, 0.8]$. If we now consider a structurally enriched BN where A, B are both parents of C, then we have the same interval $\theta_{C=0|A=0,B=b} \in [0.3, 0.8]$ for $b \in \{0, 1\}$, but, crucially, the parameters for $b = 0$ and $b = 1$ can take different values in this interval.

Definition 13. *Given a CrBN $\mathbb{C}_{\mathcal{N}}[\boldsymbol{\mathcal{C}}]$ and ordering σ, the* maximal structural enrichment $\mathbb{C}_{\mathcal{N}^+_\sigma}[\boldsymbol{\mathcal{C}}^+_\sigma]$ *is the (unique) structurally enriched CrBN which has an edge (V_i, V_j) for all $i <_\sigma j$.*

The maximal structural enrichment of a CrBN with some ordering simply allows for the choice of parameters (within the credal set) at some variable to depend on all variables earlier in the order. In the case of the treatment example, the ordering S, R, V, T (used for compilation in Figure 2a) would give a structurally enriched CrBN where the parameter $\boldsymbol{\theta}_{V|s_3}$ is allowed to depend on R, as it does in the CSPN (Figure 2c).

Theorem 3. *The output $\texttt{MAR}_{\max}(\mathcal{S}, \boldsymbol{\mathcal{C}}, e)$ returned by Algorithm 2 using ordering σ satisfies*

$$\texttt{MAR}_{\max}(\mathcal{S}, \boldsymbol{\mathcal{C}}, e) = \texttt{MAR}_{\max}(\mathcal{N}^+_\sigma, \boldsymbol{\mathcal{C}}^+_\sigma, e).$$

An implication of this result (see Appendix for the proof) is that the ordering σ used when compiling the SPN can affect the tightness of the bound. Consequently, it is possible to search over topological orderings to obtain a better bound, at the cost of additional computation; we exploit this in our experiments as the method $\textbf{CUB}_{max}$. It also demonstrates

that if we do, in fact, want to bound the probability of an event in a maximally ordered enrichment, then Algorithm 2 will give an exact result.

We can also make use of this result to lower bound $\texttt{MAR}_{\max}$. We can *project* the optimal parameters $\boldsymbol{\theta}^{+}_{V_i|\boldsymbol{u}_i,\boldsymbol{w}_i}$ found for $\mathbb{C}_{\mathcal{N}^{+}_{\sigma}}[\boldsymbol{\mathcal{C}}^{+}_{\sigma}]$ to obtain parameters $\boldsymbol{\theta}_{V_i|u_i} = \boldsymbol{\theta}^{+}_{V_i|\boldsymbol{u}_i,\boldsymbol{w}^{*}_i}$ valid for $\mathbb{C}_{\mathcal{N}}[\boldsymbol{\mathcal{C}}]$, by fixing some w^{*}_i for each credal set. It is not guaranteed that the exact solution for $\mathbb{C}_{\mathcal{N}}[\boldsymbol{\mathcal{C}}]$ will be such a projection, but it is much easier to search over projections than parameter values and this can provide a strong lower bound in many cases, or serve as a way to initialize a more thorough search algorithm. In our experiments we will evaluate a local greedy search algorithm **CLB**, which is initialized to an arbitrary projection given by some $\boldsymbol{w}_i$ for each credal set $\boldsymbol{\mathcal{C}}_i$, and tries a series of local changes $\boldsymbol{w}_i \rightarrow \boldsymbol{w}'_i$, keeping any that increase the probability. It terminates if it reaches parity with the upper bound or no local improvement can be found. Note that there is no guarantee of convergence to the upper bound – by Theorem 3 it is only possible when $\texttt{MAR}_{\max}(\mathcal{N}^{+}_{\sigma}, \boldsymbol{\mathcal{C}}^{+}_{\sigma}, e) = \texttt{MAR}_{\max}(\mathcal{N}, \boldsymbol{\mathcal{C}}, e)$, and even when this holds **CLB** can get stuck in local optima.

5 Experimental Results

We evaluate our method on the CREPO [Cabañas and Antonucci, 2021] [Huber *et al.*, 2020] credal inference benchmark, which consists of 960 queries over 377 small-to-moderately sized networks, and, to evaluate scalability, hepar2, a 70-node Bayesian network. We include three of our methods[1]: (i) **CUB**, which computes an upper bound; (ii) **CUB**$_{max}$, which searches over ($n = 30$) orderings to obtain a better bound; and (iii) **CLB**, which computes a lower bound as described in Section 4.4 (capped to $n = 100$ steps).

We compare the performance of our methods to exact credal variable elimination [Cozman, 2000] (where feasible) and ApproxLP [Antonucci *et al.*, 2015], an approximate method returning a lower bound which has been shown to be state of the art both in terms of scalability and accuracy of inferences. We do not consider comparison to the IntRob algorithm presented in [Wang *et al.*, 2021] as it cannot address arbitrary credal sets, and Algorithm 2 is equivalent to theirs in the limited cases where both can be applied (when all credal sets are either singletons or maximal).

