CREDICI: A Java Library for Causal Inference by Credal Networks

Rafael Cabañas RCABANAS@IDSIA.CH
Alessandro Antonucci ALESSANDRO@IDSIA.CH
David Huber DAVID@IDSIA.CH
Marco Zaffalon ZAFFALON@IDSIA.CH
Istituto Dalle Molle di Studi sull'Intelligenza Artificiale (IDSIA), Lugano, Switzerland

Abstract

We present CREDICI, a Java open-source tool for causal inference based on credal networks. Credal networks are an extension of Bayesian networks where local probability mass functions are only constrained to belong to given, so-called *credal*, sets. CREDICI is based on the recent work of Zaffalon et al. (2020), where an equivalence between Pearl's structural causal models and credal networks has been derived. This allows to reduce a counterfactual query in a causal model to a standard query in a credal network, even in the case of unidentifiable causal effects. The necessary transformations and data structures are implemented in CREDICI, while inferences are eventually computed by CREMA (Huber et al., 2020), a twin library for general credal network inference. Here we discuss the main implementation challenges and possible outlooks.

Keywords: Causal inference; structural causal models; credal networks; counterfactuals.

1. Introduction

Causal analysis is emerging as a major topic for modern data science, with lots of important applications in various domains such as social and biomedical sciences. This makes particularly important the development of stable software tools for causal inference. Notable examples in this field are the popular Python library *DoWhy*[1] developed by Microsoft and based on Pearl's do-calculus, and the R package *bartCause*[2] based on regression trees. Tools of this kind are typically designed to process identifiable queries, while only few libraries such as *causaloptim*[3] can process unidentifiable scenarios with some limitations in terms of scalability and generality.

Zaffalon et al. (2020) recently derived an equivalence relation between Pearl's structural causal models and *credal networks* (Cozman, 2000), which are Bayesian networks whose local parameters have the freedom to vary in, so-called *credal*, sets. The CREDICI library we present here exploits such equivalence to compute (the bounds of) causal inferences even for unidentifiable queries. Given input observational data, a structural causal model is first converted into an equivalent credal network, which is eventually queried by standard inference algorithms for those models. CREDICI is an open-source Java library distributed through Maven under LGPL-3.0 License.[4] The tool is built on the top of the CREMA library[5] (Huber et al., 2020), which contains the inference algorithms and the data structures for credal networks.

1. https://github.com/microsoft/dowhy.
2. https://github.com/vdorie/bartCause.
3. https://github.com/sachsmc/causaloptim
4. https://github.com/IDSIA/credici.
5. https://github.com/IDSIA/crema.

Let us introduce CREDICI here by means of an illustrative example. The reader is referred to the online documentation[6] for technical details.

2. Structural Causal Models and Credal Networks

Consider the directed acyclic graph in Figure 1.a corresponding to the *Party example* of Balke and Pearl (1994). Non-root nodes are associated with observable, *endogenous*, variables. Root nodes are instead associated with latent, *exogenous*, variables determining, together with the other parents, the values of their endogenous children by *structural equations* such as $X_3 = f(X_1, X_2, U_3)$. This is an example of *structural causal model* as in Pearl (2009). If also marginal probabilities of the exogenous variables (e.g., $P(U_1)$) are provided, the model becomes a Bayesian network. This allows to compute causal queries such as $P(X_3|\text{do}(x_2))$, i.e., the probability of X_3 when X_2 is forced by *intervention* to take its state x_2, as standard inference the post-intervention graph (Figure 1.b). Notation $P(X_{3\,x_2}|x_2')$ denotes instead a *counterfactual* scenario where we still evaluate the effect on X_3 of an intervention forcing $X_2 = x_2$, under the evidential information that X_2 was observed in its state $X_2 = x_2' \neq x_2$. Following (Balke and Pearl, 1994), we compute such query in an augmented model called *twin network graph* where the endogenous variables are duplicated. After the necessary post-interventional surgery, the inference can be computed as in Figure 1.c.

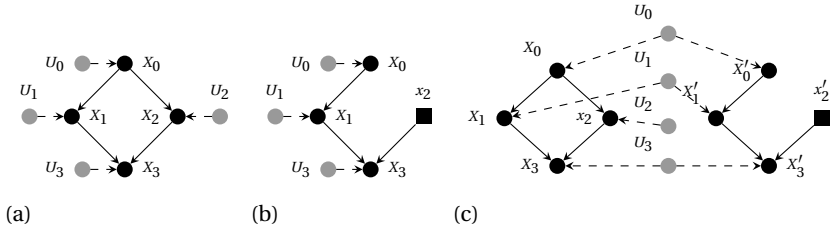

Figure 1: Examples of structural causal diagrams.

Unfortunately, as the exogenous variables are not directly observable, their marginal probabilities are typically unavailable, and causal queries like the ones considered for above example can be computed only in special, so-called *identifiable*, cases. Following Zaffalon et al. (2020), we might obtain (exact) bounds on unidentifiable queries by exploiting the equivalence between structural causal models and *credal networks* (Cozman, 2000). Linear constraints on the values of $P(U)$, for each exogenous node U, can be obtained from the observational data about the endogenous variable. The above discussed post-interventional queries in Bayesian networks can be therefore addressed in the corresponding credal networks even in unidentifiable cases.

