The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19)

Probabilistic Logic Programming with Beta-Distributed Random Variables*

Federico Cerutti
Cardiff University
Cardiff
UK

Lance Kaplan
Army Research Laboratory
Adelphi, MD
USA

Angelika Kimmig
Cardiff University
Cardiff
UK

Murat Şensoy
Özyeğin University
Istanbul
Turkey

Abstract

We enable aProbLog—a probabilistic logical programming approach—to reason in presence of uncertain probabilities represented as Beta-distributed random variables. We achieve the same performance of state-of-the-art algorithms for highly specified and engineered domains, while simultaneously we maintain the flexibility offered by aProbLog in handling complex relational domains. Our motivation is that faithfully capturing the distribution of probabilities is necessary to compute an expected utility for effective decision making under uncertainty: unfortunately, these probability distributions can be highly uncertain due to sparse data. To understand and accurately manipulate such probability distributions we need a well-defined theoretical framework that is provided by the Beta distribution, which specifies a distribution of probabilities representing all the possible values of a probability when the exact value is unknown.

1 Introduction

In the last years, several probabilistic variants of Prolog have been developed, such as ICL (Poole 2000), Dyna (Eisner, Goldlust, and Smith 2005), PRISM (Sato and Kameya 2001) and ProbLog (De Raedt, Kimmig, and Toivonen 2007), with its aProbLog extension (Kimmig, Van den Broeck, and De Raedt 2011) to handle arbitrary labels from a semiring (Section 2.1). They all are based on definite clause logic (pure Prolog) extended with facts labelled with probability values. Their meaning is typically derived from Sato's distribution semantics (Sato 1995), which assigns a probability to every literal. The probability of a Herbrand interpretation, or possible world, is the product of the probabilities of the literals occurring in this world. The success probability is the probability that a query succeeds in a randomly selected world.

Faithfully capturing the distribution of the probabilities of such queries is necessary for effective decision making under uncertainty to compute an expected utility (Von Neumann and Morgenstern 2007). Often such distributions are learned from prior experiences that can be provided either by subject matter experts or by objective recordings.

Unfortunately, these probability distributions can be highly uncertain and this significantly affects decision making (Anderson, Hare, and Maskell 2016; Antonucci, Karlsson, and Sundgren 2014). In fact, not all scenarios are blessed with a substantial amount of data enabling reasonable characterisation of probability distributions. For instance, when dealing with adversarial behaviours such as policing operations, training data is sparse or subject matter experts have limited experience to elicit the probabilities.

To understand and accurately manipulate such probability distributions, we need a well-defined theoretical framework that is provided by the Beta distribution, which specifies a distribution of probabilities representing all the possible values of a probability when the exact value is unknown. This has been recently investigated in the context of singly-connected Bayesian Network, in an approach named Subjective Bayesian Network (SBN) (Ivanovska et al. 2015; Kaplan and Ivanovska 2016; 2018), that shows higher performance against other traditional approaches dealing with uncertain probabilities, such as Dempster-Shafer Theory of Evidence (Dempster 1968; Smets 1993), and replacing single probability values with closed intervals representing the possible range of probability values (Zaffalon and Fagiuoli 1998). SBN is based on Subjective Logic (Jøsang 2016) (Section 2.2) that provides an alternative, more intuitive, representation of Beta distributions as well as a calculus for manipulating them. Subjective logic has been successfully applied in a variety of domains, from trust and reputation (Jøsang, Hayward, and Pope 2006), to urban water management (Moglia, Sharma, and Maheepala 2012), to assessing the confidence of neural networks for image classification (Sensoy, Kaplan, and Kandemir 2018).