We split CREPO into two subsets, CREPO-exact (768 queries), where an exact solution could be computed, and CREPO-hard (192 queries), where it ran out of memory (16GB). Since other methods do not support inferences involving decision functions, we use an augmented BN only for hepar2, where both the exact and ApproxLP methods run out of memory even without a decision function.

In Table 2, for all benchmarks we report the time taken by each method. For CREPO-exact, we compute the average difference in computed probability to the exact result (positive/negative for upper/lower bounds respectively), while for the other sets we report the average difference to the best upper bound. Remarkably, we see that our upper-bounding and lower-bounding algorithms dominate ApproxLP on CREPO-exact, with better lower bounds being produced in an order of magnitude less time. Given the simplicity of the greedy iteration in **CLB**, this is primarily explained by the effectiveness of projection from the upper bound as a starting heuristic. On CREPO-hard, our upper bounding is the only method capable of providing guarantees. Meanwhile, our lower bound performs worse on average than ApproxLP, but only by a small amount, while using significantly less time. Finally, we see that our method is the only one to scale to the challenging hepar2 network, completing in a reasonable amount of time even with the significant additional computational expense of incorporating a decision function.

Network		**Exact**	**ApproxLP**	**CLB**	**CUB**	**CUB**$_{max}$
CREPO-exact	Diff	0	-0.0523	-0.0432	0.0018	0.0015
	Time(ms)	626	384	46(6)	2(6)	209(618)
CREPO-hard	Diff	-	-0.0529	-0.0742	0.0220	0
	Time(ms)	-	1154	65(6)	2(6)	231(618)
Hepar2	Diff	-	-	-0.0917	0	-
	Time (s)	-	-	429(287)	4(287)	-

Table 2: Average computation time (compilation time in parenthesis) and difference in computed probability to exact/upper bound. − indicates the method ran out of memory (16GB).

6 Conclusions

We have demonstrated how to construct an SPN whose parameters (sum node weights) can be semantically interpreted as representing specific conditional probability distributions in a CrBN. The result relies on a novel SPN compilation technique, which ensures that (i) all sum nodes correspond to some variable V and (ii) that the values of all parents of V can be uniquely determined. This is significant as (after applying constraint relaxation) it enables a direct mapping of the credal sets of the CrBN to a CSPN, which, unlike the CrBN, can be tractably maximized. This gives an efficient method to analyse robustness of decision functions learnt from data in the presence of imprecise knowledge, distributional shifts and exogenous variables. Our method provides formally guaranteed upper and lower bounds on the probability of an event of interest, and the experimental evaluation has additionally demonstrated that it compares favourably in accuracy to state-of-the-art approximate methods while being orders of magnitude faster.

In future work the upper bound could be improved through reintroducing some of the dropped equality constraints between weights of sum nodes in the CSPN, though this will involve trade-offs between computational challenge and accuracy. The methodology could also be extended to handle more challenging queries such as maximum expectations, by imposing additional structure on the compiled circuit.

Acknowledgements

This project was funded by the ERC under the European Union's Horizon 2020 research and innovation programme (FUN2MODEL, grant agreement No. 834115), and by the Future of Humanity Institute, Oxford University.

[1]Code for algorithms and experiments available at https://github.com/HjalmarWijk/credal-bound

References

[Antonucci *et al.*, 2010] Alessandro Antonucci, Yi Sun, Cassio P. de Campos, and Marco Zaffalon. Generalized loopy 2u: A new algorithm for approximate inference in credal networks. *Int. J. Approx. Reasoning*, 51(5):474–484, jun 2010.

[Antonucci *et al.*, 2015] Alessandro Antonucci, Cassio P. de Campos, David Huber, and Marco Zaffalon. Approximate credal network updating by linear programming with applications to decision making. *Int. J. Approx. Reasoning*, 58:25–38, 2015.

[Cabañas and Antonucci, 2021] Rafael Cabañas and Alessandro Antonucci. Crepo: An open repository to benchmark credal network algorithms. *arXiv preprint arXiv:2105.04158*, 2021.

[Cano *et al.*, 2007] Andrés Cano, Manuel Gómez, Serafín Moral, and Joaquín Abellán. Hill-climbing and branch-and-bound algorithms for exact and approximate inference in credal networks. *Int. J. Approx. Reasoning*, 44(3):261–280, 2007. Reasoning with Imprecise Probabilities.

[Chan and Darwiche, 2004] Hei Chan and Adnan Darwiche. Sensitivity analysis in bayesian networks: From single to multiple parameters. In *UAI*, page 67–75, Arlington, Virginia, USA, 2004. AUAI Press.