3. Using the Application Programming Interface

To illustrate the practical use of CREDICI, let us consider again the Party example. Observational information about the model in the form of a Bayesian network over the endogenous variables is assumed to be available as an input. The code snippet in Figure 2 parses the Bayesian network (the popular UAI format for graphical models is supported) and initialize the corresponding causal model. With the basic setting (line 4), the model graph is built as a Markovian

6. `http://credici.readthedocs.io`.

model, with an exogenous parent for each endogenous model and the states of the exogenous variables enumerating all the possible functional relations between an endogenous variable and its endogenous parents. The class `CausalBuilder` allows more complex settings involving non-Markovian models and more informative structural equations. The resulting causal model is encoded as an object (class `StructuralCausalModel`), from which information about the model can be accessed (e.g., the lists of the different types of variables as in lines 6 and 7).

```
1 //Load the empirical model
2 BayesianNetwork bnet = (BayesianNetwork) IO.read("models/party-empirical.uai");
3 // Build the causal model
4 StructuralCausalModel causalModel = CausalBuilder.of(bnet).build();
5 // Get the endogenous and exogenous variables
6 int[] x = causalModel.getEndogenousVars();
7 int[] u = causalModel.getExogenousVars();
```

Figure 2: Initializing a structural causal model in CREDICI.

The snippet in Figure 3 shows how the methods `toVCredal` and `toHCredal` achieve the credal network conversions, the two variants corresponding, respectively, to credal networks whose credal sets are specified by enumeration of the vertices (line 2) or linear constraints (line 3). Methods to access these credal sets and the local models are shown in lines 4-9.

```
1 // Convert the causal models into credal networks
2 SparseModel vcredal = causalModel.toVCredal(bnet.getFactors());
3 SparseModel hcredal = causalModel.toHCredal(bnet.getFactors());
4 // Access to the equations
5 VertexFactor fx0 = (VertexFactor) vcredal.getFactor(x[0]);
6 SeparateHalfspaceFactor fx0_ = (SeparateHalfspaceFactor) hcredal.getFactor(x[0]);
7 // Access to the credal sets of the exogenous variables
8 VertexFactor pu0 = (VertexFactor) vcredal.getFactor(u[0]);
9 SeparateHalfspaceFactor pu0_ = (SeparateHalfspaceFactor) hcredal.getFactor(u[0]);
```

Figure 3: Converting a causal models in a credal network in CREDICI.

The snippet in Figure 4 demonstrates the methods to eventually compute inferences in the resulting credal network: `CredalCausalVE` is an exact method based on a credal version of variable elimination, while `CredalCausalApproxLP` is an approximate methods based on the linear programming reduction proposed by Antonucci et al. (2015).

```
1 // Exact inference engine
2 CausalInference infExact =
3         new CredalCausalVE(causalModel, bnet.getFactors());
4 // Approximate inference engine
5 CausalInference infApprox =
6         new CredalCausalAproxLP(causalModel, bnet.getFactors());
```

Figure 4: Setting up the causal inference engines in CREDICI.

Finally, Figure 5 shows how the above discussed causal queries are specified in CREDICI. When required, the twin network graph is automatically generated by the methods.

```
// Set up and run a causal query
VertexFactor resExact = (VertexFactor) infExact
        .causalQuery()
        .setTarget(x[3])
        .setIntervention(x[2],1).run();
// Set up an run a counterfactual query
IntervalFactor resApprox = (IntervalFactor) infApprox
        .counterfactualQuery()
        .setTarget(x[3])
        .setIntervention(x[2],1)
        .setEvidence(x[2], 0).run();
```

Figure 5: Causal effects and counterfactual queries in CREDICI.

4. Conclusions

To the best of our knowledge, CREDICI represents the most effective and general tool to compute bounds for unidentifiable queries in general structural causal models. The equivalence between these models and credal networks derived by Zaffalon et al. (2020) allows to obtain the bounds by inference algorithm for credal networks. As an important future work we intend to add support to more sophisticated causal queries such as those involving a measurement bias (Pearl, 2010), and the non-atomic interventions proposed by Correa and Bareinboim (2020).

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.

A. Balke and J. Pearl. Probabilistic evaluation of counterfactual queries. In B. Hayes-Roth and R. E. Korf, editors, *Proceedings of AAAI*, pages 230–237. AAAI Press / The MIT Press, 1994.

J. Correa and E. Bareinboim. A calculus for stochastic interventions: Causal effect identification and surrogate experiments. In *Proceedings of AAAI 2020*, New York, NY, 2020. AAAI Press.

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

D. Huber, R. Cabañas, A. Antonucci, and M. Zaffalon. CREMA: A Java library for credal network inference. In *Proceedings of the tenth International Conference on Probabilistic Graphical Models*, Proceedings of Machine Learning Research, Aalborg, Denmark, 23–25 Sep 2020. PMLR.

J. Pearl. *Causality*. Cambridge University Press, 2009.

J. Pearl. On measurement bias in causal inference. In P. Grünwald and P. Spirtes, editors, *Proceedings of UAI 2010*, pages 425–432. AUAI Press, 2010.

M. Zaffalon, A. Antonucci, and R. Cabañas. Structural causal models are (solvable by) credal networks. In *Proceedings of the tenth International Conference on Probabilistic Graphical Models*, Proceedings of Machine Learning Research, Aalborg, Denmark, 23–25 Sep 2020. PMLR.