Summary of Inference in Probabilistic Answer Set Programs with Imprecise Probabilities via Optimization

Damiano Azzolini[1,*], Fabrizio Riguzzi[2]

[1]Department of Environmental and Prevention Sciences – University of Ferrara
[2]Department of Mathematics and Computer Science – University of Ferrara

Abstract
Credal probabilistic facts and credal annotated disjunctions have been recently introduced in the Probabilistic Answer Set Programming framework to manage imprecise probabilities. In a recent paper, inference within this formalism has been cast as a constrained non-linear optimization problem. In this paper, we review that contribution.

Keywords
Probabilistic Answer Set Programming, Optimization, paper formatting, Imprecise Probabilities

1. Introduction

Probabilistic Answer Set Programming (PASP) combines the effectiveness in solving combinatorial problems of Answer Set Programming with the flexibility in modelling complex distributions of Probabilistic Programming. Recently [1], PASP has been extended with primitives called credal probabilistic facts and credal annotated disjunctions to model imprecise probabilities, i.e., probabilities described by a range, rather than a sharp value. Here, we summarize the paper "Inference in Probabilistic Answer Set Programs with Imprecise Probabilities via Optimization" presented at the UAI 2024 conference [2], where we considered the inference task as a constrained non-linear optimization problem. Empirical results showed the effectiveness of this approach. The paper is structured as follows: Section 2 surveys the background knowledge, Section 3 shows how to cast inference as an optimization problem and discusses the experimental evaluation, and Section 4 concludes the paper.

2. Background

Probabilistic facts [3] are of the form $\pi :: a$ with the meaning that π is the probability associated with the fact a. According to the distribution semantics [4], a *world* is identified by including or not each probabilistic fact in the program. $P(w)$, the probability of a world w, is computed as:

$$P(w) = \prod_{a_i \in w} \pi_i \prod_{a_i \notin w} (1 - \pi_i).$$

A probabilistic answer set program under the credal semantics (PASP) is composed by an answer set program extended with probabilistic facts [5]. The credal semantics describes the probability of a query q, i.e., a conjunction of ground atoms, with a range $[\underline{\mathrm{P}}(q), \overline{\mathrm{P}}(q)]$ where

$$\underline{\mathrm{P}}(q) = \sum_{w_i \mid \forall m \in AS(w_i),\ m \models q} P(w_i),$$
$$\overline{\mathrm{P}}(q) = \sum_{w_i \mid \exists m \in AS(w_i),\ m \models q} P(w_i).$$

AIxIA 2024 Discussion Papers - 23rd International Conference of the Italian Association for Artificial Intelligence, Bolzano, Italy, November 25–28, 2024
*Corresponding author.
damiano.azzolini@unife.it (D. Azzolini); fabrizio.riguzzis@unife.it (F. Riguzzi)

$\underline{\mathrm{P}}(q)$ and $\overline{\mathrm{P}}(q)$ are called, respectively, lower and upper probability. The credal semantics requires that every world has at least one answer set [6]. Inference in these programs can be cast as a Second Level Algebraic Model Counting (2AMC) problem [7]. aspcs is a framework that extends aspmc [8] allowing inference in PASP, that converts the program into a Negation Normal Form (NNF, which is a rooted directed acyclic graph where internal nodes are labeled with AND (conjunction) and OR (disjunction) and leaves are associated with literals), via a process called knowledge compilation [9]. This representation allows fast inference.

Credal probabilistic facts and credal annotated disjunctions (ADs) are two possible constructs to model imprecise probabilities [1]. Their syntaxes are, respectively,

$$[\alpha, \beta] :: a$$

with $0 \leq \alpha \leq \beta \leq 1$ and

$$[\alpha_1, \beta_1] :: h1; \ldots ; [\alpha_m, \beta_m] :: h_m :- b_1, \ldots , b_n$$

with $0 \leq \alpha_i \leq \beta_i \leq 1$, $\alpha_i + \sum_{j \neq i} \beta_j \geq 1$, $\beta_i + \sum_{j \neq i} \alpha_j \leq 1$, $\forall i \in \{1, \ldots , m\}$. We will refer to PASP extended with either one of the two constructs as PASP with imprecise probabilities.