[Chan and Darwiche, 2006] Hei Chan and Adnan Darwiche. On the robustness of most probable explanations. In *UAI*, page 63–71, July 2006.

[Coupé *et al.*, 2000] Veerle M. H. Coupé, Linda C. Van Der Gaag, and J. Dik F. Habbema. Sensitivity analysis: An aid for belief-network quantification. *Knowl. Eng. Rev.*, 15(3):215–232, sep 2000.

[Cozman, 2000] Fabio G. Cozman. Credal networks. *Artificial Intelligence*, 120(2):199–233, 2000.

[Darwiche, 2003] Adnan Darwiche. A differential approach to inference in bayesian networks. *Journal of the ACM (JACM)*, 50(3):280–305, 2003.

[De Campos and Cozman, 2007] Cassio Polpo De Campos and Fabio Gagliardi Cozman. Inference in credal networks through integer programming. In *Proceedings of the Fifth International Symposium on Imprecise Probability: Theories andApplications*, 2007.

[Fagiuoli and Zaffalon, 1998] Enrico Fagiuoli and Marco Zaffalon. 2u: An exact interval propagation algorithm for polytrees with binary variables. *Artif. Intell.*, 106(1):77–107, nov 1998.

[Huber *et al.*, 2020] D. Huber, R. Cabañas, A. Antonucci, and M. Zaffalon. CREMA: a Java library for credal network inference. In M. Jaeger and T.D. Nielsen, editors, *Proceedings of the 10th International Conference on Probabilistic Graphical Models (PGM 2020)*, Proceedings of Machine Learning Research, Aalborg, Denmark, 2020. PMLR.

[Katz *et al.*, 2017] Guy Katz, Clark Barrett, David L Dill, Kyle Julian, and Mykel J Kochenderfer. Reluplex: An efficient smt solver for verifying deep neural networks. In *CAV*, pages 97–117. Springer, 2017.

[Lipton *et al.*, 2018] Zachary Lipton, Yu-Xiang Wang, and Alexander Smola. Detecting and correcting for label shift with black box predictors. In *ICML*, pages 3122–3130. PMLR, 2018.

[Mattei *et al.*, 2020] Lilith Mattei, Alessandro Antonucci, Denis Deratani Mauá, Alessandro Facchini, and Julissa Villanueva Llerena. Tractable inference in credal sentential decision diagrams. *International Journal of Approximate Reasoning*, 125:26–48, 2020.

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

[Mauá and Cozman, 2020] Denis Deratani Mauá and Fabio Gagliardi Cozman. Thirty years of credal networks: Specification, algorithms and complexity. *Int. J. Approx. Reasoning*, 126:133–157, 2020.

[Narodytska *et al.*, 2018] Nina Narodytska, Shiva Kasiviswanathan, Leonid Ryzhyk, Mooly Sagiv, and Toby Walsh. Verifying properties of binarized deep neural networks. In *AAAI*, volume 32, 2018.

[Pearl, 1985] Judea Pearl. Bayesian networks: A model of self-activated memory for evidential reasoning. In *Proceedings of the 7th conference of the Cognitive Science Society, University of California, Irvine, CA, USA*, pages 15–17, 1985.

[Poon and Domingos, 2011] Hoifung Poon and Pedro Domingos. Sum-product networks: A new deep architecture. In *UAI*, page 337–346, July 2011.

[Quiñonero-Candela *et al.*, 2009] Joaquin Quiñonero-Candela, Masashi Sugiyama, Neil D Lawrence, and Anton Schwaighofer. *Dataset shift in machine learning*. Mit Press, 2009.

[Rooshenas and Lowd, 2014] Amirmohammad Rooshenas and Daniel Lowd. Learning sum-product networks with direct and indirect variable interactions. In *ICML*, pages 710–718. PMLR, 2014.

[Ruan *et al.*, 2018] Wenjie Ruan, Xiaowei Huang, and Marta Kwiatkowska. Reachability analysis of deep neural networks with provable guarantees. In *IJCAI*, IJCAI'18, page 2651–2659. AAAI Press, 2018.

[Wang *et al.*, 2021] Benjie Wang, Clare Lyle, and Marta Kwiatkowska. Provable guarantees on the robustness of decision rules to causal interventions. In *IJCAI*, 2021.

[Zaffalon *et al.*, 2020] Marco Zaffalon, Alessandro Antonucci, and Rafael Cabañas. Structural causal models are (solvable by) credal networks. In *International Conference on Probabilistic Graphical Models*, pages 581–592. PMLR, 2020.

[Zhang *et al.*, 2015] Kun Zhang, Mingming Gong, and Bernhard Schölkopf. Multi-source domain adaptation: A causal view. In *AAAI*, 2015.

