Updating Credal Networks is Approximable in Polynomial Time

Denis D. Mauá[a,*], Cassio P. de Campos[a], Marco Zaffalon[a]

[a] *Istituto Dalle Molle di Studi sull'Intelligenza Articiale (IDSIA)*
Galleria 2, Manno, 6928 Switzerland

Abstract

Credal networks relax the precise probability requirement of Bayesian networks, enabling a richer representation of uncertainty in the form of closed convex sets of probability measures. The increase in expressiveness comes at the expense of higher computational costs. In this paper, we present a new variable elimination algorithm for exactly computing posterior inferences in extensively specified credal networks, which is empirically shown to outperform a state-of-the-art algorithm. The algorithm is then turned into a provably good approximation scheme, that is, a procedure that for any input is guaranteed to return a solution not worse than the optimum by a given factor. Remarkably, we show that when the networks have bounded treewidth and bounded number of states per variable the approximation algorithm runs in time polynomial in the input size and in the inverse of the error factor, thus being the first known fully polynomial-time approximation scheme for inference in credal networks.

Keywords: Probabilistic graphical models, credal networks, approximation scheme, valuation algebra.

1. Introduction

Credal networks [12] are generalizations of Bayesian networks that allow for a richer representation of uncertainty in the form of set-valued probabilities—in contrast to the sharp numeric values required by their Bayesian counterpart. They are imprecise probabilistic models as advocated by Walley [25].

In a nutshell, a credal network relies on a directed acyclic graph (DAG) to compactly represent a closed convex set of joint probability mass functions over a set of variables, much in the same way that a Bayesian network does for a single joint probability mass function. For instance, the variables in a credal network are assumed to respect the Markov condition: each variable (uniquely represented by a node in the DAG) is independent of its non-descendant non-parents conditional on its parents. Unlike the Bayesian case, however, independence between variables in a credal network can be characterized in more than one way. In this paper, we adopt the concept of *strong independence* [13]. Strong independence is justified by a sensitivity analysis interpretation, where we assume that there exists a single probability mass function (i.e., a Bayesian network) representing our knowledge which we do not know precisely for lack of resources [25].

In order to enable efficient computation, additional constraints need to be imposed on the set-valued specifications of the local probabilities. In this paper, we assume that credal networks are *extensively specified*, meaning that each variable in the model is associated to a set of conditional probability tables. An extensively specified credal network ultimately specifies a set of Bayesian networks over the same DAG.

Inference with credal networks has been theoretically and empirically shown to be a difficult problem. For example, computing bounds for marginal probabilities exactly in credal networks is known to be NP-hard even for singly connected networks,[1] while the analogous inference in Bayesian networks can be performed in polynomial time [7].

Despite the hardness of the problem, several algorithms are known to perform reasonably well under certain conditions. For instance, the 2U algorithm [15] is able to perform exact posterior inference in polynomial time

*Corresponding author

Email addresses: denis@idsia.ch (Denis D. Mauá), cassio@idsia.ch (Cassio P. de Campos), zaffalon@idsia.ch (Marco Zaffalon)

[1] A DAG is singly connected if it contains no loops in the underlying undirected graph. Otherwise it is multi-connected.

algorithm	complexity	restrictions	inference
2U [15]	polynomial	singly connected binary variables	exact
GL2U [3]	polynomial	–	approximate
A/R+ [22]	exponential	singly connected	approximate
MILP [8]	exponential	–	exact/approx.
MLP [6]	exponential	–	exact/approx.
HC [9]	exponential	–	exact/approx.

Table 1: Comparison of some existing algorithms for inference in credal networks.

in singly connected, separately specified credal networks with binary variables. The GL2U algorithm [3], which generalizes 2U to arbitrary networks, runs in polynomial time in multi-connected networks. Unfortunately, GL2U's efficiency might come at the expense of accuracy in the results, since the algorithm does not provide any guarantees on the quality of the solutions it returns. Recently, de Cooman et al. [11] developed an exact polynomial-time algorithm for tree-shaped credal networks that operate under epistemic irrelevance, a different characterization of the Markov condition.

Another notable method, against which we compare the algorithms we devise in this paper, was proposed by de Campos and Cozman [8]. They showed that it is possible to efficiently convert a problem of posterior inference in an arbitrary credal network into a mixed integer linear program, which can then be solved using standard solvers. As with any mixed integer linear program, the solver can be run until a desired accuracy is achieved, and stopped at any moment to produce a solution with known error bounds. In practice, however, it might take a prohibitively long time to achieve a provably good solution, and the bounds returned by the solver in any feasible time might be too loose.

Many other approaches have been proposed that implement a branch-and-bound method with local searches at each step [6, 9, 22]. Table 1 contrasts some of the available algorithms according to time complexity, restrictions on the input and accuracy of inferences. The only two polynomial algorithms are either constrained to a very restricted class of credal networks (the class of singly connected, separately specified credal networks with binary variables) or are arbitrarily inexact (i.e., they can return solutions whose quality is arbitrarily low). The trade-off between time complexity and accuracy that appears to occur is most likely not accidental. It is known that the existence of a polynomial-time algorithm that produces provably good approximations would show that P=NP if the variables are allowed to take on arbitrary number of states, even if we admit only networks of bounded treewidth [7]. Hence, it is a necessary condition to any efficient (provably good) approximation algorithm that the number of states per variable be bounded.

In this paper, we present a new algorithm for computing posterior inferences exactly in extensively specified credal networks (Section 4). The algorithm implements a sophisticated variable elimination scheme in credal networks that mitigates the explosion in the cardinality of propagated sets by discarding Pareto dominated points. We show how the algorithm can be relaxed to provide a provably good approximation scheme, that is, an algorithm that finds a solution whose quality is not worse than the optimum by a factor given as input (Section 5). Remarkably, we show that for networks of bounded treewidth and bounded number of states per variable that algorithm runs in time polynomial in the input and in the inverse of the error factor, and hence constitutes the first fully polynomial-time approximation scheme (FPTAS) for updating credal networks. The FPTAS however is essentially a theoretical result due to very large constants that are hidden in the asymptotical running time analysis.

We begin by stating the basic elements of our algorithm framework (Section 2), followed by a formal definition of inference in extensively specified credal networks under strong independence (Section 3). We then evaluate the empirical performance of both exact and approximation algorithms on a large collection of randomly generated networks (Section 6). The experiments show that our exact algorithm is orders-of-magnitude faster than the mixed integer linear programming approach of de Campos and Cozman, and it is able to solve networks substantially larger. Also, our experiments show that, even though the FPTAS is intended as a theoretical result, it is able to solve problems which the exact method is unable to, and for most cases its performance is comparable to the exact algorithm. We conclude in Section 7.

2. Background

In this section, we introduce the main ingredients of the variable elimination algorithms that we present later on, as well as the basic results needed to guarantee the correctness and efficiency of computations.

2.1. Valuation Algebra

From an algebraic viewpoint, the primitive entities of our formalism are the so-called *labeled valuations* (ϕ, x), which encode information about a (local) domain through a *valuation* ϕ and a set of *variables* x. Here we adopt the equivalent notation ϕ_x to denote the pair (ϕ, x), or simply ϕ when the associated domain x is either not important or clear from the context. Concretely, valuations can take straightforward forms like bounded real-valued functions (Section 2.4), or represent more complicated objects such as sets of pairs of real-valued functions (Section 2.5).

The set of all variables we consider relevant to a problem, denoted by $\mathcal{U}$, is the largest set of variables that can be considered for a (labeled) valuation in our setting, which we assume to be finite. We write variables with capital Latin letters (e.g., $X_1, \ldots, X_n \in \mathcal{U}$) and sets of variables with lowercase Latin letters (e.g., $x = \{X_1, \ldots, X_n\}$). Any variable X is assumed to be associated with a finite set of values Ω_X called its *frame*. The elements $\mathbf{x} \in \Omega_X$ are called states. If x is a nonempty set of variables, Ω_x is given by the Cartesian product of the frames of variables in x, $\Omega_x \triangleq \bigtimes_{X \in x} \Omega_X$. We call Ω_x the frame of x. An element $\mathbf{x}$ in Ω_x is called a configuration and written using boldface lowercase Latin letters (even when x is a singleton). Given $y \neq \emptyset$, $x \neq \emptyset$, and $\mathbf{x} \in \Omega_x$, the notation $\mathbf{x}^{\downarrow y}$ denotes the projection of $\mathbf{x}$ onto $y \subseteq x$, that is, $\mathbf{x}^{\downarrow y}$ is the tuple of elements of $\mathbf{x}$ that are associated with variables in y. By convention, we assume that the empty set contains a single configuration λ in its frame, which is not an element of any other domain. Then, for any configuration $\mathbf{x}$ in the frame of a nonempty set, we have that $\mathbf{x}^{\downarrow \emptyset} = \lambda$.

The set of all valuations ϕ_x over a subset $x \subseteq \mathcal{U}$ is denoted by Φ_x. The set of all valuations is denoted by $\Phi \triangleq \bigcup_{x \subseteq \mathcal{U}} \Phi_x$. The algebra comes with two basic operations of *combination* and *marginalization*. Intuitively, combination represents aggregation of two pieces of information. If ϕ_x and ϕ_y are two arbitrary valuations, then $\phi_x \times \phi_y$ is a valuation $\phi_{x \cup y}$ labeled by set $x \cup y$. Marginalization, on the other hand, acts by coarsening information. If ϕ_x is a valuation and $y \subseteq x$, the *marginal* $\phi_x^{\downarrow y}$ is a valuation labeled by y. It is notationally convenient to define the *elimination* operation, which is in a one-to-one correspondence to marginalization. Formally, if ϕ_x is a valuation then $\phi_x^{-y} \triangleq \phi_x^{\downarrow x \setminus y}$ is the result of the elimination of variables in y from ϕ_x. When clear from the context, we write Y to denote a singleton $y = \{Y\}$, for example $\phi_x^{-Y} = \phi_x^{\downarrow x \setminus \{Y\}}$.

A system $(\Phi, \mathcal{U}, \times, \downarrow)$ closed under combination and marginalization is said to be a *valuation algebra* if it satisfies the following three axioms [20, 23].

(A1) Combination is commutative and associative.

(A2) For $y \subseteq x \subseteq z$, $\left(\phi_z^{\downarrow x}\right)^{\downarrow y} = \phi_z^{\downarrow y}$.

(A3) If $x \subseteq z \subseteq x \cup y$ then $\left(\phi_x \times \phi_y\right)^{\downarrow z} = \phi_x \times \phi_y^{\downarrow z \cap y}$.

The purpose of a valuation algebra is the computation of marginals of the form $(\bigtimes_i \phi_{u_i})^{\downarrow y}$, where the joint valuation $\bigtimes_i \phi_{u_i}$ is computationally too expensive to be obtained explicitly. The complexity of the operations of combination and marginalization is given by the size of the valuations involved, which is in general a function of the cardinality of the domain. Hence, as a rule-of-thumb, the larger the domain of a valuation the more expensive are the operations involving it. Axioms (A1)–(A3) provide the necessary conditions for breaking down the computation of a marginal from a joint valuation into a sequence of computations of marginals over smaller domains. The pseudo-code in Algorithm 1 exhibits the variable elimination procedure (also known as fusion algorithm), which exploits the axioms of valuation algebra to more efficiently compute a marginal of a factorized valuation.

Instead of computing a valuation $\bigtimes_{\phi \in \Psi} \phi$ over a large domain $\Omega_{\mathcal{U}}$ and then marginalizing to y, the algorithm computes marginals $(\bigtimes_{\phi \in \mathcal{B}_i} \phi)^{-X_i}$ over possibly much smaller domains. The overall complexity of the algorithm is given by the size of the largest valuation Ψ^i generated at the loop step. If such a size is bounded then (A1)–(A3) are sufficient to show that the algorithm efficiently outputs the desired marginal [20]. The size of the largest valuation, and hence the complexity of the algorithm, depends on the ordering given as input. Finding an optimal ordering (i.e, one that induces a minimum maximum-size valuation Ψ^i) is an NP-hard problem [24]. Fortunately, there are good heuristics that find suboptimal orderings in time polynomial in the number of variables [19, 14].