3. Inference via Optimization

We proposed to perform inference in PASP with uncertain probabilities via optimization. Briefly, we extract a symbolic formula from the NNF representation of the program where the probabilities associated with credal facts and credal AD are kept symbolic and then call an optimization solver to minimize/maximize such formulas subject to constraints deriving from the structure of credal probabilistic facts and credal ADs. Namely, if $f_{lp}(X)$ and $f_{up}(X)$ are the formulas for the lower and upper probability for a query q and $X = \{\pi_1, \ldots \pi_k\}$ denotes the set of probability parameters associated with credal probabilistic facts, the lower and upper probability of a query can be computed, respectively, as

$$\begin{aligned} minimize \quad & f_{lp}(X) \\ s.t. \quad & \pi_i \in [l_i, u_i], \forall i \in \{1, \ldots , k\} \end{aligned}$$

and

$$\begin{aligned} maximize \quad & f_{up}(X) \\ s.t. \quad & \pi_i \in [l_i, u_i], \forall i \in \{1, \ldots , k\}. \end{aligned}$$

Let us clarify this with an example.

Example 1. *Consider the following PASP with two credal facts.*

```
[0.3,0.4]::a.
[0.4,0.9]::b.
q:- a.
q ; r :- b.
```

First, we convert it into

```
pa::a.
pb::b.
q:- a.
q ; r :- b.
```

where pa and pb are parameters. By traversing the NNF for the query q we obtain two equations: $f_{lp}(pa) = pa$ for the lower probability, and $f_{up}(pa, pb) = pa - pb \cdot (pa - 1)$ for the upper probability. $\underline{\mathrm{P}}(q)$ is computed by minimizing $f_{lp}(pa)$ with $pa \in [0.3, 0.4]$, obtaining 0.3. For $\overline{\mathrm{P}}(q)$, we need to maximize $f_{up}(pa, pb)$ with $pa \in [0.3, 0.4]$ and $pb \in [0.4, 0.9]$. In this case, we obtain 0.94. So, $[\underline{\mathrm{P}}(q), \overline{\mathrm{P}}(q)] = [0.3, 0.94]$.

If credal ADs are present, the optimization process becomes more involved and first it requires converting each credal AD into a combination of probabilistic facts and normal rules [10]. For example,

```
[0.1,0.3]::red;[0.2,0.4]::green;[0.4,0.6]::blue.
```

is expanded into

```
p1::f1.
p2::f2.
red :- f1.
green :- not f1, f2.
blue :- not f1, not f2.
```

Then, the optimization process also needs to consider constraints on the probabilities on the credal facts.

With n_{ad} credal ADs, the optimization problem for the lower probability is

$$
\begin{aligned}
minimize \quad & f(X) \\
s.t. \quad & \pi_i^l \cdot \prod_{j<i}(1 - \pi_j^l) - \alpha_i^l \geq 0, \\
& \beta_i^l - \pi_i^l \cdot \prod_{j<i}(1 - \pi_j^l) \geq 0, \\
& \forall l \in \{1, \ldots, n_{ad}\},\ \forall i \in \{1, \ldots, m_l\}
\end{aligned}
$$

assuming $\pi_i^{m_l} = 1$, where π_i^k is the probability associated with the i-th probabilistic fact related to the i-th head of the k-th AD and m_l is the number of disjuncts in the l-th AD. This requires imposing $2 \cdot m$ constraints for each credal AD with m heads. The problem to solve for the computation of the upper probability is analogous. More concretely, with the credal AD shown above, we have the following set of constraints: $c_1 - 0.1 \geq 0$, $0.3 - \pi_1 \geq 0$, $(1 - \pi_1) \cdot \pi_2 - 0.2 \geq 0$, $0.4 - (1 - \pi_1) \cdot \pi_2 \geq 0$, $(1 - \pi_1) \cdot (1 - \pi_2) - 0.4 \geq 0$, and $0.6 - (1 - \pi_1) \cdot (1 - \pi_2) \geq 0$.

The overall algorithm is as follows: first, credal facts and credal ADs are converted to remove ranges. Then, the program is converted into a NNF, and two equations are extracted, which are simplified to reduce the number of operations involved. Lastly, these are sent to a non-linear optimization solver that can manage non-linear constraints.

The algorithm was implemented in Python and the experimental evaluation was conducted by considering SymPy [11] to simplify equations and SciPy [12] as optimization solver with the COBYLA [13] and SLSQP [14] algorithms. Empirical results showed that: i) solving the optimization problem by considering a simplified version of the equations is much faster than considering the equations directly extracted from the NNF, even if the simplification process may be slow (for larger datasets, this takes more than 50% of the total execution time); ii) this approach is much faster than already existing solver based on enumeration [15]; iii) COBYLA is often more efficient and effective than SLSQP.

4. Conclusions

In this paper, we summarized [2] where inference in probabilistic answer set programs under the credal semantics with imprecise probabilities has been cast as a constrained non-linear optimization problem. Empirical results against an already existing solver based on enumeration shows the effectiveness of this approach.