A Proofs

A.1 Proof of Lemma 1

Lemma 1. *Suppose that $\mathcal{T}$ computes $\mathcal{N}$ respecting some topological order. Let t be a $+$-node in $\mathcal{T}$ splitting on some variable V. Then all complete subcircuits α which include t must agree on the value of its parents $\text{pa}_{\mathcal{N}}(V)$.*

Proof. First, we show that the part of the subcircuit "external to" the sub-AC $\mathcal{T}_t$ is sufficient to determine the value of the parents of V.

Suppose that α, α' are two complete subcircuits which both include t and differ only in the chosen $+$-node children in the sub-AC from t. Since $\mathcal{T}$ respects a topological ordering, by Defn. 8 no nodes in $\mathcal{T}_t$ split on any variable in $\text{pa}_{\mathcal{N}}(V)$. Then, these subcircuits must assign the same values to $\text{pa}_{\mathcal{N}}(V)$.

Now for the main result, suppose for contradiction there exist two complete subcircuits α_1, α_2 which both include t and specify different values $\boldsymbol{u}_1, \boldsymbol{u}_2$ for $\text{pa}_{\mathcal{N}}(V)$ through indicators. We will write $\alpha_1 = (\alpha_{1P}, \alpha_{1S})$, where α_{1P} (prefix) is the part of the subcircuit external to $\mathcal{T}_t$, and α_{1S} (suffix) is the part internal to $\mathcal{T}_t$ (similar for α_2). Now we consider constructing a subcircuit $\alpha_2' = (\alpha_{2_P}, \alpha_{1_S})$ which has the prefix of α_2, but suffix of α_1.

Comparing α_1, α_2', we see that they are identical in $\mathcal{T}_t$, meaning that they specify the same value of V (say, v), but they specify different values $\boldsymbol{u}_1, \boldsymbol{u}_2$ of $\text{pa}_{\mathcal{N}}(V)$. Since each subcircuit corresponds to a term of the network/AC polynomial, the subcircuits must contain parameters $\theta_{v|\boldsymbol{u}_1}, \theta_{v|\boldsymbol{u}_2}$ respectively. Since these parameters depend on the value of V, they must appear in $\mathcal{T}_t$. But this is a contradiction as both subcircuits are identical in that sub-AC. □

A.2 Proof of Theorem 3

Theorem 3. *The output $\mathtt{MAR}_{\max}(\mathcal{S}, \boldsymbol{\mathcal{C}}, e)$ returned by Algorithm 2 using ordering σ satisfies*

$$\mathtt{MAR}_{\max}(\mathcal{S}, \boldsymbol{\mathcal{C}}, e) = \mathtt{MAR}_{\max}(\mathcal{N}_\sigma^+, \boldsymbol{\mathcal{C}}_\sigma^+, e).$$

To prove Theorem 3 we will introduce an expansion operation EX on SPNs, which intuitively 'distributes' all products over the sum nodes, so that product nodes only have leaf node children. As we perform the expansion we will use labels on the sum nodes to track their origin in the original SPN.

Definition 14. *Let $\mathcal{S}$ be a SPN respecting the ordering σ, where each sum node is given a unique label. We define the* expanded SPN $\mathtt{EX}(\mathcal{S})$ *to be an SPN constructed as follows:*

- *If the root t is a single leaf node, then $\mathtt{EX}(\mathcal{S}) = \mathcal{S}$;*
- *If the root t is a sum node labelled l with children $t_1, ..., t_n$, then $\mathtt{EX}(\mathcal{S})$ is a sum node also labelled l with children $\mathtt{EX}(\mathcal{S}_{t_1}), ..., \mathtt{EX}(\mathcal{S}_{t_n})$ and the same weights;*
- *If the root is a product node which has only leaf nodes as children, then $\mathtt{EX}(\mathcal{S}) = \mathcal{S}$;*
- *If the root is a product node t with at least one product node child p, then $\mathtt{EX}(\mathcal{S})$ is the result of applying EX to a single product node with all the children of t and p together;*
- *If the root is a product node t with at least one sum node child and no product node children, then let s (labelled l) be the first sum node child in σ, let $s_1, ..., s_n$ be the children of s and let $t_1, ..., t_m$ be the other children of t. Then $\mathtt{EX}(\mathcal{S})$ is a sum node (labelled l) with n children $\mathtt{EX}(p_1), ..., \mathtt{EX}(p_n)$, and weights identical to s. Each SPN p_i has a product node at the root and children $t_1, ..., t_m, s_i$, labelled with their original labels.*

It should be noted that each of these operations performs recursive calls on SPNs with fewer nodes than the original SPN. Thus the recursion will always finish.