Algorithm 1: Variable Elimination

input : A finite set of valuations Ψ, a set of target variables $y \subset \mathcal{U} \triangleq \bigcup_{\phi_u \in \Psi} u$, and an ordering $o = (X_1, \ldots, X_n)$ of the variables in $\mathcal{U} \setminus y$

output: The marginal $(\bigtimes_{\phi \in \Psi} \phi)^{\downarrow y}$

for $i \leftarrow 1$ **to** n **do**

  Set $\mathcal{B}_i \leftarrow \{\phi_u \in \Psi : X_i \in u\}$;

  Compute $\Psi^i \triangleq (\bigtimes_{\phi \in \mathcal{B}_i} \phi)^{-X_i}$;

  Set $\Psi \leftarrow (\Psi \setminus \mathcal{B}_i) \cup \{\Psi^i\}$;

end

return $\Gamma \triangleq \bigtimes_{\phi \in \Psi} \phi$;

2.2. Ordered Valuation Algebra

Some optimization tasks like the credal network inferences we aim at here admit a partial order $\leq$ over the valuations Φ (i.e., a reflexive, antisymmetric and transitive relation). An *ordered valuation algebra* [17, 21] is a system $(\Phi, \mathcal{U}, \times, \downarrow, \leq)$, where $(\Phi, \mathcal{U}, \times, \downarrow)$ is a valuation algebra and $\leq$ is a partial order monotonic with respect to $\times$ and $\downarrow$:[2]

(A4) If $\phi_x \leq \psi_x$ and $\phi_y \leq \psi_y$ then $(\phi_y \times \phi_x) \leq (\psi_y \times \psi_x)$ and $\phi_x^{\downarrow y} \leq \psi_x^{\downarrow y}$.

Given a set of valuations $\Psi \subseteq \Phi$, we say that $\phi \in \Psi$ is *maximal* if for all $\psi \in \Psi$ such that $\phi \leq \psi$ it holds that $\psi \leq \phi$. In other words, a valuation ϕ is maximal in Φ if there is no other valuation greater than it. The operation $\max(\Psi)$ returns the set of maximal valuations of a set Ψ. If Ψ is finite and nonempty, the set $\max(\Psi)$ contains at least one valuation.

Given any relation R on Ψ, a subset $\Psi' \subseteq \Psi$ is an *R-covering* of Ψ if for every $\phi \in \Psi$ there is $\psi \in \Psi'$ such that $\phi R \psi$. For example, the set $\max(\Psi)$ is a $\leq$-covering of Ψ.

2.3. Set-Valuations

The algorithms we develop use the more sophisticated entities of sets of valuations, called *set-valuations*. Theses entities can nevertheless be cast in the algebra of valuations, and manipulated by the variable elimination algorithm to produce sets of marginal valuations.

Let $2_f^{\Phi_x}$ denote the set of all *finite* subsets of Φ_x, that is, $2_f^{\Phi_x}$ contains all possible finite sets of valuations ϕ_x. Likewise, the set 2_f^{Φ} denotes the set of all finite subsets of valuations in Φ. We call an element $\Psi \in 2_f^{\Phi}$ a set-valuation.

If $\Psi_x \in 2_f^{\Phi_x}$ and $\Psi_y \in 2_f^{\Phi_y}$ are any two set-valuations, we define their set-combination $\otimes$ as the set-valuation resulting from element-wise combination of their elements,

$$\Psi_x \otimes \Psi_y \triangleq \{\phi_x \times \phi_y : \phi_x \in \Psi_x, \phi_y \in \Phi_y\}.$$

Likewise, we define the set-marginalization operation $\Downarrow$ on 2^{Φ} as the element-wise marginalization of the valuations in a set,

$$\Psi_x^{\Downarrow y} \triangleq \{\phi_x^{\downarrow y} : \phi_x \in \Psi_x\}.$$

Proposition 1. *The system* $(2_f^{\Phi}, \mathcal{U}, \otimes, \Downarrow)$ *of set-valuations with set-combination and set-marginalization is a valuation algebra.*

Proof. The result follows trivially from element-wise application of axioms (A1)-(A3). □

[2] The ordered valuation algebra we define here is actually weaker than the one defined by Haenni [17], as we do not assume the existence of infima and neutral elements, and it resembles more closely Kohlas and Wilson's definition [21]. Nevertheless, the concrete algebras with which we work can be shown to also satisfy Haenni's definition.

We show in Section 3 that inference in credal networks can be easily mapped into a problem of computing the maximal valuation of a set of marginals. A naive way of doing this is to apply the variable elimination procedure in Algorithm 1 with set valuation and set marginalization to obtain a set of marginals and then find the maximal valuation of its output Γ. The complexity of this method is given by the size of the set-valuations Ψ^i generated in the loop step, which is given by the cardinality of the associated frame of a valuation in it times the cardinality of Ψ^i. This creates set-valuations with cardinality exponential in the cardinality of the input sets.

We can mitigate this problem by "distributing" the max operation over the set-combination and set-marginalization operations, that is, we remove non-maximal elements after each set-combination and set-marginalization performed. This has the potential benefit of reducing the cardinality of set-valuations, and considerably speed up computations. To show that this operation indeed produces the desired outcome (the maximal valuation of a joint set-valuation), we introduce the concept of *max-combination* and *max-marginalization*, show that max is a homomorphism of set-valuation algebra to max-valuation algebra, and finally show that maximal set-valuations with max-valuation and max-marginalization form a valuation algebra.

Let $\max(2^\Phi_f) \triangleq \{\max(\Psi) : \Psi \in 2^\Phi_f\}$ denote the set of sets of maximal valuations in 2^Φ_f with respect to a partial order $\leq$ over Φ. The max-combination $\oplus$ of set-valuations Ψ_x and Ψ_y is the set of maximal valuations of their set-combination:

$$\Psi_x \oplus \Psi_y \triangleq \max(\Psi_x \otimes \Psi_y)\,.$$

Similarly, the max-marginalization of Ψ_x to y is defined as

$$\Psi_x^{\downarrow\downarrow y} \triangleq \max(\Psi_x^{\Downarrow y})\,.$$

If $(\Phi_1, \mathcal{U}, \times_1, \downarrow_1)$ and $(\Phi_2, \mathcal{U}, \times_2, \downarrow_2)$ are two valuation algebras, we say that a mapping $h : \Phi_1 \to \Phi_2$ is a *homomorphism* if for any $\phi_x, \phi_y \in \Phi_1$ we have that $h(\phi_x) \times_2 h(\phi_y) = h(\phi_x \times_1 \phi_y)$ and $h(\phi_x)^{\downarrow_2 y} = h(\phi_x^{\downarrow_1 y})$. Thus, if we are interested in computing $h(\phi_1^{\downarrow_1 y})$ for some valuation $\phi_1 \in \Phi_1$ that we know that factorizes as $\phi_1 = \psi_1 \times_1 \cdots \times_1 \psi_m$, we can equivalently obtain $(h(\psi_1) \times_2 \cdots \times_2 h(\psi_m))^{\downarrow_2 y}$, which might be computationally more convenient. The following result relates the algebras of set-valuations and maximal set-valuations (w.r.t. a partial order).

Lemma 2. max *is a homomorphism from* $(2^\Phi_f, \mathcal{U}, \otimes, \Downarrow)$ *to* $(\max(2^\Phi_f), \mathcal{U}, \oplus, \downarrow\downarrow)$.

Proof. We need to show that for any two set-valuations Ψ_x and Ψ_y, and any set of variables z it follows that

(i) $\max(\Psi_x) \oplus \max(\Psi_y) = \max(\Psi_x \otimes \Psi_y)$,

(ii) $\max(\Psi_x)^{\downarrow\downarrow z} = \max(\Psi_x^{\Downarrow z})$.

Part (i) has been shown in [16, Lemma 1(iv)]. We use a similar argument to prove part (ii).

If Ψ_x is empty, the result follows trivially. Assume Ψ_x is non-empty. We first show that $\max(\Psi_x^{\Downarrow z}) \subseteq \max(\Psi_x)^{\downarrow\downarrow z}$; note that $\max(\Psi_x)^{\downarrow\downarrow z} = \max(\max(\Psi_x)^{\Downarrow z})$ by definition. Suppose the assumption is false and there is an element $\phi_x^{\downarrow z} \in \max(\Psi_x^{\Downarrow z})$, where $\phi_x \in \Psi_x$, which is not an element of $\max(\Psi_x)^{\downarrow\downarrow z}$. By definition of $\max(\Psi_x)$, there is a $\psi_x \in \max(\Psi_x)$ such that $\phi_x \leq \psi_x$. Hence, (A4) implies $\phi_x^{\downarrow z} \leq \psi_x^{\downarrow z}$, and because $\psi_x^{\downarrow z} \in \Psi_x^{\Downarrow z}$ it follows that $\phi_x^{\downarrow z} = \psi_x^{\downarrow z}$, and therefore $\phi_x^{\downarrow z} \in \max(\Psi_x)^{\Downarrow z}$. Since we assumed that $\phi_x^{\downarrow z} \notin \max(\Psi_x)^{\downarrow\downarrow z}$, there must be a $\phi_z \in \max(\Psi_x)^{\downarrow\downarrow z}$ such that $\phi_x^{\downarrow z} \leq \phi_z$. But this contradicts our initial assumption that $\phi_x^{\downarrow z} \in \max(\Psi_x^{\Downarrow z})$, as $\phi_z \in \Psi_x^{\Downarrow z}$.

Let us now show that $\max(\Psi_x^{\Downarrow z}) \supseteq \max(\Psi_x)^{\downarrow\downarrow z}$. Assume to show a contradiction that there is a $\psi_z \in \max(\Psi_x)^{\downarrow\downarrow z} \setminus \max(\Psi_x^{\Downarrow z})$. Since $\max(\Psi_x^{\Downarrow z}) \subseteq \Psi_x^{\Downarrow z}$ by definition, we have that $\psi_z \in \Psi_x^{\Downarrow z}$. Hence, if ψ_z is not in $\max(\Psi_x^{\Downarrow z})$ as we assumed there must be a $\phi_z \in \max(\Psi_x^{\Downarrow z})$ such that $\psi_z \leq \phi_z$. But we have shown that $\max(\Psi_x^{\Downarrow z}) \subseteq \max(\Psi_x)^{\downarrow\downarrow z}$, so that $\psi_z \leq \phi_z$ implies $\phi_z \leq \psi_z$ (by definition of a partial order) and thus $\psi_z \in \max(\Psi_x^{\Downarrow z})$, a contradiction. □

Proposition 3. *The system* $(\max(2^\Phi_f), \mathcal{U}, \oplus, \downarrow\downarrow)$ *of maximal set valuations with max-combination and max-marginalization is also a valuation algebra.*

Proof. Property (A1): Commutativity of $\oplus$ follows trivially from its definition and the commutativity of $\otimes$. To show that associativity holds, consider three maximal set-valuations Ψ_1, Ψ_2, Ψ_3. Notice that $\Psi_1 = \max(\Psi_1)$, $\Psi_2 = \max(\Psi_2)$ and $\Psi_3 = \max(\Psi_3)$. We have by definition of max-combination that $\Psi_1 \oplus (\Psi_2 \oplus \Psi_3) = \max(\Psi_1) \oplus \max(\Psi_2 \otimes \Psi_3)$. By

Lemma 2, the latter equals $\max(\Psi_1 \otimes (\Psi_2 \otimes \Psi_3))$, which by associativity of set-combination equals $\max((\Psi_1 \otimes \Psi_2) \otimes \Psi_3)$. Finally, by applying Lemma 2 once more, we get to $\max(\Psi_1 \otimes \Psi_2) \oplus \max(\Psi_3) \triangleq (\Psi_1 \oplus \Psi_2) \oplus \Psi_3$.

Property (A2): By definition, for $y \subseteq x \subseteq z$ we have that $(\Psi_z^{\downarrow\downarrow x})^{\downarrow\downarrow y})$ equals $\max(\max(\Psi_z^{\Downarrow x})^{\Downarrow y})$, which by Lemma 2 equals $\max([\Psi_z^{\Downarrow x}]^{\Downarrow y})$, which in turn is equal to $\max(\Psi_z^{\Downarrow y}) \triangleq \Psi_z^{\downarrow\downarrow y}$ by property (A2) for set-marginalization.