In this paper, we enable aProbLog (Kimmig, Van den Broeck, and De Raedt 2011) to reason in presence of uncertain probabilities represented as Beta distribution. Among other features, aProbLog is freely available[1] and it directly

*This research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence under Agreement Number W911NF-16-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

[1]https://dtai.cs.kuleuven.be/problog/

handles Bayesian networks,[2] which simplifies our experimental setting when comparing against SBN and other approaches on Bayesian Networks with uncertain probabilities. We determine a parametrisation for aProbLog (Section 3) deriving operators for addition, multiplication, and division operating on Beta-distributed random variables matching the results to a new Beta-distributed random variable using the moment matching method (Minka 2001; Kleiter 1996; Allen et al. 2008; Kaplan and Ivanovska 2018).

We achieve the same results of highly engineered approaches for inferencing in single-connected Bayesian networks—in particular in presence of high uncertainty in the distribution of probabilities which is our main research focus—and simultaneously we maintain the flexibility offered by aProbLog in handling complex relational domains. Results of our experimental analysis (Section 4) indeed indicate that the proposed approach (1) handles inferences in general aProbLog programs better than using standard subjective logic operators (Jøsang 2016) (Appendix A), and (2) it performs equivalently to state-of-the-art approaches of reasoning with uncertain probabilities (Kaplan and Ivanovska 2018; Zaffalon and Fagiuoli 1998; Smets 1993), despite the fact that they have been highly engineered for the specific case of single connected Bayesian Networks while we can handle general aProbLog programs.

2 Background

2.1 aProbLog

For a set J of ground facts, we define the set of literals $\mathrm{L}(J)$ and the set of interpretations $\mathcal{I}(J)$ as follows:

$$\mathrm{L}(J) = J \cup \{\neg f \mid f \in J\} \tag{1}$$

$$\mathcal{I}(J) = \{S \mid S \subseteq \mathrm{L}(J) \wedge \forall l \in J : \ l \in S \leftrightarrow \neg l \notin S\} \tag{2}$$

An algebraic Prolog (aProbLog) program (Kimmig, Van den Broeck, and De Raedt 2011) consists of:

- a *commutative semiring* $\langle \mathcal{A}, \oplus, \otimes, e^{\oplus}, e^{\otimes} \rangle$[3]
- a finite set of ground *algebraic facts* $\mathrm{F} = \{f_1, \ldots, f_n\}$
- a finite set BK of *background knowledge clauses*
- a *labeling function* $\delta : \mathrm{L}(\mathrm{F}) \to \mathcal{A}$

Background knowledge clauses are definite clauses, but their bodies may contain negative literals for algebraic facts. Their heads may not unify with any algebraic fact.

For instance, in the following aProbLog program

```
alarm :- burglary.
0.05 :: burglary.
```

`burglary` is an algebraic fact with label `0.05`, and `alarm :- burglary` represents a background knowledge clause, whose intuitive meaning is: in case of burglary, the alarm should go off.

The idea of splitting a logic program in a set of facts and a set of clauses goes back to Sato's distribution semantics (Sato 1995), where it is used to define a probability distribution over interpretations of the entire program in terms of a distribution over the facts. This is possible because a truth value assignment to the facts in F uniquely determines the truth values of all other atoms defined in the background knowledge. In the simplest case, as realised in ProbLog (De Raedt, Kimmig, and Toivonen 2007; Fierens et al. 2015), this basic distribution considers facts to be independent random variables and thus multiplies their individual probabilities. aProbLog uses the same basic idea, but generalises from the semiring of probabilities to general commutative semirings. While the distribution semantics is defined for countably infinite sets of facts, the set of ground algebraic facts in aProbLog must be finite.