Lemma 3. *For any SPN $\mathcal{S}$ respecting ordering σ, the expanded SPN $\mathtt{EX}(\mathcal{S})$ has the same SPN polynomial $l_{\mathcal{S}}[\boldsymbol{\lambda}, \boldsymbol{\Theta}] = l_{\mathtt{EX}(\mathcal{S})}[\boldsymbol{\lambda}, \boldsymbol{\Theta}]$. Further, $\mathtt{EX}(\mathcal{S})$ also respects σ, and all its product nodes have only leaf nodes as children.*

Proof. We will prove this by induction on the SPNs $\mathtt{EX}(\mathcal{S}_t)$, where we proceed in reverse topological order of the nodes t in the SPN (i.e. children before parents).

First, if the root is a leaf node or a product node with only leaf node children then all the statements are trivially true.

If the root is a sum node t labelled l with children $t_1, ..., t_n$, then the SPN polynomial is the weighted sum of the polynomials of the expanded children which are unchanged (by the induction hypothesis), where the weights (parameters) are the same as in the original SPN. Further, the expanded children split on the same variables as before, respecting ordering, so the sum node as a whole does so as well.

If the root is a product node with another product node as a child, then it is enough to observe that (by the associativity of products) moving all children to one product node has no effect on the polynomial or ordering - the remaining result follows from the inductive hypothesis on the combined product node.

It remains to show the result for product nodes with at least one sum node child (and no product node children). Adopting the same notation used in the definition, we will show that the sum node s must split on a variable V_i such that $V_i < V_j$ for all other variables V_j split on somewhere in $\mathcal{S}$. For V_k split on at one of the other sum node children $t_1, ..., t_m$ we have $V_i < V_k$ by construction. For some V_j split on elsewhere any node splitting on it must be a descendant of either s or some t_k, and since $\mathcal{S}$ obeys the ordering condition $V_k < V_j$ which shows the claim by transitivity. This (along with the inductive hypothesis) shows that ordering is respected. The polynomial being the same follows immediately from the distributive law.

The overall procedure only produces sum nodes, leaf nodes, and product nodes with only leaf nodes as children. □

Note that requiring all product nodes to have only leaf node children is a very restrictive condition. Any complete subcircuit of such an SPN must be a single path of sum nodes $t_1, ..., t_n$ followed by a single product node containing all its leaf nodes at the end. We can now prove Theorem 3.

Proof. Given the CrBN $\mathbb{C}_{\mathcal{N}}[\boldsymbol{\mathcal{C}}_{\mathcal{N}}]$ (with maximal ordered enrichment $\mathbb{C}_{\mathcal{N}_\sigma^+}[\boldsymbol{\mathcal{C}}_{\mathcal{N}}^+]$), let $\mathbb{C}_{\mathcal{S}}[\boldsymbol{\mathcal{C}}_{\mathcal{S}}]$ be the CSPN constructed by Algorithm 2 respecting ordering σ.

We now define credal sets $\boldsymbol{\mathcal{C}}^-_{\mathcal{S}}$ and $\boldsymbol{\mathcal{C}}^+_{\mathcal{S}}$ over the weights/parameters of the *expanded* SPN $\mathtt{EX}(\mathcal{S})$. In both, for a sum node s in $\mathtt{EX}(\mathcal{S})$, we assign individual credal sets $\mathcal{C}_i$ to the weights of s from the node with the same label in the original SPN $\mathcal{S}$. However, for $\boldsymbol{\mathcal{C}}^-_{\mathcal{S}}$, we additionally impose the constraint that sum nodes with the same label in $\mathtt{EX}(\mathcal{S})$ must also have the same weights; $\mathbb{C}_{\mathtt{EX}(\mathcal{S})}[\boldsymbol{\mathcal{C}}^-_{\mathcal{S}}]$ is thus not a CSPN (while $\mathbb{C}_{\mathtt{EX}(\mathcal{S})}[\boldsymbol{\mathcal{C}}^+_{\mathcal{S}}]$ is).

Firstly, we show that $\mathtt{MAR}_{\max}(\mathcal{S}, \boldsymbol{\mathcal{C}}_{\mathcal{S}} e) = \mathtt{MAR}_{\max}(\mathtt{EX}(\mathcal{S}), \boldsymbol{\mathcal{C}}^-_{\mathcal{S}}, e)$. Since $\mathcal{S}$ and $\mathtt{EX}(\mathcal{S})$ have the same polynomial, the distributions are equivalent (for the same parameters). Further, since the credal sets $\boldsymbol{\mathcal{C}}_{\mathcal{S}}, \boldsymbol{\mathcal{C}}^-_{\mathcal{S}}$ are the same by construction, the maximal marginal probability (and parameter assignment) is the same for both.