Property (A3): Let $x \subseteq z \subseteq x \cup y$. By definition,

$$(\Psi_x \oplus \Psi_y)^{\downarrow\downarrow z} = \max(\max(\Psi_x \otimes \Psi_y)^{\Downarrow z}),$$

which by Lemma 2 equals $\max([\Psi_x \otimes \Psi_y]^{\Downarrow z})$. Since set-valuations with set-marginalization and set-combination form a valuation algebra, it follows that the latter is equal to $\max(\Psi_x \otimes \Psi_y^{\Downarrow z})$, which equals $\max(\Psi_x) \oplus \max(\Psi_y^{\Downarrow z}) \triangleq \Psi_x \oplus \Psi_y^{\downarrow\downarrow z}$ by Lemma 2. □

Since maximal set-valuations with max-marginalization and max-combination form a valuation algebra, we can use Algorithm 1 to obtain maximal marginals; and since the set of maximal elements of a set-valuation is in the worst case as large as the set-valuation itself but often much smaller, working with maximal set-valuations instead of set-valuations it is computationally much more efficient.

2.4. Probability Potentials

We now turn our attention to the concrete valuation algebras that we will use in our framework. We start with the most basic structures of probability potentials, and then proceed to define pairs of potentials, and finally, the elements of our algorithms, sets of (maximal) pairs of potentials.

Probability potentials are perhaps the most common example of a valuation algebra. They generalize (conditional) probability mass functions by dropping the requirement of numbers adding to one. If $x \subseteq \mathcal{U}$ is a nonempty set of variables, we define a *potential* p_x as a mapping from Ω_x to the set of nonnegative reals. A potential $p_\emptyset$ over the empty set is defined as a nonnegative real number. The size of a potential p_x is the cardinality of its domain. Combination of potentials is given by element-wise multiplication: for $\mathbf{z} \in \Omega_{x \cup y}$,

$$(p_x \times p_y)(\mathbf{z}) \triangleq p_x(\mathbf{z}^{\downarrow x}) p_y(\mathbf{z}^{\downarrow y}). \tag{1}$$

Marginalization is defined as the sum of compatible elements. For $\mathbf{y} \in \Omega_y$,

$$p_x^{\downarrow y}(\mathbf{y}) \triangleq \sum_{\mathbf{x} \in \Omega_x : \mathbf{x}^{\downarrow y} = \mathbf{y}} p_x(\mathbf{x}). \tag{2}$$

Note that if $y = \emptyset$, the marginal $p_x^{\downarrow y}$ is a (nonnegative real) number.

We define a partial order over potentials by weak Pareto dominance: given two potentials p_x and q_x over the same set of variables x, we define $p_x \leq q_x$ if $p_x(\mathbf{x}) \leq q_x(\mathbf{x})$ for all $\mathbf{x} \in \Omega_x$. In this case, we say that p_x is dominated by q_x. Note that if p_x and q_x have equal sum (i.e., $\sum_{\mathbf{x} \in \Omega_x} p_x(\mathbf{x}) = \sum_{\mathbf{x} \in \Omega_x} q_x(\mathbf{x})$) then $p_x \not\leq q_x$ and $q_x \not\leq p_x$ (unless $p_x = q_x$). This is the case, for example, of potentials representing (conditional) probability mass functions. Therefore, the identity $\mathcal{P}_x = \max(\mathcal{P}_x)$ holds for any set $\mathcal{P}_x$ of (conditional) probability mass functions. Let $\mathcal{P}$ denote the set of all probability potentials. Probability potentials constitute the first example of an ordered valuation algebra [17]:

Proposition 4. *The system $(\mathcal{P}, \mathcal{U}, \times, \downarrow, \leq)$ is an ordered valuation algebra.*

The approximation algorithm we devise requires a function that groups similar potentials whose error introduced by discarding elements from the group is not greater than a given real number. Mathematically, for a given real $\alpha > 1$, we define an equivalence relation $\equiv_\alpha$ over potentials that identifies potentials that are not more than a factor α of each other: any two potentials p_x and q_x over the same set of variables x are α-equivalent (i.e., $p_x \equiv_\alpha q_x$) if for all $\mathbf{x} \in \Omega_x$ either $p_x(\mathbf{x}) = q_x(\mathbf{x}) = 0$ or $p_x(\mathbf{x})$ and $q_x(\mathbf{x})$ are both positive and $\lfloor \log_\alpha p_x(\mathbf{x}) \rfloor = \lfloor \log_\alpha q_x(\mathbf{x}) \rfloor$. For any set of variables x, the relation $\equiv_\alpha$ partitions the space $\mathcal{P}_x$ of potentials over x so that any two potentials in the same partition do not differ in any dimension by more than a factor α. This is depicted in Figure 1.

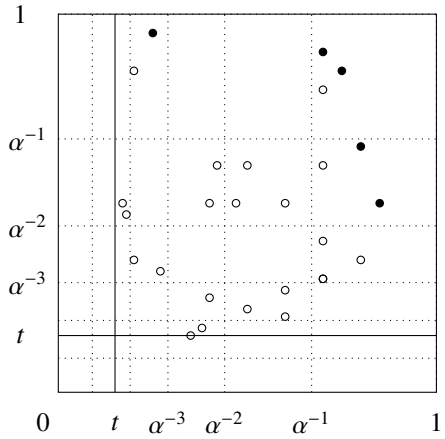


Figure 1: An α-equivalence relation over a set of two-dimensional potentials. Points inside the same rectangle are α-equivalent. Black points indicate maximal elements of the set. t is the smallest value returned by a potential in the set.

2.5. Pairs of Potentials

Inference in credal networks is basically a complicated combinatorial problem with a fractional objective function. This implies (in the case of upper bounds) finding solutions with a good compromise between maximizing the numerator and minimizing the denominator. It is in part this dichotomy in the objective that makes posterior inferences in credal networks much harder than their Bayesian counterpart. We represent the dichotomy in a solution by a pair of potentials: one potential contains the contribution of a (partial) candidate solution to the numerator, while the other potential is the contribution to the denominator. The pairs of potentials corresponding to different contributions of the same (partial) solutions are the primitive entities of the exact and approximate procedures we devise later on.

Let $\phi_x = (p^\ell_x, p^r_x)$ denote a pair of probability potentials associated to the same set of variables x. The potentials p^ℓ_x and p^r_x are referred to as the left and right potentials of ϕ_x, respectively. For any two pairs of potentials $\phi_x = (p^\ell_x, p^r_x)$ and $\psi_x = (q^\ell_x, q^r_x)$, we define a partial order $\leq$ such that $\phi_x \leq \psi_x$ if $q^\ell_x \leq p^\ell_x$ and $p^r_x \leq q^r_x$. The partial order reflects the nature of computations with credal networks. For a given set of pairs of potentials over the same set of variables, we seek valuations that partly dominate (according to right potentials) other potentials and partly are dominated by them (according to left potentials).

If $\phi_x = (p^\ell_x, p^r_x)$ and $\phi_y = (p^\ell_y, p^r_y)$ are two pairs of potentials, we define their combination as the pair of left and right combinations of potentials, that is, $\phi_x \times \phi_y \triangleq (p^\ell_x \times p^\ell_y, p^r_x \times p^r_y)$. Similarly, the marginalization of a pair $\phi_x = (p^\ell_x, p^r_x)$ is performed on both potentials: $\phi^{\downarrow y}_x \triangleq ((p^\ell_x)^{\downarrow y}, (p^r_x)^{\downarrow y})$.

Proposition 5. *Let Φ denote the set of all pairs of potentials. The system $(\Phi, \mathcal{U}, \times, \downarrow, \leq)$ is an ordered valuation algebra.*

Proof. Properties (A1) to (A3) follow directly from the corresponding properties for probability potentials. To show that Property (A4) holds, consider a set of variables z and pairs of potentials $\phi_x = (p, q)$, $\psi_x = (r, s)$, $\phi_y = (t, u)$, and $\psi_y = (v, w)$ such that $\phi_x \leq \psi_x$ and $\phi_y \leq \psi_y$. We thus have that $vr \leq tp$, $uq \leq ws$, $r^{\downarrow z} \leq p^{\downarrow z}$, and $q^{\downarrow z} \leq s^{\downarrow z}$, which implies $\phi_y \times \phi_x = (tp, uq) \leq (vr, ws) = \psi_y \times \psi_x$ and $\phi^{\downarrow z}_x = (p^{\downarrow z}, q^{\downarrow z}) \leq (r^{\downarrow z}, s^{\downarrow z}) = \psi^{\downarrow z}_x$. □

Let 2^Φ_f and $\max(2^\Phi_f)$ denote, respectively, the set of all finite sets of pairs of potentials and the set of all finite sets of maximal pairs of potentials (where maximality is taken with respect to the partial order we defined). It follows from Propositions 1 and 3 that the systems $(2^\Phi_f, \mathcal{U}, \otimes, \Downarrow)$ and $(\max(2^\Phi_f), \mathcal{U}, \oplus, \downdownarrows)$ are valuation algebras. Moreover, max is a homomorphism from 2^Φ_f to $\max(2^\Phi_f)$. Thus, given a collection of finite sets of pairs $\Psi_{x_1}, \ldots, \Psi_{x_n}$, we can obtain the set $\max(\Psi_y) \triangleq \max((\bigotimes \Psi_{x_i})^{\Downarrow y})$ of maximal marginal valuations potentially more efficiently by performing computations in the algebra of sets of maximal pairs, that is, by computing $\max((\bigoplus_i \max(\Psi_{x_i}))^{\downdownarrows y})$. Bentley et al. [5]

showed that sets of n uniformly distributed pairs of potentials over a domain Ω_y have, on average, $O((\log n)^{2|\Omega_y|-1})$ maximal elements. Unfortunately, the uniformity assumption does not hold in the computations we perform, as we deal with sets generated by combination and marginalization of others. To our knowledge, it remains to be obtained any bounds or expectations on the size of maximal sets obtained from propagated valuations such as those generated by variable elimination. Note that, as with sets of probability potentials, if Ψ contains only valuations whose left or right potentials specify a probability mass function, then $\Psi = \max(\Psi)$.

The approximation algorithm we devise enables the trade-off between accuracy and speed by relaxing the partial order to an approximate Pareto dominance relation. Given a real number $\alpha > 1$, we define a relation $\preceq_\alpha$ such that $\phi \preceq_\alpha \psi$ denotes that by mistakenly assuming $\phi \leq \psi$ we introduce an error factor no greater than α on each coordinate. More formally, we define $\phi \preceq_\alpha \psi$ if $\phi \leq (\alpha^{-1}, \alpha) \times \psi$. Note that $\preceq_\alpha$ is neither transitive nor antisymmetric, and, more importantly, that we may have $\phi \preceq_\alpha \psi$ for $\phi \not\leq \psi$. This allows $\preceq_\alpha$-coverings to be much smaller than their corresponding $\leq$-coverings (i.e., the set of maximal elements with respect to the partial order).

The α-equivalence relation over potentials can easily be extended to pairs of potentials. Two pairs (p_x^ℓ, p_x^r) and (p_y^ℓ, p_y^r) are α-equivalent if $p_x^\ell \equiv_\alpha p_y^\ell$ and $p_y^r \equiv_\alpha p_y^r$. It is not difficult to see that $\phi \equiv_\alpha \psi$ implies both $\phi \preceq_\alpha \psi$ and $\psi \preceq_\alpha \phi$.