In aProbLog, the label of a complete interpretation $I \in \mathcal{I}(\mathrm{F})$ is defined as the product of the labels of its literals

$$\mathbf{A}(I) = \bigotimes_{l \in I} \delta(l) \tag{3}$$

and the label of a set of interpretations $S \subseteq \mathcal{I}(\mathrm{F})$ as the sum of the interpretation labels

$$\mathbf{A}(S) = \bigoplus_{I \in S} \bigotimes_{l \in I} \delta(l) \tag{4}$$

A *query* q is a finite set of algebraic literals and atoms from the Herbrand base,[4] $q \subseteq \mathrm{L}(\mathrm{F}) \cup HB(\mathrm{F} \cup \mathrm{BK})$. We denote the set of interpretations where the query is true by $\mathcal{I}(q)$,

$$\mathcal{I}(q) = \{I \mid I \in \mathcal{I}(\mathrm{F}) \wedge I \cup \mathrm{BK} \models q\} \tag{5}$$

The label of query q is defined as the label of $\mathcal{I}(q)$,

$$\mathbf{A}(q) = \mathbf{A}(\mathcal{I}(q)) = \bigoplus_{I \in \mathcal{I}(q)} \bigotimes_{l \in I} \delta(l). \tag{6}$$

As both operators are commutative and associative, the label is independent of the order of both literals and interpretations.

In the context of this paper, we extend aProbLog to queries with evidence by introducing an additional division operator $\oslash$ that defines the conditional label of a query as follows:

$$\mathbf{A}(q|\boldsymbol{E} = \boldsymbol{e}) = \mathbf{A}(\mathcal{I}(q \ \wedge \ \boldsymbol{E} = \boldsymbol{e})) \oslash \mathbf{A}(\mathcal{I}(\boldsymbol{E} = \boldsymbol{e})) \tag{7}$$

where $\mathbf{A}(\mathcal{I}(q \ \wedge \ \boldsymbol{E} = \boldsymbol{e})) \oslash \mathbf{A}(\mathcal{I}(\boldsymbol{E} = \boldsymbol{e}))$ returns the label of $q \ \wedge \ \boldsymbol{E} = \boldsymbol{e}$ given the label of $\boldsymbol{E} = \boldsymbol{e}$. We refer to a specific choice of semiring, labeling function and division operator as an *aProbLog parametrisation*.

ProbLog is an instance of aProbLog with the following parameterisation, which we denote $\mathfrak{S}_p$:

$$\begin{array}{l} \mathcal{A} = \mathbb{R}_{\geqslant 0}; \\ a \ \oplus \ b = a + b; \\ a \ \otimes \ b = a \cdot b; \\ e^{\oplus} = 0; \\ e^{\otimes} = 1; \\ \delta(f) \in [0, 1]; \\ \delta(\neg f) = 1 - \delta(f); \\ a \ \oslash \ b = \frac{a}{b} \end{array} \tag{8}$$

[2] As pointed out by (Fierens et al. 2015), for such Bayesian network models, ProbLog inference is tightly linked to the inference approach of (Sang, Bearne, and Kautz 2005).

[3] That is, *addition* $\oplus$ and *multiplication* $\otimes$ are associative and commutative binary operations over the set $\mathcal{A}$, $\otimes$ distributes over $\oplus$, $e^{\oplus} \in \mathcal{A}$ is the neutral element with respect to $\oplus$, $e^{\otimes} \in \mathcal{A}$ that of $\otimes$, and for all $a \in \mathcal{A}$, $e^{\oplus} \otimes a = a \otimes e^{\oplus} = e^{\oplus}$.

[4] I.e., the set of ground atoms that can be constructed from the predicate, functor and constant symbols of the program.

2.2 Beta Distribution and Subjective Logic Opinions

When probabilities are uncertain—for instance because of limited observations—such an uncertainty can be captured by a Beta distribution, namely a distribution of possible probabilities. Let us consider only binary variables such as X that can take on the value of true or false, i.e., $X = x$ or $X = \bar{x}$. The value of X does change over different instantiations, and there is an underlying ground truth value for the probability p_x that X is true ($p_{\bar{x}} = 1 - p_x$ that X is false). If p_x is drawn from a Beta distribution, it has the following probability density function:

$$f_{\boldsymbol{\beta}}(p_x; \boldsymbol{\alpha}) = \frac{1}{\beta(\alpha_x, \alpha_{\bar{x}})} p_x^{\alpha_x - 1} (1 - p_x)^{\alpha_{\bar{x}} - 1} \tag{9}$$

for $0 \leqslant p_x \leqslant 1$, where $\beta(\cdot)$ is the beta function and the beta parameters are $\boldsymbol{\alpha}_X = \langle \alpha_x, \alpha_{\bar{x}} \rangle$, such that $\alpha_x > 1, \alpha_{\bar{x}} > 1$. Given a Beta-distributed random variable X,

$$s_X = \alpha_x + \alpha_{\bar{x}} \tag{10}$$

is its *Dirichlet strength* and

$$\mu_X = \frac{\alpha_x}{s_X} \tag{11}$$

is its mean. From (10) and (11) the beta parameters can equivalently be written as:

$$\boldsymbol{\alpha_X} = \langle \mu_X s_X,\ (1 - \mu_X) s_X \rangle. \tag{12}$$

The variance of a Beta-distributed random variable X is

$$\sigma_X^2 = \frac{\mu_X(1 - \mu_X)}{s_X + 1} \tag{13}$$

and from (13) we can rewrite s_X (10) as

$$s_X = \frac{\mu_X(1 - \mu_X)}{\sigma_X^2} - 1. \tag{14}$$

Parameter Estimation Given a random variable Z with known mean μ_Z and variance σ_Z^2, we can use the method of moments and (14) to estimate the $\boldsymbol{\alpha}$ parameters of a Beta-distributed variable Z' of mean $\mu_{Z'} = \mu_Z$ and

$$s_{Z'} = \max\left\{ \frac{\mu_Z(1 - \mu_Z)}{\sigma_Z^2} - 1, \frac{W a_Z}{\mu_Z}, \frac{W(1 - a_Z)}{(1 - \mu_Z)} \right\}. \tag{15}$$

(15) is needed to ensure that the resulting Beta-distributed random variable Z' does not lead to a $\boldsymbol{\alpha}_{Z'} \leqslant \langle 1, 1 \rangle$.

Beta-Distributed Random Variables from Observations The value of X can be observed from N_{ins} independent observations of X. If over these observations, n_x times $X = x$, $n_{\bar{x}} = N_{ins} - n_x$ times $X = \bar{x}$, then $\boldsymbol{\alpha}_X = \langle n_x + W a_X, n_{\bar{x}} + W(1 - a_X) \rangle$: a_X is the prior assumption, i.e. the probability that X is true in the absence of observations; and $W > 0$ is a prior weight indicating the strength of the prior assumption. Unless specified otherwise, in the following we will assume $\forall X, a_X = 0.5$ and $W = 2$, so to have an uninformative, uniformly distributed, prior.

Subjective Logic Subjective logic (Jøsang 2016) provides (1) an alternative, more intuitive, way of representing the parameters of a Beta-distributed random variables, and (2) a set of operators for manipulating them. A subjective opinion about a proposition X is a tuple $\omega_X = \langle b_X, d_X, u_X, a_X \rangle$, representing the belief, disbelief and uncertainty that X is true at a given instance, and, as above, a_X is the prior probability that X is true in the absence of observations. These values are non-negative and $b_X + d_X + u_X = 1$. The projected probability $P(x) = b_X + u_X \cdot a_X$, provides an estimate of the ground truth probability p_x.