Second, we show that $\mathtt{MAR}_{\max}(\mathtt{EX}(\mathcal{S}), \boldsymbol{\mathcal{C}}^+_{\mathcal{S}}, e) = \mathtt{MAR}_{\max}(\mathtt{EX}(\mathcal{S}), \boldsymbol{\mathcal{C}}^-_{\mathcal{S}}, e)$, that is, that the same value can be obtained despite the additional constraints on $\boldsymbol{\mathcal{C}}^-_{\mathcal{S}}$. Recall that, since $\mathbb{C}_{\mathtt{EX}(\mathcal{S})}[\boldsymbol{\mathcal{C}}^+_{\mathcal{S}}]$ is a CSPN, we can find $\mathtt{MAR}_{\max}(\mathtt{EX}(\mathcal{S}), \boldsymbol{\mathcal{C}}^+_{\mathcal{S}}, e)$ by maximizing the weights at each sum node locally based on the maximized value of its children (2). Looking at the definition of the expansion process (Defn. 14), we see that the only way in which two sum nodes r, r' in $\mathtt{EX}(\mathcal{S})$ can have the same label is if two product nodes in the original SPN have a common sum node child (with that label). Then, using similar notation as in the Definition, r will have children $\mathtt{EX}(p_1), ..., \mathtt{EX}(p_n)$, where p_i is an SPN with a product node root and children $t_1, ..., t_m, s_i$, while r' will have children $\mathtt{EX}(p'_1), ..., \mathtt{EX}(p'_n)$ where p'_i is an SPN with product node root and children $t'_1, ..., t'_{m'}, s_i$. Since applying $\mathtt{EX}$ does not change the SPN polynomial, the i^{th} child of r has polynomial $l_{\mathcal{S}_{p_i}}[\boldsymbol{\lambda}, \Theta] = l_{\mathcal{S}_{s_i}}[\boldsymbol{\lambda}, \Theta] \prod_{j=1}^{m} l_{\mathcal{S}_{t_i}}[\boldsymbol{\lambda}, \Theta]$. Since SPN polynomials are multilinear functions of their parameters, each parameter can appear in only one of the RHS polynomials. Thus we can maximize each separately, i.e. $\max_{\Theta \in \boldsymbol{\mathcal{C}}^+_{\mathcal{S}}} l_{\mathcal{S}_{p_i}}[\boldsymbol{\lambda}, \Theta] = \max_{\Theta \in \boldsymbol{\mathcal{C}}^+_{\mathcal{S}}} l_{\mathcal{S}_{s_i}}[\boldsymbol{\lambda}, \Theta] \prod_{j=1}^{m} \max_{\Theta \in \boldsymbol{\mathcal{C}}^+_{\mathcal{S}}} l_{\mathcal{S}_{t_j}}[\boldsymbol{\lambda}, \Theta] = \max_{\Theta \in \boldsymbol{\mathcal{C}}^+_{\mathcal{S}}} l_{\mathcal{S}_{s_i}}[\boldsymbol{\lambda}, \Theta] \prod_{j=1}^{m} c$ where $c := \prod_{j=1}^{m} \max_{\Theta \in \boldsymbol{\mathcal{C}}^+_{\mathcal{S}}} l_{\mathcal{S}_{t_j}}[\boldsymbol{\lambda}, \Theta]$ is independent of i. Applying a similar argument to the i^{th} child of r', we get $\max_{\Theta \in \boldsymbol{\mathcal{C}}^+_{\mathcal{S}}} l_{\mathcal{S}_{p'_i}}[\boldsymbol{\lambda}, \Theta] = \max_{\Theta \in \boldsymbol{\mathcal{C}}^+_{\mathcal{S}}} l_{\mathcal{S}_{s_i}}[\boldsymbol{\lambda}, \Theta] \prod_{j=1}^{m} c'$ where $c' := \prod_{j=1}^{m'} \max_{\Theta \in \boldsymbol{\mathcal{C}}^+_{\mathcal{S}}} l_{\mathcal{S}_{t'_j}}[\boldsymbol{\lambda}, \Theta]$ is again independent of i. Thus we see that the value of the i^{th} child of r and r' are proportional, and hence the same choice of parameters/weights for r, r' maximize both. Thus we have that the maximizing weight in $\boldsymbol{\mathcal{C}}^+_{\mathcal{S}}$ is also in $\boldsymbol{\mathcal{C}}^-_{\mathcal{S}}$, and $\mathtt{MAR}_{\max}(\mathtt{EX}(\mathcal{S}), \boldsymbol{\mathcal{C}}^+_{\mathcal{S}}, e) = \mathtt{MAR}_{\max}(\mathtt{EX}(\mathcal{S}), \boldsymbol{\mathcal{C}}^-_{\mathcal{S}}, e)$.