An $\preceq_\alpha$-covering for a set of pairs of potentials Ψ_x provides an approximated version of Ψ_x, one in which for each $\phi_x \in \Psi_x$ we are guaranteed to have a pair ψ_x in the covering such that the left and right potentials of ψ_x and ϕ_x differ in each coordinate by a factor no greater than α. We can easily obtain an $\preceq_\alpha$-covering of Ψ_x of bounded cardinality by discarding one of any two α-equivalent pairs in Ψ_x. We denote this operation by Ψ_x/α, in analogy to the notion of quotient sets. For instance, we can obtain an $\preceq_\alpha$-covering of the set of points in Figure 1 by selecting exactly one point in each rectangle. As the number of equivalency classes (i.e., hyper-rectangles) in a set of pairs of potentials over x is at most $(1 - \lfloor \log_\alpha t \rfloor)^{2|\Omega_x|}$, where t is the smallest value returned by a potential in the set, this procedure is guaranteed to produce $\preceq_\alpha$-coverings whose cardinality is polynomial in t (but exponential in $|\Omega_x|$). Moreover, the smallest value t of a set of pairs of potentials produced by set-combination and set-marginalization (e.g., during variable elimination) is a polynomial in the input number. The approximation algorithm we develop in Section 5 strongly relies on the following results that formalize and extend this idea.

Lemma 6. *If $k_1, \ldots, k_m$ are positive integers and $\Psi_{x_1}, \Psi'_{x_1}, \ldots, \Psi_{x_m}, \Psi'_{x_m}$ are set valuations such that for $i = 1, \ldots, m$ Ψ'_{x_i} is an $\preceq_{\alpha^{k_i}}$-covering for Ψ_{x_i}, then $\Psi'_{x_1} \otimes \cdots \otimes \Psi'_{x_m}$ is a $\preceq_\beta$-covering for $\Psi_{x_1} \otimes \cdots \otimes \Psi_{x_m}$, where $\beta = \alpha^{\sum_{i=1}^m k_i}$.*

Proof. We work by induction on $j = 1, \ldots, m$. The base case for $j = 1$ follows trivially since Ψ'_{x_1} is an $\preceq_{\alpha^{k_1}}$-covering for Ψ_{x_1}. Assume that the result holds for $1 \leq j < m - 1$, and consider a pair $\phi = \phi' \times \phi''$ in $\Psi_{x_1} \otimes \cdots \otimes \Psi_{x_{j+1}}$, where $\phi' \in \Psi_{x_1} \otimes \cdots \otimes \Psi_{x_j}$ and $\phi'' \in \Psi_{x_{j+1}}$. There is $\psi = \psi' \times \psi''$ in $\Psi'_{x_1} \otimes \cdots \otimes \Psi'_{x_{j+1}}$, where $\psi' \in \Psi'_{x_1} \otimes \cdots \otimes \Psi'_{x_j}$ and $\psi'' \in \Psi_{x_{j+1}}$, such that (by assumption) $\phi' \leq (\alpha^{-\sum_{i=1}^j k_i}, \alpha^{\sum_{i=1}^j k_i}) \times \psi'$ and $\phi'' \leq (\alpha^{-k_{j+1}}, \alpha^{k_{j+1}}) \times \psi''$. It follows thus from (A4) that $\phi \leq (\alpha^{-\sum_{i=1}^{j+1} k_i}, \alpha^{\sum_{i=1}^{j+1} k_i}) \times \psi$. □

Let $\Psi_{x_1}, \ldots, \Psi_{x_m}$ denote sets of pairs of potentials which take values on the interval $[0, 1]$, and let b be the number of bits required to encode these sets as a bit string. Also, let $\Psi_y \triangleq (\Psi_{x_1} \otimes \cdots \otimes \Psi_{x_n})^{\Downarrow y}$ and Ψ_y/α be the set obtained by selecting one valuation per α-equivalence class from Ψ_y. The following result shows that the cardinality of Ψ_y/α is polynomial in the approximation factor α and size of the input b.

Proposition 7. *The cardinality of Ψ_y/α is $O((bm\alpha/(\alpha - 1))^{2|\Omega_y|})$.*

Proof. For any potential $\phi_{x_i} = (p, q) \in \Psi_{x_i}$ and $\mathbf{x_i} \in \Omega_{x_i}$ it follows that $p(\mathbf{x_i})$ and $q(\mathbf{x_i})$ are rational numbers that are either zero or greater than or equal to 2^{-b} (otherwise we would need more than b bits to encode it). Let t be the smallest positive number in a pair $\phi_y \in \Psi_y$. By definition of combination and marginalization, we have for some $\mathbf{y} \in \Omega_y$ that

$$t = \sum_{\mathbf{x}:\mathbf{x}^{\downarrow y}=\mathbf{y}} \prod_{i=1}^{n} p_{x_i}(\mathbf{x}^{\downarrow x_i}) \geq \max_{\mathbf{x}:\mathbf{x}^{\downarrow y}=\mathbf{y}} \prod_{i=1}^{n} p_{x_i}(\mathbf{x}^{\downarrow x_i}) \geq 2^{-bn},$$

where the p_{x_i}'s are either the left or the right potential of pairs $\phi_{x_i} \in \Psi_{x_i}$.

For any $(p_y^\ell, p_y^r) \in \Psi_y$ and $\mathbf{y} \in \Omega_y$ the values $\lfloor \log_\alpha p_y^\ell(\mathbf{y}) \rfloor$ and $\lfloor \log_\alpha p_y^r(\mathbf{y}) \rfloor$ are integers between zero and $-\lfloor \log_\alpha t \rfloor$. Hence, by definition of $\equiv_\alpha$ over pairs of potentials, the number of elements of Ψ_y/α is not greater than $(1 - \lfloor \log_\alpha t \rfloor)^{2|\Omega_y|}$

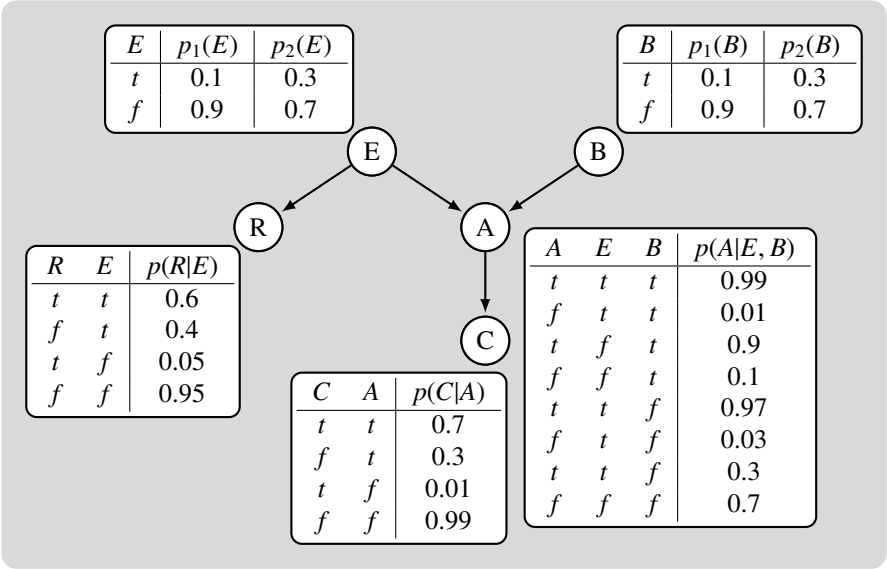


Figure 2: Example of an extensively specified credal network.

(the added one accounts for zero values). Since $t \geq 2^{-bn}$, the number of elements is smaller than $(1 + bn/\log_2 \alpha)^{2|\Omega_y|}$, and it follows from the inequality $\ln(\alpha) \geq (\alpha - 1)/\alpha$, valid for any $\alpha \geq 1$, that $|\Psi_y|$ is $O((bn\alpha/(\alpha - 1))^{2|\Omega_y|})$. □

The latter result is in fact an adaptation of Papadimitriou and Yannakakis' result on the boundedness of ϵ-approximate Pareto curves in multi-objective optimization problems [1, Theorem 1].

3. Credal Networks

In this section we review the basic concepts and computational challenges of extensively specified credal networks.

3.1. Definitions

We start with some basic terminology from graph theory. Let $\mathcal{G} = (\mathcal{U}, \mathcal{E})$ be a DAG, and X a node in $\mathcal{U}$. We write $\text{pa}(X) \triangleq \{Y \in \mathcal{U} : (Y, X) \in \mathcal{E}\}$ to denote the parents of X, $\text{ch}(X) \triangleq \{Y \in \mathcal{U} : (X, Y) \in \mathcal{E}\}$ to denote the children of X in $\mathcal{U}$, and $\text{fa}(X) \triangleq \{X\} \cup \text{pa}(X)$ to denote the family of X. We call Y a descendant of X if there is a directed path from X to Y in $\mathcal{G}$.

In the language of (extensively specified) credal networks, the relationships among variables are quantitatively specified using extensive credal sets. An *extensive credal set* is a set of probability potentials over the same domain. According to Proposition 1, the system $(2^{\mathcal{P}}_f, \mathcal{U}, \otimes, \Downarrow)$, where $2^{\mathcal{P}}_f$ denotes the set of all possible finite extensive credal sets over subsets of $\mathcal{U}$, is a valuation algebra. Given an extensive credal set K_x of potentials over x, we write $\text{H}(K_x)$ to denote its convex hull (i.e., the set obtained by all convex combinations of elements in K_x), and $\text{ext}[\text{H}(K_x)]$ to denote its extreme points (i.e., the elements of $\text{H}(K_x)$ that cannot be written as a convex combination of other elements). The convex hull of a set and the set of its extreme points are themselves extensive credal sets.

An *extensively specified credal network* is a triple $(\mathcal{U}, \mathcal{G}, \mathcal{K})$, where $\mathcal{U}$ is a set of variables, $\mathcal{G}$ is a DAG over $\mathcal{U}$, and $\mathcal{K}$ is a collection of *finite* extensive credal sets K_X, one for each $X \in \mathcal{U}$, such that each potential $p_{\text{fa}(X)} \in K_X$ satisfies $\sum_{\mathbf{x}:\mathbf{x}^{\downarrow\text{pa}(X)}=\pi} p_{\text{fa}(X)}(\mathbf{x}) = 1$ for all $\pi \in \Omega_{\text{pa}(X)}$ (i.e., they represent conditional probability mass functions $p(X|\,\text{pa}(X) = \pi)$). Figure 2 depicts a simple extensively specified credal network over five binary-valued variables. For instance, the extensive credal sets associated to E and C are given, respectively, by $K_E = \{p_1(E), p_2(E)\}$, $K_C = \{p(C|A)\}$.

The *strong extension* of a credal network is the largest set of probability mass functions over $\mathcal{U}$ whose extreme points factorize according to G and $\mathcal{K}$. In other words, the strong extension S is the extensive credal set obtained by

the convex closure of the set-combination of all extensive credal sets in $\mathcal{K}$:

$$S \triangleq \mathrm{H}\left(\bigotimes_{X\in\mathcal{U}} K_X\right). \tag{3}$$

3.2. Belief Updating

When reasoning under uncertainty, we are most often interested in updating our initial model in the light of additional information about some of the variables. In the credal network framework, this is done by computing upper and lower bounds for the posterior probability of a target variable $Q \in \mathcal{U}$ taking on value $\mathbf{q} \in \Omega_Q$ given some evidence $\mathbf{e} \in \Omega_e$ on a set of evidence variables $e \in \mathcal{U} \setminus \{Q\}$. This is known as the *belief updating task*.

Assume $p^{\downarrow e}(\mathbf{e}) > 0$ for all $p \in S$. The belief updating task consists in obtaining the lower and upper bounds:

$$\underline{p}(\mathbf{q}|\mathbf{e}) \triangleq \min_{p\in S} \frac{p^{\downarrow Q,e}(\mathbf{q},\mathbf{e})}{p^{\downarrow e}(\mathbf{e})}, \tag{4}$$

$$\overline{p}(\mathbf{q}|\mathbf{e}) \triangleq \max_{p\in S} \frac{p^{\downarrow Q,e}(\mathbf{q},\mathbf{e})}{p^{\downarrow e}(\mathbf{e})}. \tag{5}$$

The computations of lower and upper bounds in Equations (4) and (5) are continuous optimization problems involving a fraction of two complicated functions. Our goal in the rest of this section is to show that these continuous optimizations can be mapped into combinatorial problems of computing sets of maximal marginal valuations. The latter problems can then be solved by the variable elimination algorithm discussed in Section 2.