References

[1] D. D. Mauá, F. G. Cozman, Specifying credal sets with probabilistic answer set programming, in: International Symposium on Imprecise Probabilities and Their Applications, 2023.

[2] D. Azzolini, F. Riguzzi, Inference in probabilistic answer set programs with imprecise probabilities via optimization, in: N. Kiyavash, J. M. Mooij (Eds.), Proceedings of the Fortieth Conference on Uncertainty in Artificial Intelligence, volume 244 of *Proceedings of Machine Learning Research*, PMLR, 2024, pp. 225–234.

[3] L. De Raedt, A. Kimmig, H. Toivonen, ProbLog: A probabilistic Prolog and its application in link discovery, in: M. M. Veloso (Ed.), 20th International Joint Conference on Artificial Intelligence (IJCAI 2007), volume 7, AAAI Press, 2007, pp. 2462–2467.

[4] T. Sato, A statistical learning method for logic programs with distribution semantics, in: L. Sterling (Ed.), Logic Programming, Proceedings of the Twelfth International Conference on Logic Programming, Tokyo, Japan, June 13-16, 1995, MIT Press, 1995, pp. 715–729. doi:10.7551/mitpress/4298.003.0069.

[5] F. G. Cozman, D. D. Mauá, The joy of probabilistic answer set programming: Semantics, complexity, expressivity, inference, International Journal of Approximate Reasoning 125 (2020) 218–239. doi:10.1016/j.ijar.2020.07.004.

[6] G. Brewka, T. Eiter, M. Truszczyński, Answer set programming at a glance, Communications of the ACM 54 (2011) 92–103. doi:10.1145/2043174.2043195.

[7] D. Azzolini, F. Riguzzi, Inference in probabilistic answer set programming under the credal semantics, in: R. Basili, D. Lembo, C. Limongelli, A. Orlandini (Eds.), AIxIA 2023 - Advances in Artificial Intelligence, volume 14318 of *Lecture Notes in Artificial Intelligence*, Springer, Heidelberg, Germany, 2023, pp. 367–380. doi:10.1007/978-3-031-47546-7_25.

[8] T. Eiter, M. Hecher, R. Kiesel, aspmc: New frontiers of algebraic answer set counting, Artificial Intelligence 330 (2024) 104109. doi:10.1016/j.artint.2024.104109.

[9] A. Darwiche, P. Marquis, A knowledge compilation map, Journal of Artificial Intelligence Research 17 (2002) 229–264. doi:10.1613/jair.989.

[10] L. De Raedt, B. Demoen, D. Fierens, B. Gutmann, G. Janssens, A. Kimmig, N. Landwehr, T. Mantadelis, W. Meert, R. Rocha, V. Costa, I. Thon, J. Vennekens, Towards digesting the alphabet-soup of statistical relational learning, in: NIPS 2008 Workshop on Probabilistic Programming, 2008.

[11] A. Meurer, C. P. Smith, M. Paprocki, O. Čertík, S. B. Kirpichev, M. Rocklin, A. Kumar, S. Ivanov, J. K. Moore, S. Singh, T. Rathnayake, S. Vig, B. E. Granger, R. P. Muller, F. Bonazzi, H. Gupta, S. Vats, F. Johansson, F. Pedregosa, M. J. Curry, A. R. Terrel, v. Roučka, A. Saboo, I. Fernando, S. Kulal, R. Cimrman, A. Scopatz, SymPy: symbolic computing in python, PeerJ Computer Science 3 (2017) e103. doi:10.7717/peerj-cs.103.

[12] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, SciPy 1.0 Contributors, SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python, Nature Methods 17 (2020) 261–272. doi:10.1038/s41592-019-0686-2.

[13] M. J. D. Powell, A direct search optimization method that models the objective and constraint functions by linear interpolation, in: S. Gomez, J.-P. Hennart (Eds.), Advances in Optimization and Numerical Analysis, Springer Netherlands, Dordrecht, 1994, pp. 51–67. doi:10.1007/978-94-015-8330-5_4.

[14] D. Kraft, Algorithm 733: TOMP-fortran modules for optimal control calculations, ACM Transactions on Mathematical Software 20 (1994) 262–281. doi:10.1145/192115.192124.

[15] R. L. Geh, J. Goncalves, I. C. Silveira, D. D. Maua, F. G. Cozman, dPASP: A comprehensive differentiable probabilistic answer set programming environment for neurosymbolic learning and reasoning, arXiv (2023).