Finally, we want to show that $\mathtt{MAR}_{\max}(\mathtt{EX}(\mathcal{S}), \boldsymbol{\mathcal{C}}^+_{\mathcal{S}}, e) = \mathtt{MAR}_{\max}(\mathcal{N}^+_\sigma, \boldsymbol{\mathcal{C}}^+_{\mathcal{N}}, e)$. We have already established that the product nodes in $\mathtt{EX}(\mathcal{S})$ have only leaf node children. Thus, each product node must correspond to a particular instantiation of the variables, and the sum nodes from the root must branch out to cover all of these instantiations. Since $\mathtt{EX}_{\mathcal{S}}$ respects the ordering σ, this means that $\mathtt{EX}(\mathcal{S})$ takes the form of a rooted tree which splits on each variable in succession in the order σ. This also provides a very clear semantics for the weight on each edge of a sum node t: it is simply $\theta_{v_i|u_i}$, where v_i is the value corresponding to that edge of the variable V_i which t is splitting on, and u_i records the values of all variables before V_i in the ordering σ (which is unique as the SPN is a tree). It is then apparent that $\mathtt{EX}(\mathcal{S})$ precisely describes the polynomial of $\mathbb{C}_{\mathcal{N}^+_\sigma}[\boldsymbol{\mathcal{C}}^+_{\mathcal{N}}]$. Further, the credal sets $\boldsymbol{\mathcal{C}}^+_{\mathcal{N}}$ and $\boldsymbol{\mathcal{C}}^+_{\mathcal{S}}$ are the same by construction, so we are done.

All together, we have that $\mathtt{MAR}_{\max}(\mathcal{S}, \boldsymbol{\mathcal{C}}_{\mathcal{S}} e) = \mathtt{MAR}_{\max}(\mathtt{EX}(\mathcal{S}), \boldsymbol{\mathcal{C}}^-_{\mathcal{S}}, e) = \mathtt{MAR}_{\max}(\mathtt{EX}(\mathcal{S}), \boldsymbol{\mathcal{C}}^+_{\mathcal{S}}, e) = \mathtt{MAR}_{\max}(\mathcal{N}^+_\sigma, \boldsymbol{\mathcal{C}}^+_{\mathcal{N}}, e)$, as required. □

B Comparison to Intervention Sets

Bayesian networks are particularly convenient for studying data-generating processes which include some adversary or unknown process that could seemingly arbitrarily *intervene* on the behaviour of certain variables in our system, thus changing the distribution. [Wang *et al.*, 2021] formalize this by defining intervention sets where some subset of variables $\boldsymbol{W} \subseteq V$. In this section, we show how our formulation of credal sets over augmented BNs generalizes their definitions.

[Wang *et al.*, 2021] consider *parametric interventions* on variables $\boldsymbol{W}$ in a BN $\mathcal{N}$ which assign new values to all the parameters associated with variables in $\boldsymbol{W}$. Each such intervention leads to a new BN $\mathcal{N}'$ with a new joint distribution $p_{\mathcal{N}'}$. We define $\mathcal{I}_N[\boldsymbol{W}]$ to be the family of distributions arising from some parametric intervention on $\boldsymbol{W}$ in $\mathcal{N}$. A *structural intervention* on $\boldsymbol{W}$ in $\mathcal{N}$ can further introduce new edges to the graph through a context function $C_{\boldsymbol{W}} : V \to \mathcal{P}(V)$, allowing the variables in $\boldsymbol{W}$ to depend on variables which previously they could not. We define $\mathcal{I}_N[\boldsymbol{W}, C_{\boldsymbol{W}}]$ to be the family of distributions arising from some structural intervention with context function $C_{\boldsymbol{W}}$ on $\boldsymbol{W}$ in $\mathcal{N}$.

B.1 Credal Sets

The interventional families of distributions assume that for each variable we either have complete certainty about its behaviour (if it is not an intervenable variable), or its behaviour is completely unknown (if it is an intervenable variable). This does not allow us to represent other forms of uncertainty, such as that which might arise from learning a model from data. However, these can be represented by credal sets.

Proposition 3. *For any BN $\mathcal{N}$ and subset of the variables $\boldsymbol{W}$, there exists a CrBN $\mathbb{C}_{\mathcal{N}}[\boldsymbol{\mathcal{C}}]$ such that*

$$\mathcal{I}_{\mathcal{N}}[\boldsymbol{W}] = \mathbb{C}_{\mathcal{N}}[\boldsymbol{\mathcal{C}}].$$

