CREMA: A Java Library for Credal Network Inference

David Huber DAVID@IDSIA.CH

Rafael Cabañas RCABANAS@IDSIA.CH

Alessandro Antonucci ALESSANDRO@IDSIA.CH

Marco Zaffalon ZAFFALON@IDSIA.CH

Istituto Dalle Molle di Studi sull'Intelligenza Artificiale (IDSIA), Lugano, Switzerland

Abstract

We present CREMA (Credal Models Algorithms), a Java library for inference in credal networks. These models are analogous to Bayesian networks, but their local parameters are only constrained to vary in, so-called *credal*, sets. Inference in credal networks is intended as the computation of the bounds of a query with respect to those local variations. For credal networks the task is harder than in Bayesian networks, being NP$^{\mathrm{PP}}$-hard in general models. Yet, scalable approximate algorithms have been shown to provide good accuracies on large or dense models, while exact techniques can be designed to process small or sparse models. CREMA embeds these algorithms and also offers an API to build and query credal networks together with a specification format. This makes CREMA, whose features are discussed and described by a simple example, the most advanced tool for credal network modelling and inference developed so far.

Keywords: Bayesian networks, credal networks; inference algorithms, convex sets.

1. Introduction

Bayesian networks are a popular class of probabilistic graphical models used in AI for both machine learning and knowledge-based decision support (Koller and Friedman, 2009). Many software tools, either commercial and open source, have been developed by companies or academics for Bayesian network modelling. Popular examples are: Agenarisk (`agenarisk.com`), BayesFusion (`bayesfusion.com`), BayesiaLab (`bayesia.com`), Dezide Advisor (`dezide.com`), Hugin Expert (`hugin.com`), Netica (`norsys.com`), and Samiam (`reasoning.cs.ucla.edu/samiam`). These are basically modelling tools embedding inference libraries and typically supporting graphical user interfaces. Besides that, also inference libraries such as Merlin (`github.com/radum2275/merlin`) or Daoopt (`github.com/lotten/daoopt`) are available to compute inferences in large models, as well as tools to process massive data streams (Masegosa et al., 2019)) such as Amidst (`amidsttoolbox.com`).

Besides the graph modelling the dependence relations between its variables, the parameters of a Bayesian network are probability mass functions for single variables conditional to the joint states of the parent variables. Such parameters can be trained from data or elicited by an expert. Yet, in both cases, the quantification might be inaccurate. The task of evaluating whether or not a change/perturbation in these parameters significantly affects the results of an inference is therefore very relevant. This is a major motivation for considering *credal networks* (Cozman, 2000), these being basically Bayesian networks whose conditional probability mass functions have the additional freedom to vary in, so-called *credal*, sets.

Computing a query such as the conditional probability of a variable given some evidence (or a marginal) consists therefore in the evaluation of the bounds of the query with respect to perturbations of the local probability mass functions consistent with the credal sets. This is how inference is intended in credal networks, the task being more complex than Bayesian network inference, namely $\mathrm{NP}^{\mathrm{PP}}$-hard instead of NP-hard (Mauá et al., 2014).

While polynomial-time message-passing algorithms for singly connected topologies are defined for Bayesian nets, the same can be done in credal networks only if all the variables are binary (Fagiuoli and Zaffalon, 1998). For topologies with bounded tree-width, variable elimination techniques allow to compute inferences in polynomial time for Bayesian networks. Something similar can be done for credal networks (Mauá et al., 2012), but without bounds on the complexity when exact inferences are required.

As a consequence of that, and as a matter of fact, the above cited software tools do not support extensive sensitivity analysis for Bayesian networks[1] nor allow to perform inference in explicit credal networks. A notable exception is OpenCossan (`cossan.co.uk`), a tool for uncertainty quantification, offering partial support to credal networks, but only under specific assumptions for the credal sets.

Yet, algorithms designed for general credal networks have been shown to provide good accuracies even with large or dense models (Antonucci et al., 2015), while a credal version of the variable elimination architecture allows to compute exact inferences in small or sparse models. Thus, in order to fill this gap, we developed CREMA[2], an open-source Java library distributed through Maven under LGPL-3.0 License. Let us present CREMA here by means of some illustrative examples. The reader is referred to the online documentation[3] for technical details.