We begin with a well-known result that the solutions to the convex optimizations in Equation (5) are attained at extreme points of the strong extension [25]. Hence, the problem can be turned into a combinatorial one of selecting which extreme of the set S maximizes or minimizes the ratio $p^{\downarrow Q,e}(\mathbf{q},\mathbf{e})/p^{\downarrow e}(\mathbf{e})$. Let S' denote the set obtained by combining extremes of local models:

$$S' \triangleq \bigotimes_{X\in\mathcal{U}} \mathrm{ext}[\mathrm{H}(K_X)].$$

Is is also well-known that any extreme of S can be obtained as a combination of local extrema [2], that is,

$$\mathrm{ext}[S] \subseteq S'.$$

These results imply that the potential p^* that maximizes the optimization in Equation (5) (i.e., that p^* achieves $\overline{p}(\mathbf{q}|\mathbf{e})$) is an element of S'. Since S' is, by definition, a subset of S, any solution $p \in S'$ cannot achieve a value higher than $\overline{p}(\mathbf{q}|\mathbf{e})$. Hence, we have that

$$\overline{p}(\mathbf{q}|\mathbf{e}) = \max_{p\in S'} \frac{p^{\downarrow Q,e}(\mathbf{q},\mathbf{e})}{p^{\downarrow e}(\mathbf{e})} \tag{6}$$

$$= \max_{p\in S'} \frac{p^{\downarrow Q,e}(\mathbf{q},\mathbf{e})}{p^{\downarrow Q,e}(\mathbf{q},\mathbf{e}) + p^{\downarrow Q,e}(\neg\mathbf{q},\mathbf{e})}, \tag{7}$$

where $p^{\downarrow Q,e}(\neg\mathbf{q},\mathbf{e}) \triangleq \sum_{\mathbf{q}'\in\Omega_q:\mathbf{q}'\neq\mathbf{q}} p^{\downarrow Q,e}(\mathbf{q}',\mathbf{e})$. We can derive analogous equations for the lower bound optimization. The passage from Equation (6) to (7) follows from the definition of marginalization. Notice that Equation (7) states a combinatorial problem over the set S', which is obtained by a finite number of combinations. If $\underline{p}(\mathbf{q},\mathbf{e}) > 0$, we can divide the numerator and the denominator of Equation (7) by $p^{\downarrow Q,e}(\mathbf{q},\mathbf{e}) > 0$ and obtain[3]

$$\overline{p}(\mathbf{q}|\mathbf{e}) = \max_{p\in S'}\left(1 + \frac{p^{\downarrow Q,e}(\neg\mathbf{q},\mathbf{e})}{p^{\downarrow Q,e}(\mathbf{q},\mathbf{e})}\right)^{-1}. \tag{8}$$

[3]Checking whether $\underline{p}(\mathbf{q},\mathbf{e}) > 0$ can be done efficiently if the network has bounded treewidth.

For any potential $p \in S'$, let $p_{\mathbf{q}|\mathbf{e}}$ denote the posterior probability obtained by p, that is,

$$p_{\mathbf{q}|\mathbf{e}} \triangleq \left(1 + \frac{p^{\downarrow Q,e}(\neg\mathbf{q}, \mathbf{e})}{p^{\downarrow Q,e}(\mathbf{q}, \mathbf{e})}\right)^{-1} .$$

Now consider two potentials p and r such that $p^{\downarrow Q,e}(\neg\mathbf{q}, \mathbf{e}) \leq r^{\downarrow Q,e}(\neg\mathbf{q}, \mathbf{e})$ and $r^{\downarrow Q,e}(\mathbf{q}, \mathbf{e}) \leq p^{\downarrow Q,e}(\mathbf{q}, \mathbf{e})$. Clearly, $r_{\mathbf{q}|\mathbf{e}} \leq p_{\mathbf{q}|\mathbf{e}}$, and r is not a solution of the maximization problem (conversely, p is not a solution of the minimization problem). This allows us to define a partial ordering among solutions $p \in S'$.

Let $\Phi_{\mathbf{q}|\mathbf{e}}$ denote the set of pairs of potentials $(p^{\downarrow Q,e}(\neg\mathbf{q}, \mathbf{e}), p^{\downarrow Q,e}(\mathbf{q}, \mathbf{e}))$, where $p \in S'$. Then Equation (8) can be rewritten as

$$\overline{p}(\mathbf{q}|\mathbf{e}) = \max_{(p^\ell, p^r) \in \max(\Phi_{\mathbf{q}|\mathbf{e}})} (1 + p^\ell / p^r)^{-1} . \tag{9}$$

Basically, what Equation (9) states is that we can narrow down the optimization space to the set of potentials whose corresponding pairs in $\Phi_{\mathbf{q}|\mathbf{e}}$ are not dominated by any other pair in the set (conversely, we take the set of minimal elements in the minimization case). Although this set could be as large as S', our experiments show that most often it is significantly smaller. Thus, if $\max(\Phi_{\mathbf{q}|\mathbf{e}})$ is sufficiently small, we can find the solution by a simple enumerative scheme, and the optimization problem is then converted into the problem of computing the maximal elements of $\Phi_{\mathbf{q}|\mathbf{e}}$, which can be done by the variable elimination, as we show in the following section.

4. Exact Inference

In this section we describe an algorithm for exact computation of upper posterior probabilities in credal networks. An algorithm for obtaining lower probabilities can be obtained in a very similar way.

For any variable X and a subset $\mathcal{X} \subset \Omega_X$, we define the identity potential $I_\mathcal{X}$ as a potential over X that returns 1 for $\mathbf{x} \in \mathcal{X}$ and 0 otherwise. If $\mathcal{X} = \{\mathbf{x}\}$ is a singleton, we write $I_\mathbf{x}$. For any $\mathbf{x} \in \Omega_X$, we define the set $\neg\mathbf{x} \triangleq \Omega_X \setminus \{\mathbf{x}\}$.

Consider a credal network $(\mathcal{U}, \mathcal{G}, \mathcal{K})$, an elimination ordering $X_1, \ldots, X_n$ of the variables in $\mathcal{U}$, a query variable Q with target state $\mathbf{q} \in \Omega_Q$, and a set of evidence variables e set to $\mathbf{e} \in \Omega_e$. An algorithm to compute the upper probability $\overline{p}(\mathbf{q}|\mathbf{e})$ can be obtained as follows. Let Ψ be the set that contains

(i) A set-valuation $\Psi_Q \triangleq \{\phi_Q = (I_{\neg\mathbf{q}}, I_\mathbf{q})\}$;

(ii) A set-valuation $\Psi_X \triangleq \{\phi_{\mathrm{fa}(X)} = (p_{\mathrm{fa}(X)}, p_{\mathrm{fa}(X)}) : p_{\mathrm{fa}(X)} \in \mathrm{ext}[\mathrm{H}(K_X)]\}$ for each $X \in \mathcal{U}$;

(iii) A set-valuation $\Psi_E \triangleq \{(I_{\mathbf{e}^{\downarrow E}}, I_{\mathbf{e}^{\downarrow E}})\}$ for each $E \in e$.

Run variable elimination (Algorithm 1) with inputs Ψ, $y = \emptyset$, using max-combination and max-marginalization to perform the operations, which computes the set of maximal marginals $\Gamma = (\Psi_Q \oplus \bigoplus_{E \in e} \Psi_E \oplus \bigoplus_{X \in \mathcal{U}} \Psi_X)^{\downarrow\downarrow\emptyset}$, and return

$$p_{\mathbf{q}|\mathbf{e}} \triangleq \max_{(p^\ell, p^r) \in \Gamma} \left(1 + p^\ell / p^r\right)^{-1} .$$

Theorem 8. *The described procedure computes the upper posterior probability, that is,* $p_{\mathbf{q}|\mathbf{e}} = \overline{p}(\mathbf{q}|\mathbf{e})$.

Proof. The sets $\Psi_X, \Psi_Q, \Psi_E \in \Psi$ as well as the sets Ψ^i generated during the variable elimination algorithm are valuations in the valuation algebra of sets of maximal pairs of potentials. It follows from (A1)–(A3) and the repeated application of Lemma 2 that

$$\Gamma = \left(\Psi_Q \oplus \bigoplus_{E \in e} \Psi_E \oplus \bigoplus_{X \in \mathcal{U}} \Psi_X\right)^{\downarrow\downarrow\emptyset} \tag{10}$$

$$= \max\left(\left[\Psi_Q \otimes \bigotimes_{E \in e} \Psi_E \otimes \bigotimes_{X \in \mathcal{U}} \Psi_X\right]^{\downarrow\emptyset}\right). \tag{11}$$

Recall that combination of pairs is defined as the pair formed by the combination of left potentials and the combination of right potentials. Therefore, Γ is a set of maximal pairs of potentials (p^ℓ, p^r), where by definition of Ψ_Q, Ψ_E, and Ψ_X,

$$p^\ell = \left(I_{\neg\mathbf{q}} \times \bigtimes_{E\in e} I_{\mathbf{e}^{\downarrow E}} \times \bigtimes_{X\in\mathcal{U}} p_{\mathrm{fa}(X)}\right)^{\downarrow\emptyset} = p^{\downarrow q\cup e}(\neg\mathbf{q}, \mathbf{e})\,, \tag{12}$$

$$p^r = \left(I_{\mathbf{q}} \times \bigtimes_{E_\in e} I_{\mathbf{e}^{\downarrow E}} \times \bigtimes_{X\in\mathcal{U}} p_{\mathrm{fa}(X)}\right)^{\downarrow\emptyset} = p^{\downarrow q\cup e}(\mathbf{q}, \mathbf{e})\,, \tag{13}$$

where $p \in S'$. Moreover, p^ℓ and p^r are compatible, that is, for any potential $p_{\mathrm{fa}(X)}$ in p^ℓ taken from a local extensive credal set K_X, the same was used to compute p^r (and no other potential from K_X). Hence, $\Gamma = \max(\Phi_{\mathbf{q}|\mathbf{e}})$. The result is obtained by comparing the definition of $p_{\mathbf{q}|\mathbf{e}}$ and Equation (9). □

The complexity of the algorithm is upper bounded by the cost of the combination of sets of pairs in computing Ψ^i during the variable elimination part. Each of these computations takes time polynomial in the size of the largest set, which might be exponential in the size of the input sets. For instance, the size of the largest potential is a function of the topology of $\mathcal{G}$ and the given elimination ordering. The number of elements of a set, on the other hand, depends on the number of non-maximal elements that are discarded at each combination or marginalization operation. In the worst-case scenario where no element is ever discarded, the algorithm runs in exponential time even if the network treewidth and the cardinality of the frames of the input sets are bounded (which is not surprising given that the problem is NP-hard even under such simplifying assumptions).

An algorithm for computing lower posterior probabilities can be obtained by substituting sets of maximal valuations and maximizations by sets of minimal valuations and minimizations, respectively. The correctness and complexity analysis are analogous to the maximization case.

5. FPTAS

The computational bottleneck of the variable elimination procedure presented in Section 4 is the existence of large sets at some point in the propagation step (apart from the inherent difficulty of manipulating potentials over large domains). We can remedy the large set problem by trading off accuracy and running time. In this section, we devise a multiplicative approximation scheme that runs in time polynomial in the number of potentials of the input extensive credal sets, but it is still exponential in the size of the largest pair ψ^{X_i} generated during the propagation step, which depends only on the sizes of the frames of the variables and the network treewidth. For the rest of this section, we assume the size of variable frames and the network treewidth to be bounded by a constant. Additionally, we require the input potentials to be represented by rational numbers, so that the size of the input is well-defined. The approximation scheme we obtain is an FPTAS, that is, a family of algorithms parameterized by $\epsilon > 0$ that returns in time polynomial in $1/\epsilon$ and the input size a feasible solution that is no worse than the optimal solution by a factor of $1 + \epsilon$, that is, if x^* is the optimal solution (of a maximization problem), the approximation algorithm returns a solution x such that $x^*/(1 + \epsilon) \leq x \leq x^*$.