Proof. Given $\boldsymbol{W}$, construct a CrBN $\mathbb{C}_{\mathcal{N}}[\boldsymbol{\mathcal{C}}]$ with a credal set such that $\mathcal{C}_{V_i|u_i}$ is maximal (all probability distributions allowed) if $V_i \in \boldsymbol{W}$, and a singleton containing only the parameter value in $\mathcal{N}$ otherwise. For any BN $\mathcal{N}' \in \mathcal{I}_{\mathcal{N}}[\boldsymbol{W}]$ it will differ from $\mathcal{N}$ only for parameters related to variables in $\boldsymbol{W}$. Since these are allowed to take any value by the credal set $\boldsymbol{\mathcal{C}}$, we must have $\mathcal{N}' \in \mathbb{C}_{\mathcal{N}}[\boldsymbol{\mathcal{C}}]$. Likewise, for any $\mathcal{N}' \in \mathbb{C}_{\mathcal{N}}[\boldsymbol{\mathcal{C}}]$ it can differ from $\mathcal{N}$ only for parameters of variables in $\boldsymbol{W}$, so there will be some intervention generating $\mathcal{N}'$. □

This demonstrates that the notion of CrBNs generalizes the notion of parametric interventions. It can also generalize

structural interventions, though we need a CrBN on a structurally enriched graph. Here we reproduce our definition of structural enrichments for CrBNs from the main paper:

Definition 12. *A structural enrichment of a CrBN $\mathbb{C}_{\mathcal{N}}[\boldsymbol{\mathcal{C}}]$ is a new CrBN $\mathbb{C}_{\mathcal{N}'}[\boldsymbol{\mathcal{C}}']$ with a new underlying graph $(\boldsymbol{V}, \boldsymbol{E}')$ such that $\boldsymbol{E} \subseteq \boldsymbol{E}'$, and a new credal set given by*

$$(\forall \boldsymbol{w}_i \in \boldsymbol{W}_i)\mathcal{C}'_{V_i|\boldsymbol{u}_i,\boldsymbol{w}_i} = \mathcal{C}_{V_i|\boldsymbol{u}_i},$$

where $\boldsymbol{U}_i$ are the parents of V_i in $\mathcal{N}$, while $\boldsymbol{W}_i$ are the newly added parents in $\mathcal{N}'$ which were not parents in $\mathcal{N}$.

Proposition 4. *For any BN $\mathcal{N}$, subset of the variables $\boldsymbol{W}$, ordering σ, and context function [Wang* et al.*, 2021] $C_{\boldsymbol{W}}$ whose edges respect σ, there exists a structurally enriched CrBN $\mathbb{C}_{\mathcal{N}}[\boldsymbol{\mathcal{C}}]$ such that*

$$\mathcal{I}'_{\mathcal{N}}[\boldsymbol{W}, C_{\boldsymbol{W}}] = \mathbb{C}_{\mathcal{N}}[\boldsymbol{\mathcal{C}}].$$

Proof. We construct $\mathbb{C}_{\mathcal{N}'}[\boldsymbol{\mathcal{C}}']$ from $\mathcal{N}$ and $\boldsymbol{W}$ as in the proof of Proposition 3. We then construct the structural enrichment $\mathbb{C}_{\mathcal{N}}[\boldsymbol{\mathcal{C}}]$ of $\mathbb{C}_{\mathcal{N}'}[\boldsymbol{\mathcal{C}}']$, where we add all edges given by the context function $C_{\boldsymbol{W}}$ (guaranteed to be possible since the context function is compatible with σ). For $V_i \in \boldsymbol{W}$, after observing any value for the newly added parents, the credal sets for any value of the remaining parents must be maximal (allow any probability distribution) by construction, and so the credal sets must be maximal for all values of all parents. For $V_i \notin \boldsymbol{W}$, after observing any value for the newly added parents the value must be a singleton for all values of the old parents by construction, so it must be a singleton for all values of all parents. As in Proposition 3, we thus have $\mathbb{C}_{\mathcal{N}}[\boldsymbol{\mathcal{C}}] = \mathcal{I}'_{\mathcal{N}}[\boldsymbol{W}, C_{\boldsymbol{W}}]$. □

This demonstrates that CrBNs fully generalize families found by allowing causal interventions of the type in [Wang *et al.*, 2021].

C Experimental Details

C.1 CREPO

The full list of queries and models included in the benchmark is given at https://github.com/IDSIA/crepo. The split into exact and hard is post-hoc and per-query, based on which queries ran out of memory (16GB) when running credal variable elimination.

C.2 hepar2

Since there is to our knowledge no existing credal sets over the hepar2 parameters (other than the ones in [Wang *et al.*, 2021], where our generalized algorithm exactly matches their results, as expected), we create credal sets by allowing perturbations of $\epsilon = 0.1$ to all parameters. The results are averaged over 100 randomized queries; the full list can be found at https://github.com/HjalmarWijk/credal-bound

C.3 Credal Set Representation

As ApproxLP is based on solving linear programming instances, it requires credal sets to be represented as a set of linear constraints. Meanwhile, credal variable elimination traditionally assumes the credal set is specified by a finite set of vertices. Our algorithm is entirely agnostic regarding the credal set representation, but in order to allow comparisons with both of these approaches we use Polco (through CREMA [Huber *et al.*, 2020]) to convert between them.