2. The CREMA Architecture

Factors, i.e., non-negative functions of one or more model variables, are the basic components of a Bayesian network or a Markov random field. To model credal networks in CREMA, these objects are extended to the credal case by a set-valued factor specification achieved by linear constraints (H-representation) or, equivalently, by enumeration of the possible sharp specifications (V-representation). Conversions between these two representations are performed by the Polco Java library (SimplifiedBSD) for polyhedral computations (Terzer and Stelling, 2009). A model is consequently a collection of generalized factors linked with the nodes of a directed acyclic graphs by JGraphT (dual LGPL-EPL), a Java library for graph data structures (`jgrapht.org`).

Credal network inferences can be computed in CREMA with the ApproxLP algorithm Antonucci et al. (2015). This consists in a reduction of the optimization required by the exact inference to a sequence of linear programming tasks (solved by the Apache Commons Math3 Solver), whose constraints are the local credal sets. The coefficient of the objective functions in these tasks are inferences in an auxiliary Bayesian network structure. Being a subclass of credal networks, CREMA supports Bayesian network modelling, while inferences in Bayesian networks are computed by dedicated variable elimination tools. Note also that ApproxLP requires a H-representation of the factors and conversion tools for that are implemented in CREMA.

1. In some cases (e.g., Samiam) sensitivity analysis is supported, but only for perturbations on single parameters.
2. `https://github.com/IDSIA/crema`.
3. `http://crema-toolbox.readthedocs.io`.

Alternatively, a variable elimination scheme can be also used for exact inference in credal networks. This requires V-representation. The standard operations of combination and marginalization for the Bayesian case should be iterated over all the possible combinations of the elements in the V-representation. To prevent an exponential blow-up, after any combination, the convex hull (of Polco) is computed in order to prune inner elements of the V-representation, as these are irrelevant for the optimization. Such exact procedure cannot provide guarantees in terms of complexity, and it can be typically used for small or sparse models only.

Regarding inputs, the classical UAI format for Bayesian network specification is supported, together with two extensions of the format supporting credal factors and networks in both V- and H-representations. In the first case, single mass functions are replaced by lists of them, while for H-representation, we use a matrix-array specification of the form $(\hat{A}, \vec{b})$ where the linear constraint $\hat{A} \cdot \vec{p} \leq \vec{b}$ are restricting the possible values of a mass function $\vec{p}$. Normalization and non-negativity constraints are left implicit as they should be satisfied by any credal set.

3. A Demonstrative Example

Consider the credal network over two variables in Figure 1. A V-representation of its local factors is on the right. The corresponding CREMA code snippet is in Figure 2. After the specification of the cardinalities of the two variables and the graph (lines 1-4), the two factors are specified (lines 5-17). A conditional query involved the computation of the bounds of $P(B|A=0)$ is achieved by credal variable elimination (lines 18-20). The same can be done with a H-representation and Figure 3 depicts the CREMA code for the credal set of B as in the top left of Figure 1.

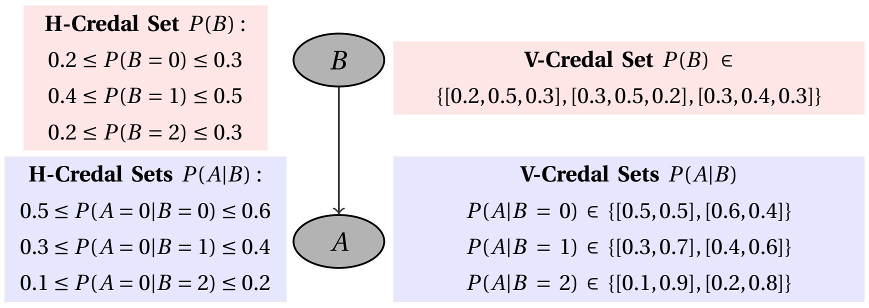
H-Credal Set $P(B)$:
$0.2 \leq P(B=0) \leq 0.3$
$0.4 \leq P(B=1) \leq 0.5$
$0.2 \leq P(B=2) \leq 0.3$
B
V-Credal Set $P(B) \in$
$\{[0.2, 0.5, 0.3], [0.3, 0.5, 0.2], [0.3, 0.4, 0.3]\}$
H-Credal Sets $P(A|B)$:
$0.5 \leq P(A=0|B=0) \leq 0.6$
$0.3 \leq P(A=0|B=1) \leq 0.4$
$0.1 \leq P(A=0|B=2) \leq 0.2$
A
V-Credal Sets $P(A|B)$
$P(A|B=0) \in \{[0.5, 0.5], [0.6, 0.4]\}$
$P(A|B=1) \in \{[0.3, 0.7], [0.4, 0.6]\}$
$P(A|B=2) \in \{[0.1, 0.9], [0.2, 0.8]\}$

Figure 1: A credal network and its V- and H-representations.