Given a real number α greater than one, we define the α-combination of two set-valuations Ψ_x and Ψ_y as the set obtained by (repeatedly) discarding any of two α-equivalent valuations from their set-combination, that is, $\Psi_x \boxtimes_\alpha \Psi_y \triangleq (\Psi_x \otimes \Psi_y)/\alpha$. The operation $\boxtimes_\alpha$ is not associative, that is, there are set-valuations Ψ_x, Ψ_y and Ψ_z for which $(\Psi_x \boxtimes_\alpha \Psi_y) \boxtimes_\alpha \Psi_z$ differs from $\Psi_x \boxtimes_\alpha (\Psi_y \boxtimes_\alpha \Psi_z)$. Nevertheless, the order in which sets are α-combined does not alter the combined approximation factor, as the following result states.

Lemma 9. *If $\Psi_{x_1}, \ldots, \Psi_{x_m}$ are set-valuations, then $\Psi_{x_1} \boxtimes_\alpha \cdots \boxtimes_\alpha \Psi_{x_m}$ (where the operations are applied in any order) is an $\leq_{\alpha^{m-1}}$-covering for $\Psi_{x_1} \otimes \cdots \otimes \Psi_{x_m}$.*

Proof. We show by induction on $k = 1, \ldots, m-1$ that $\Psi_{x_1} \boxtimes_\alpha \cdots \boxtimes_\alpha \Psi_{x_{k+1}}$ is a $\leq_\beta$-covering for $\Psi_{x_1} \otimes \cdots \otimes \Psi_{x_{k+1}}$, where $\beta = \alpha^k$. For $k = 1$, it follows directly from the definition of α-combination that $\Psi_{x_1} \boxtimes_\alpha \Psi_{x_2}$ is an $\leq_\alpha$-covering for $\Psi_{x_1} \otimes \Psi_{x_2}$ (since the former is obtained by removing only α-equivalent elements). Assume for $k \in \{1, \ldots, m-2\}$ that $\Psi_{x_1} \boxtimes_\alpha \cdots \boxtimes_\alpha \Psi_{x_{k+1}}$ is a $\leq_\beta$-covering for $\Psi_{x_1} \otimes \cdots \otimes \Psi_{x_{k+1}}$, where $\beta = \alpha^k$. Consider any pair $\phi = \phi' \times \phi''$ in $\Psi_{x_1} \otimes \cdots \otimes \Psi_{x_{k+2}}$, where $\phi' \in \Psi_{x_1} \otimes \cdots \otimes \Psi_{x_{k+1}}$ and $\phi'' \in \Psi_{x_{k+2}}$. There is $\psi = \psi' \times \psi''$ in $\Psi_{x_1} \boxtimes_\alpha \cdots \boxtimes_\alpha \Psi_{x_{k+1}} \otimes \Psi_{x_{k+2}}$,

where $\psi' \in \Psi_{x_1} \boxtimes_\alpha \cdots \boxtimes_\alpha \Psi_{x_{k+1}}$ and $\psi'' \in \Psi_{x_{k+2}}$, such that $\phi' \leq_\beta \psi'$ (by assumption) and $\psi'' = \phi''$. Then it follows from (A4) that $\phi \leq_\beta \psi$, or equivalently that $\phi \leq (\beta^{-1}, \beta) \times \psi$. But since $\Psi_{x_1} \boxtimes_\alpha \cdots \boxtimes_\alpha \Psi_{x_{k+2}}$ is an $\leq_\alpha$-covering for $\Psi_{x_1} \boxtimes_\alpha \cdots \boxtimes_\alpha \Psi_{x_{k+1}} \otimes \Psi_{x_{k+2}}$, there is $\psi''' \in \Psi_{x_1} \boxtimes_\alpha \cdots \boxtimes_\alpha \Psi_{x_{k+2}}$ such that $\psi \leq_\alpha \psi'''$, or equivalently that $\psi \leq (\alpha^{-1}, \alpha) \times \psi'''$. By combining both sides with (β^{-1}, β) and applying (A4) we get to

$$\phi \leq (\beta^{-1}, \beta) \times \psi \leq (\beta^{-1}, \beta) \times (\alpha^{-1}, \alpha) \times \psi''',$$

and hence $\phi \leq (\alpha^{-(k+1)}, \alpha^{k+1}) \times \psi'''$, and $\phi \leq_{\alpha^{k+1}} \psi'''$. The lemma follows from the induction. □

Thus, by properly choosing the value of α we can obtain a covering that approximates a combination of set-valuations with errors as small as we want. In addition, Proposition 7 guarantees that the sets obtained after each α-combination have cardinality polynomial in the input size and in the maximum error.

We can modify the exact variable elimination algorithm devised in Section 4 to provide an FPTAS by substituting max-combination and max-marginalization with α-combination with $\alpha = 1 + \epsilon/4n$ and set-marginalization. Let Ψ^i and $\Psi^i_{(\alpha)}$ denote, respectively, the sets obtained in the ith iteration of the loop step of variable elimination using set-combination and α-combinations (and both with set-marginalization). In other words, Ψ^i is the set obtained by a brute-force elimination algorithm, whereas $\Psi^i_{(\alpha)}$ denote the sets obtained by the approximation algorithm. Similarly, we let Γ and $\Gamma_{(\alpha)}$ denote the outputs of variable elimination with set-combination and α-combination, respectively.

Let s_1 denote the number of set-valuations that are combined to compute $\Psi^1_{(\alpha)}$ (and also Ψ^1) minus one, that is, $s_1 \triangleq |\mathcal{B}_1| - 1$. Then, for $i = 2, \ldots, n$, we define s_i recursively as $s_i \triangleq |\mathcal{B}_i| - 1 + \sum_{j:\Psi^j_{(\alpha)} \in \mathcal{B}_i} s_j$. Intuitively, s_i denote the number of valuations from the input that are required either directly or indirectly to compute $\Psi^i_{(\alpha)}$ (and also Ψ^i) minus one. Hence, if Ψ is the set obtained after the loop step, we have that $|\Gamma_{(\alpha)}| + \sum_{i:\Psi^i_{(\alpha)} \in \Psi} s_i = n$, since there are n set-valuations given as input and each is used exactly once in the computation of some $\Psi^i_{(\alpha)}$ (or Ψ^i).

The following lemma relates the set-valuations propagated by variable elimination with α-combination to the corresponding sets obtained by set-combination.

Lemma 10. *For $i = 1, \ldots, n$, the set-valuation $\Psi^i_{(\alpha)}$ is an $\leq_{\alpha^{s_i}}$-covering for Ψ^i.*

Proof. For $i = 1$ the result follows directly from Lemma 9. Consider some $1 \leq i < n$, and let $m = |\mathcal{B}_i|$. Without loss of generality, let $\Psi^i = [\Psi_{x_1} \otimes \cdots \otimes \Psi_{x_k} \otimes \cdots \otimes \Psi_{x_m}]^{-X_i}$, where $\Psi_{x_1}, \ldots, \Psi_{x_k}$ denote set-valuations given as input and $\Psi_{x_{k+1}}, \ldots, \Psi_{x_m}$ denote sets Ψ^j ($j < i$) generated in the propagation step. Similarly, let $\Psi^i_{(\alpha)} = [\Psi_{x_1} \boxtimes_\alpha \cdots \boxtimes_\alpha \Psi_{x_k} \boxtimes_\alpha \Psi'_{x_{k+1}} \boxtimes_\alpha \cdots \boxtimes_\alpha \Psi'_{x_m}]^{-X_i}$, where, for $k+1 < \ell < m$, $\Psi_{x_\ell} = \Psi^j$ implies $\Psi'_{x_\ell} = \Psi^j_{(\alpha)}$. Assume by induction that the result holds for $1, \ldots, i-1$. Hence, if $\Psi'_{x_\ell} = \Psi^j_{(\alpha)}$ then Ψ'_{x_ℓ} is an $\leq_{\alpha^{s_j}}$-covering for Ψ_{x_ℓ}. Now, consider any pair $\phi = [\phi' \times \phi'']^{-X_i} \in \Psi^i$, where $\phi' \in \Psi_{x_1} \otimes \cdots \otimes \Psi_{x_k}$ and $\phi'' \in \Psi_{x_{k+1}} \otimes \cdots \otimes \Psi_{x_m}$. It follows from Lemma 9 that there is $\psi' \in \Psi_{x_1} \boxtimes_\alpha \cdots \boxtimes_\alpha \Psi_{x_k}$ such that $\phi' \leq (\alpha^{-k+1}, \alpha^{k-1}) \times \psi'$. Likewise, since $\Psi'_{x_{k+1}} \boxtimes_\alpha \cdots \boxtimes_\alpha \Psi'_{x_m}$ is an $\leq_{\alpha^{m-(k+1)}}$-covering for $\Psi'_{x_{k+1}} \otimes \cdots \otimes \Psi'_{x_m}$ (by Lemma 9) and $\Psi'_{x_{k+1}} \otimes \cdots \otimes \Psi'_{x_m}$ is an $\leq_{\alpha^{\sum_\ell s_\ell}}$-covering for $\Psi_{x_{k+1}} \otimes \cdots \otimes \Psi_{x_m}$ (by Lemma 6 and the induction hypothesis), there is $\psi'' \in \Psi'_{x_{k+1}} \boxtimes_\alpha \cdots \boxtimes_\alpha \Psi'_{x_m}$ such that $\phi'' \leq (\alpha^{-s_i+k}, \alpha^{s_i-k}) \times \psi''$. Since $\equiv_\alpha$ implies $\leq_\alpha$, there is $\psi \in (\Psi_{x_1} \boxtimes_\alpha \cdots \boxtimes_\alpha \Psi_{x_k}) \boxtimes_\alpha (\Psi_{x_{k+1}} \boxtimes_\alpha \cdots \boxtimes_\alpha \Psi_{x_m})$ such that $\psi' \times \psi'' \leq (\alpha^{-1}, \alpha) \times \psi$. Thus, it follows from (A4) that $\phi \leq [(\alpha^{-s_i}, \alpha^{s_i}) \times \psi]^{-X_i}$. But from (A3) we have that $[(\alpha^{-s_i}, \alpha^{s_i}) \times \psi]^{-X_i} = (\alpha^{-s_i}, \alpha^{s_i}) \times \psi^{-X_i}$, where $\psi^{-X_i} \in \Psi^i_{(\alpha)}$. Since this is true for any $\phi \in \Psi^i$, the result holds for i. The lemma follows from the induction. □

Consider a credal network $(\mathcal{U}, \mathcal{G}, \mathcal{K})$, an elimination ordering $X_1, \ldots, X_n$ of the variables in $\mathcal{U}$, a query variable Q, a set of evidence variables e, and a query-evidence pair $(\mathbf{q}, \mathbf{e}) \in \Omega_{\{Q\} \cup e}$. Let Ψ be a collection of sets of pairs as defined in Section 4, and consider the variable elimination algorithm with inputs Ψ, $y = \emptyset$, and α-combination and set-marginalization. Finally, return $p_{\mathbf{q}|\mathbf{e}} \triangleq \max_{(p^\ell, p^r) \in \Gamma_\alpha} (1 + p^\ell / p^r)^{-1}$ as the approximate solution.

Theorem 11. *The described procedure is an FPTAS for computing upper posterior probabilities for networks of bounded treewidth and number of states per variable.*

Proof. First, we analyze the time complexity of the algorithm. We are thus interested in the maximum cardinality of a set $\Psi^i_{(\alpha)}$, and in the cardinality of the domain of a valuation generated in the loop step. The boundedness assumptions imply that the cardinality of the domain of any propagated valuation is smaller than a constant. Hence, the polynomial time complexity depends on $|\Psi^i_{(\alpha)}|$ being bounded. For $i = 1, \ldots, n$, any valuation $\phi^i \in \Psi^i_{(\alpha)}$ is produced by first

combining valuations that are either in some previously generated set $\Psi^j_{(\alpha)}$ ($j < i$) or in a set given as input, and then eliminating X_i from it. Thus, by recursively applying (A1)–(A3) to factorize each valuation from a $\Psi^j_{(\alpha)}$ into a combination of valuations and moving the eliminations out, we have that $\phi^i = [\phi_{x_1} \times \cdots \times \phi_{x_{s_i+1}}]^{-\{X_1,\ldots,X_i\}}$, where each ϕ_{x_j} is in a set-valuation given as input. Hence, each $\Psi^i_{(\alpha)}$ can be factorized as $[\Psi_{x_1} \otimes \cdots \otimes \Psi_{x_{s_i}}]^{-\{X_1,\ldots,X_i\}}$, where each Ψ_{x_j} is a subset of a set-valuation given as input. It follows then from Proposition 7 that $\Psi^i_{(\alpha)}$ has $O([bs_i\alpha/(\alpha-1)]^{2\omega})$ elements, where ω is a constant greater than the cardinality of the domain of any ϕ^i. Since $\alpha = 1 + \epsilon/4n$, we have that

$$O\left(\left[bs_i\frac{\alpha}{(\alpha-1)}\right]^{2\omega}\right) \leq O\left(\left(\frac{4n^2b}{\epsilon}\right)^{2\omega}\right),$$

where b is the size of the input in bits. Therefore the algorithm runs in time polynomial in the input, in the given approximation factor ϵ, and in the number of variables n.

Let $\overline{p}(\mathbf{q}|\mathbf{e}) \triangleq \max_{(p^\ell_*,p^r_*)\in\Gamma}(1 + p^\ell_*/p^r_*)^{-1}$ denote the optimum value. We now show that the approximation algorithm yields a solution such that $p_{\mathbf{q}|\mathbf{e}} \geq \overline{p}(\mathbf{q}|\mathbf{e})/(1+\epsilon)$ for any given positive ϵ. Let $\Psi'_{x_1},\ldots,\Psi'_{x_m}$ denote the sets $\Psi^i_{(\alpha)}$ in Ψ after the loop step of the approximation algorithm, where $m = |\Gamma_{(\alpha)}|$, and let $\Psi_{x_1},\ldots,\Psi_{x_m}$ be the sets Ψ^i in Ψ after the loop step of the brute-force version. Then, $\Gamma_{(\alpha)} = \Psi'_{x_1} \boxtimes_\alpha \cdots \boxtimes_\alpha \Psi'_{x_m}$ and $\Gamma = \Psi_{x_1} \otimes \cdots \otimes \Psi_{x_m}$. It follows from Lemma 9 that $\Gamma_{(\alpha)}$ is an $\leq_{\alpha^{m-1}}$-covering for $\Psi'_{x_1} \otimes \cdots \otimes \Psi'_{x_m}$, which in turn is an $\leq_{\alpha^{n-m}}$-covering for Γ, by Lemma 10. Hence, for any $\phi \in \Gamma$ there is $\psi \in \Gamma_{(\alpha)}$ such that $\phi \leq (\alpha^{-(n-1)}, \alpha^{n-1}) \times \psi$ an thus $\phi \leq (\alpha^{-n}, \alpha^n) \times \psi$. In particular, there is $\psi = (p^\ell, p^r) \in \Gamma_{(\alpha)}$ such that $(p^\ell_*, p^r_*) = \phi^* \leq_{\alpha^n} \psi$. Therefore, $p^\ell \leq \alpha^n p^\ell_*$, $p^r_* \leq \alpha^n p^r$, and

$$\begin{aligned}(1 + p^\ell/p^r)^{-1} &\geq (1 + \alpha^{2n}p^\ell_*/p^r_*)^{-1} \\ &\geq \alpha^{-2n}(1 + p^\ell_*/p^r_*)^{-1}.\end{aligned}$$

Since $\alpha = (1 + \epsilon/4n)$, we have that

$$\begin{aligned}(1 + p^\ell/p^r)^{-1} &\geq (1 + \epsilon/4n)^{-2n}(1 + p^\ell_*/p^r_*)^{-1} \\ &\geq (1+\epsilon)^{-1}(1 + p^\ell_*/p^r_*)^{-1} \\ &= (1+\epsilon)^{-1}\overline{p}(\mathbf{q}|\mathbf{e}),\end{aligned}$$

where the second passage is due to the inequality $(1 + x/z)^z \leq 1 + 2x$, valid for any $x \in [0,1]$ and any positive integer z. Hence, $p_{\mathbf{q}|\mathbf{e}} \geq \overline{p}(\mathbf{q}|\mathbf{e})/(1+\epsilon)$. □

The asymptotical complexity analysis that verifies the efficiency of the FPTAS might hide huge constants, making that the number of elements in a set $\Psi^i_{(\alpha)}$ might be, in practice, prohibitively high. For example, suppose that at some point during variable elimination we generate a $\Psi^i_{(\alpha)}$ whose valuations are associated to a set of five four-state variables y (so that $|\Omega_y| = 4^5$), and such the smallest value in a valuation $\phi_i \in \Psi^i_{(\alpha)}$ is $t = 0.05$. For $\alpha = 1.1$, we can only guarantee that the cardinality of $\Psi^i_{(\alpha)}$ is no greater than $(1 - \log_\alpha t)^{2|\Omega_y|} > 32^{2\cdot 4^5} = 2^{10240}$! The reason why this is still considered polynomially bounded in the input is that the exponent is taken as a constant, since both the number of variable states and the treewidth of the network are bounded, which hides the large cardinality of state spaces. However, computing such a set would be prohibitively expensive with any current computer.

We can make the approximation algorithm more efficient in practice by discarding non-maximal pairs (with respect to the partial order $\leq$) from the sets $\Psi^i_{(\alpha)}$, like in the exact algorithm in Section 4. Since a dominated valuation is guaranteed to induce a smaller probability than the dominating valuation, discarding non-maximal elements does not alter the accuracy of the algorithm, but might decrease the cardinality of sets considerably. This is done in our implementation of the algorithm whose performance we evaluate in the next section.

6. Experiments

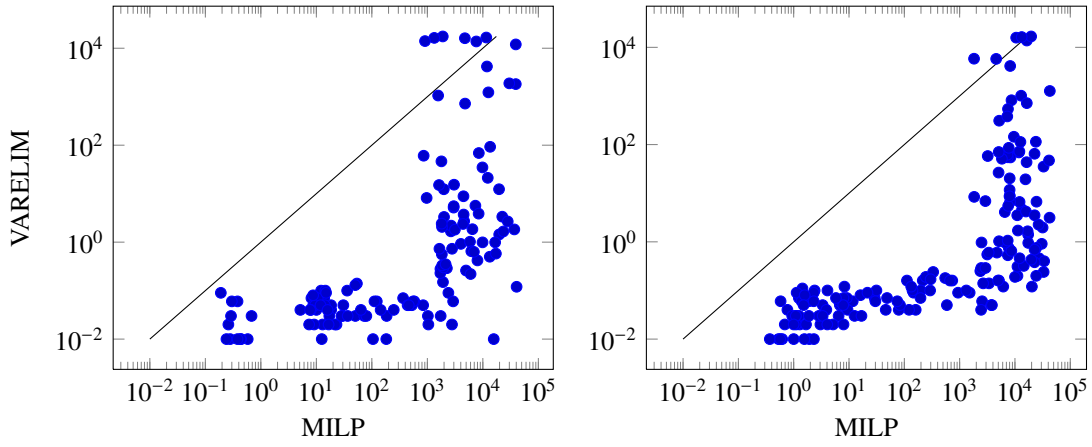
We compared the performance of the exact and the approximation variable elimination algorithms against the mixed linear integer programming method of de Campos and Cozman [8] on a collection of 1860 extensively specified credal networks randomly generated using the BNGen package [18]. The networks were generated containing

treewidth no greater than four, 10 to 30 nodes, 2 to 4 states per variable, and 2 to 16 potentials in each local extensive credal set. For each network, we set some evidence to every leaf node (i.e., the set of evidence variables e correspond to set of leaves of the network) and arbitrarily chose a node with no parents as query. This created belief updating problems where a brute-force approach would have to execute $O(c^n)$ Bayesian network inferences, where c is the maximum cardinality of an extensive credal set in the input and n is the number of variables in the network. The elimination ordering of the variable elimination procedures on each problem was decided using the Maximum Minimum Neighborhood Weight heuristic that greedily attempts to minimize the cardinality of the frame of propagated potentials [14]. The heuristic finds optimum orderings (in the sense of minimum weighted treewidth) for singly connected networks, and it has been found to perform empirically well in multi-connected networks [14, 19]. To make the removal of non-maximal valuations more effective, we ensure the set-valuation Ψ_Q is in $\mathcal{B}_1$, even if it is not required (i.e., if $Q \notin \text{fa}(X_1)$). Since the query has no parents, this increases the treewidth of the elimination ordering by at most one. In all experiments, we used an approximation factor of $\epsilon = 0.1$.

For each network, we granted each algorithm 12 hours of CPU time and 2GB of RAM on a fast computer. Figure 3 depicts the running time of the exact variable elimination procedure (VARELIM) against the running time of the mixed integer linear programming method (MILP) on a log-log scale (plots 3a and 3b), as well as the running time of the approximation against the exact version of the variable elimination procedure (plots 3c and 3d). For each plot, only networks which both methods being compared were able to solve within the time and memory limits are shown. Since the MILP method implements an anytime procedure, we ran it in each problem until the difference between lower and upper bounds were smaller than 0.0001. Hence, we considered a problem unsolved by MILP if the CPLEX mixed linear integer solver was not able to meet such a requirement within the time and memory limits we set. The approximation and exact VARELIM solved, respectively, 805 and 660 out of the 1860 problems, whereas MILP was able to solve only 357 problems. The approximation method solved all instances which the exact solved (but not the opposite). The exact VARELIM was able to solve 346 problems for which MILP failed to find a solution. On the other hand, MILP was able to solve only 43 instances which the exact VARELIM was not able to solve.

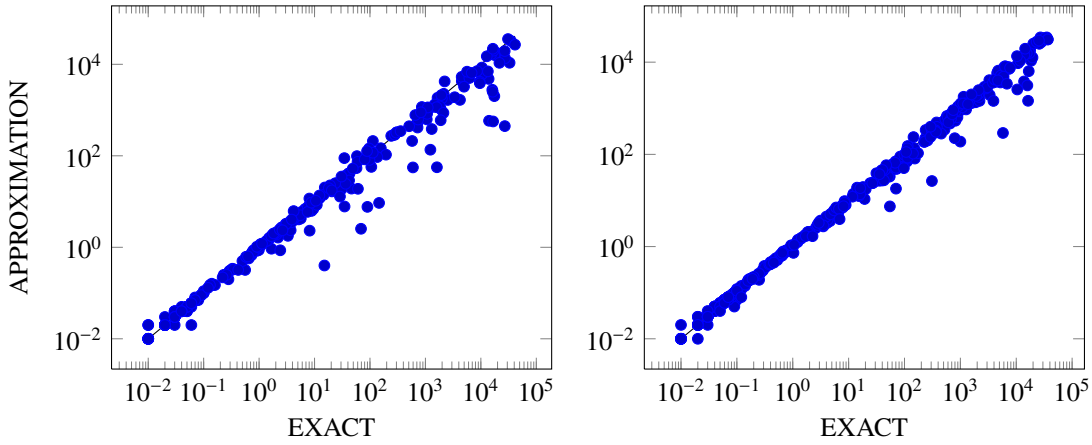
As one can seen from the plots, the exact variable elimination algorithm was much faster than MILP on the large majority of problems (points on the right-hand side of the straight line indicate networks for which the method on the x-axis performed slower than the method on the y-axis). MILP was faster than VARELIM on 10 problems, and the converse was true on 304 instances. Regarding the 660 networks that both approximation and exact variable elimination algorithms were able to solve, the exact was faster than the approximation on 282 cases (approx. 42%). This is due to the overhead in the approximation introduced by finding α-equivalent valuations, and to the fact that the removal of non-maximal elements greatly reduces the size of sets (so that the removal of α-equivalent valuations may cause a small decrease in many problems). On the other hand, in more than half of the cases the gains in speed due to the removal of α-equivalent valuations in the approximation was significant to overcome the overhead of the operation, causing the approximation to outperform the exact version (we also remove dominated valuations in the approximation version). The maximum number of elements in a set produced during variable elimination by the approximation and the exact procedures are compared in Figure 4. We can see that for the vast majority of cases, the approximation produces smaller sets than the exact version. This however need not be the case, as the eye-catching point above the line and close to the center in Figure 4a shows, because the approximation might discard a valuation that otherwise would have caused the removal of many other instances. Nevertheless, we see from the plots that in practice this is rarely the case.