4. Conclusions

We presented CREMA, a new Java library for modelling and inference with credal networks. The tool can be used for global sensitivity analysis in Bayesian networks, as well as for dedicated applications of credal networks. To the best of our knowledge, CREMA represents the more advanced tool for these models released so far. Future work concerns the development of learning (both structure and parameters) tools, as well as the support to generalized inference tasks (e.g., marginal MAP). Finally, let us note that CREMA is currently used by the twin project CREDICI,[4] a Java library for causal inference in structural models, build on the top of CREMA.

4. `https://github.com/IDSIA/credici`

```
SparseModel cnet = new SparseModel();
int a = cnet.addVariable(2);
int b = cnet.addVariable(3);
cnet.addParent(a,b);
VertexFactor fb = new VertexFactor(cnet.getDomain(b), Strides.empty());
fb.addVertex(new double[]{0.2, 0.5, 0.3});
fb.addVertex(new double[]{0.3, 0.4, 0.3});
fb.addVertex(new double[]{0.3, 0.2, 0.5});
cnet.setFactor(b,fb);
VertexFactor fa = new VertexFactor(cnet.getDomain(a), cnet.getDomain(b));
fa.addVertex(new double[]{0.5, 0.5}, 0);
fa.addVertex(new double[]{0.6, 0.4}, 0);
fa.addVertex(new double[]{0.3, 0.7}, 1);
fa.addVertex(new double[]{0.4, 0.4}, 1);
fa.addVertex(new double[]{0.2, 0.8}, 2);
fa.addVertex(new double[]{0.1, 0.9}, 2);
cnet.setFactor(a,fa);
Inference inf = new CredalVariableElimination(cnet);
VertexFactor res1 = (VertexFactor) inf.query(b, ObservationBuilder.observe(a,0));
VertexFactor res2 = (VertexFactor) inf.query(b);
```

Figure 2: Credal network modelling (V-representation) and querying in CREMA.

```
SeparateHalfspaceFactor fb = new SeparateHalfspaceFactor(cnet.getDomain(b),
Strides.empty());
fb.addConstraint(new double[]{1,0,0}, Relationship.GEQ, 0.2);
fb.addConstraint(new double[]{1,0,0}, Relationship.LEQ, 0.3);
fb.addConstraint(new double[]{0,1,0}, Relationship.GEQ, 0.4);
fb.addConstraint(new double[]{0,1,0}, Relationship.LEQ, 0.5);
fb.addConstraint(new double[]{0,0,1}, Relationship.GEQ, 0.2);
fb.addConstraint(new double[]{0,0,1}, Relationship.LEQ, 0.3);
cnet.setFactor(b,fb);
```

Figure 3: Credal set modelling (H-representation) in CREMA.

References

A. Antonucci, C. P. de Campos, D. Huber, and M. Zaffalon. Approximate credal network updating by linear programming with applications to decision making. *International Journal of Approximate Reasoning*, 58:25–38, 2015.

F. G. Cozman. Credal networks. *Artificial intelligence*, 120(2):199–233, 2000.

E. Fagiuoli and M. Zaffalon. 2U: an exact interval propagation algorithm for polytrees with binary variables. *Artificial Intelligence*, 106(1):77–107, 1998. doi: 10.1016/S0004-3702(98)00089-7.

D. Koller and N. Friedman. *Probabilistic graphical models: principles and techniques*. MIT, 2009.

A. R. Masegosa, A. M. Martinez, D. Ramos-López, R. Cabañas, A. Salmerón, H. Langseth, T. D. Nielsen, and A. L. Madsen. AMIDST: A Java toolbox for scalable probabilistic machine learning. *Knowledge-Based Systems*, 163:595–597, 2019.

D. D. Mauá, C. P. De Campos, and M. Zaffalon. Updating credal networks is approximable in polynomial time. *International Journal of Approximate Reasoning*, 53(8):1183–1199, 2012.

D. D. Mauá, C. P. De Campos, A. Benavoli, and A. Antonucci. Probabilistic inference in credal networks: new complexity results. *Journal of Artificial Intelligence Research*, 50:603–637, 2014.

M. Terzer and J. Stelling. Parallel extreme ray and pathway computation. In *International Conference on Parallel Processing and Applied Mathematics*, pages 300–309. Springer, 2009.