To investigate how the performance of the variable elimination algorithms varies with respect to several important features of the inferential tasks, we group networks according to the topology of the network, the number of nodes N, the maximum number of states per variable k, and the maximum number of potentials in one of the initial sets c. The median, average and standard deviation of running times and maximum set sizes for each group are shown in Tables 2 and 3, respectively. As expected, the running time of the algorithms increases as we increase any of the selected features. The standard deviation of running times for each group is very high, and often higher than the average. A similar phenomenon occurs when we analyze the size of the largest set size generated. This shows that these parameters alone are not sufficient to provide a good estimate of the performance of the algorithms, which are likely very sensitive to the numerical values of the local potentials.



(a) Singly connected networks

(b) Multi-connected networks



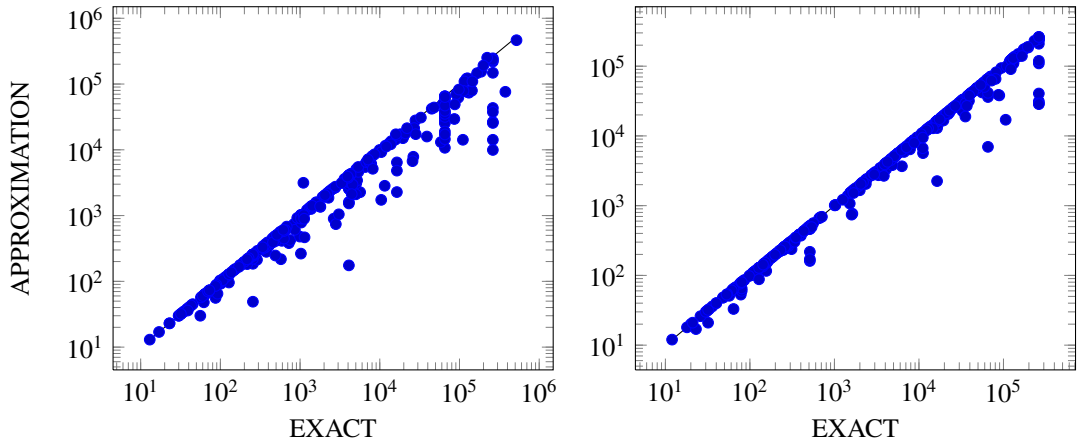
(c) Singly connected networks

(d) Multi-connected networks

Figure 3: Comparison of the log running times (in sec) of the exact and approximation variable elimination procedure and the mixed integer linear programming.

				Exact				Approximation			
Topology	N	k	c	% solved	Median	AVG	SD	% solved	Median	AVG	SD
multi-connected	10	2	16	20	824	5617	9923	21	955	6978	11157
multi-connected	10	2	2	100	0.04	0.04	0.03	100	0.04	0.04	0.03
multi-connected	10	2	4	100	4	1096	3906	100	3	276	1025
multi-connected	10	4	2	100	0.19	0.38	0.46	100	0.2	0.41	0.49
multi-connected	10	4	4	100	248	2030	4407	100	238	1992	4335
multi-connected	20	2	2	95	113	1835	4304	96	95	1592	3747
multi-connected	20	4	2	81	1154	5864	9584	81	1266	6009	9594
multi-connected	30	2	2	26	8560	12170	11710	30	4032	13775	13734
singly connected	10	4	16	10	15428	16877	14159	10	16719	16470	13080
singly connected	10	4	2	100	0.04	0.04	0.03	100	0.04	0.05	0.03
singly connected	10	4	4	100	4	1977	5075	100	4	2095	5476
singly connected	20	4	2	100	39	2055	5097	100	32	1691	4483
singly connected	20	4	4	6	20669	20669	20588	6	13484	13484	13400
singly connected	30	4	2	6	8207	8207	1385	6	5171	5171	1306
tree-shaped	10	4	16	13	1559	1381	687	16	1855	9778	16704
tree-shaped	10	4	2	100	0.04	0.04	0.02	100	0.04	0.05	0.02
tree-shaped	10	4	4	100	6	784	3554	100	6	674	3129
tree-shaped	20	4	2	96	89	2415	6164	96	73	2597	7009

Table 2: Running time (in sec) of variable elimination algorithm grouped by topology type, number of nodes N, maximum number of variable states k and maximum number of potentials in a input set c. Columns show percentage of solved cases, median, average (AVG) and standard deviation (SD) for each group. Numbers greater than one are truncated.



(a) Singly connected networks

(b) Multi-connected networks

Figure 4: Comparison of the log of the maximum set size of the exact and the approximate variable elimination procedures.

				Exact		Approximation	
Topology	N	k	c	AVG	SD	AVG	SD
multi-connected	10	2	16	36046	28928	34579	28563
multi-connected	10	2	2	154	141	130	109
multi-connected	10	2	4	24642	64632	7254	9439
multi-connected	10	4	2	225	128	224	127
multi-connected	10	4	4	46147	65056	42664	55941
multi-connected	20	2	2	37515	61606	28977	46774
multi-connected	20	4	2	67573	73868	66185	73362
multi-connected	30	2	2	93213	55519	81624	57996
singly connected	10	4	16	104468	75687	92784	64183
singly connected	10	4	2	115	100	114	100
singly connected	10	4	4	37155	78008	31361	64117
singly connected	20	4	2	24856	44469	20337	37219
singly connected	20	4	4	76083	68966	58358	51241
singly connected	30	4	2	92744	5476	65708	16654
tree-shaped	10	4	16	11840	9570	11834	9572
tree-shaped	10	4	2	135	108	132	107
tree-shaped	10	4	4	17178	49396	13706	41225
tree-shaped	20	4	2	57055	104187	49044	96469

Table 3: Average (AVG) and standard deviation (SD) of the maximum number of pairs of a set. Numbers are truncated.

7. Conclusion

We derived a new variable-elimination algorithm for exact posterior inference in extensively specified credal networks under strong independence. In our experiments, the algorithm outperforms a previous state-of-the-art method based on integer programming, being able to solve many networks which previous algorithms fail to solve in feasible time. We then showed how to relax the algorithm so as to make it deliver provably good approximations, that is, solutions whose accuracy is within a given factor of the optimum. We showed that for networks of bounded treewidth and bounded number of states per variable, this leads to an algorithm that runs in time polynomial in the input size and in the inverse of the error; or, in other words, to the first fully polynomial-time approximation scheme for inference in credal networks. Such a result is currently mostly theoretical because the asymptotic complexity analysis hides large constants behind the boundedness assumptions. Indeed, our experiments show that the approximation algorithm performs equally to the exact algorithm. Future work is necessary to decrease the hidden constants, making it a more competitive procedure.

It is well known that points that are convex combination of other points can be discarded from the sets generated during variable elimination at any step without affecting optimality [10]. Although this might significantly decrease the size of set-valuations, in our implementation the additional complexity of finding the extrema of a set of points slowed down considerably the algorithms, and increased the overall running time. Nevertheless, we believe that fast algorithms for finding extreme points might be a good addition to both the exact and approximation algorithms devised here. This could be an interesting avenue to explore in the future.

Acknowledgements

This work was supported by the Swiss NSF grants nos. 200020_134759 / 1, 200020_137680 / 1 and 200020_132252, and by the Hasler foundation grant no. 10030.

References

[1] C. Papadimitriou and M. Yannakakis. On the approximability of the trade-offs and optimal access of web sources. In *Proceedings of the Annual Symposium on Foundations of Computer Science*, pp. 86–92, 2000.

[2] A. Antonucci and M. Zaffalon. Decision-theoretic specification of credal networks: A unified language for uncertain modeling with sets of Bayesian networks. *International Journal of Approximate Reasoning* 49(2), pp. 345-361, 2008.

[3] A. Antonucci, Y. Sun, C. P. de Campos and M. Zaffalon. Generalized loopy 2U: a new algorithm for approximate inference in credal networks. *International Journal of Approximate Reasoning* 55(5), pp. 474–484, 2010.

[4] E. Bachoore and H. L. Bodlaender. Weighted treewidth algorithmic techniques and results. In *Proc. of the 18th International Conference on Algorithms and Computation*, pp. 893–903, 2007.

[5] J. L. Bentley, H. T. Kung, M. Schkolnick and C. D. Thompson. On the average number of maxima in a set of vectors and applications. *Journal of the ACM* 25, pp. 536–543, 1978.

[6] C. P. de Campos and F. G. Cozman. Inference in credal networks using multilinear programming. In *Proceedings of the 2nd Starting AI Researchers' Symposium*, pp. 50–61, 2004.

[7] C. P. de Campos and F. G. Cozman. The inferential complexity of Bayesian and credal networks. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, pp. 1313–1318, 2005.

[8] C. P. de Campos and F. G. Cozman. Inference in credal networks through integer programming. In *Proceedings of the 5th International Symposium on Imprecise Probability: Theories and Applications*, pp. 145–154, 2007.

[9] A. Cano, M. Gomez, S. Moral and J. Abellan. Hill-climbing and branch-and-bound algorithms for exact and approximate inference in credal networks. *International Journal of Approximate Reasoning* 44(3), pp. 261–280, 2007.

[10] A. Cano, S. Moral. A review of propagation algorithms for imprecise probabilities. In *Proceedings of the First International Symposium on Imprecise Probabilities and Their Applications*, pp. 51–60, 1999.

[11] G. de Cooman, F. Hermans, A. Antonucci, and M. Zaffalon. Epistemic irrelevance in credal nets: The case of imprecise Markov trees. In *International Journal of Approximate Reasoning* 51(9), pp. 1029–1052, 2010.

[12] F. G. Cozman. Credal networks. *Artificial Intelligence* 120(2), pp. 199–233, 2000.

[13] F. G. Cozman. Graphical models for imprecise probabilities. *International Journal of Approximate Reasoning* 39(2–3), pp. 167–184, 2004.

[14] F. van den Eijkhof and H. L. Bodlaender and M. C. A. Koster. Safe reduction rules for weighted treewidth. *Algorithmica* 47(2), pp. 139–158, 2007.

[15] E. Fagiuoli and M. Zaffalon. 2U: an exact interval propagation algorithm for polytrees with binary variables. *Artificial Intelligence* 106(1), pp. 77–107, 1998.

[16] H. Fargier, E. Rollon and Nic Wilson. Enabling local computation for partially ordered preferences. In *Constraints* 15(4), pp. 516–539, 2010.

[17] R. Haenni. Ordered valuation algebras: a generic framework for approximating inference. *International Journal of Approximate Reasoning* 37(1), pp. 1–41, 2004.

[18] J. S. Ide , F. G. Cozman and F. T. Ramos. Generating random Bayesian networks with constraints on induced width. In *Proceedings of the 16th European Conference on Artificial Intelligence*, pp. 323–327, 2004.
[19] U. Kjaerulff. Triangulation of graphs – Algorithms giving small total state space. Tech. rep. R-90-09, Dept. of Mathematics and Computer System, Aalborg University, Denmark, 1990.
[20] J. Kohlas. Information algebras: generic structures for inference. Springer-Verlag, 2003.
[21] J. Kohlas and N. Wilson. Semiring induced valuation algebras: Exact and approximate local computation algorithms. *Artificial Intelligence* 172(11), pp. 1360-1399, 2008.
[22] J. C. F. da Rocha and F. G. Cozman. Inference in credal networks: branch-and-bound methods and the A/R+ algorithm. *International Journal of Approximate Reasoning* 39(3), pp. 279–296, 2005.
[23] P. Shenoy and G. Shafer. Axioms for probability and belief-function propagation. In *Proceeding of the 4th Conference on Uncertainty in Artificial Intelligence*, pp. 169–198, 1988.
[24] M. Yannakakis. Computing the minimum fill-In is NP-complete. *Journal on Algebraic and Discrete Methods*, 1(2), pp. 77–79, 1981.
[25] P. Walley. Statistical reasoning with imprecise probabilities. Chapman and Hall, 1991.