The mapping from a Beta-distributed random variable X with parameters $\boldsymbol{\alpha}_X = \langle \alpha_x, \alpha_{\bar{x}} \rangle$ to a subjective opinion is:

$$\omega_X = \left\langle \frac{\alpha_x - W a_X}{s_X}, \frac{\alpha_{\bar{x}} - W(1 - a_X)}{s_X}, \frac{W}{s_X}, a_X \right\rangle \tag{16}$$

With this transformation, the mean of X is equivalent to the projected probability $P(x)$, and the Dirichlet strength is inversely proportional to the uncertainty of the opinion:

$$\mu_X = P(x) = b_X + u_X a_X, \quad s_X = \frac{W}{u_X} \tag{17}$$

Conversely, a subjective opinion ω_X translates directly into a Beta-distributed random variable with:

$$\boldsymbol{\alpha}_X = \left\langle \frac{W}{u_X} b_X + W a_X, \frac{W}{u_X} d_X + W(1 - a_X) \right\rangle \tag{18}$$

Subjective logic is a framework that includes various operators to indirectly determine opinions from various logical operations. In particular, we will make use of $\boxplus_{SL}$, $\boxtimes_{SL}$, and $\boxslash_{SL}$, resp. summing, multiplying, and dividing two subjective opinions as they are defined in (Jøsang 2016) (Appendix A). Those operators aim at faithfully matching the projected probabilities: for instance the multiplication of two subjective opinions $\omega_X \boxtimes_{SL} \omega_Y$ results in an opinion ω_Z such that $P(z) = P(x) \cdot P(z)$.

The straightforward approach to derive a aProbLog parametrisation for operations in subjective logic is to use the operators $\boxplus$, $\boxtimes$, and $\boxslash$.

Definition 1. The aProbLog parametrisation $\mathfrak{S}_{\text{SL}}$ is defined as follows:

$$\begin{array}{l}
\mathcal{A}_{\text{SL}} = \mathbb{R}^4_{\geqslant 0};\\
a \oplus_{\text{SL}} b = a \boxplus_{\text{SL}} b;\\
a \otimes_{\text{SL}} b = a \boxtimes_{\text{SL}} b;\\
e^{\oplus}\text{SL} = \langle 0, 1, 0, 0 \rangle;\\
e^{\otimes}\text{SL} = \langle 1, 0, 0, 1 \rangle;\\
\delta_{\text{SL}}(f_i) = \langle b_{f_i}, d_{f_i}, u_{f_i}, a_{f_i} \rangle \in [0, 1]^4;\\
\delta_{\text{SL}}(\neg f_i) = \langle d_{f_i}, b_{f_i}, u_{f_i}, 1 - a_{f_i} \rangle;\\
a \oslash_{\text{SL}} b = \begin{cases} a \boxslash_{\text{SL}} b & \text{if defined} \\ \langle 0, 0, 1, 0.5 \rangle & \text{otherwise} \end{cases}
\end{array} \tag{19}$$

Note that $\langle \mathcal{A}_{\text{SL}}, \oplus_{\text{SL}}, \otimes_{\text{SL}}, e^{\oplus}\text{SL}, e^{\otimes}\text{SL} \rangle$ does not form a commutative semiring in general. If we consider only the projected probabilities—i.e. the means of the associated Beta distributions—then $\boxplus$ and $\boxtimes$ are indeed commutative, associative, and $\boxtimes$ distributes over $\boxplus$. However, the uncertainty of the resulting opinion depends on the order of operands.

3 Operators for Beta-Distributed Random Variables

While SL operators try to faithfully characterise the projected probabilities, they employ an uncertainty maximisation principle to limit the belief commitments, hence they have a looser connection to the Beta distribution. The operators we derive in this section aim at maintaining such a connection.

Let us first define a sum operator between two independent Beta-distributed random variables X and Y as the Beta-distributed random variable Z such that $\mu_Z = \mu_{X+Y}$ and $\sigma^2_Z = \sigma^2_{X+Y}$. The sum (and in the following the product as well) of two Beta random variables is not necessarily a Beta random variable. Our approach, consistent with (Kaplan and Ivanovska 2018), approximates the resulting distribution as a Beta distribution via moment matching on mean and variance: this guarantees to approximate the result as a Beta distribution.

Definition 2 (Sum)**.** Given X and Y independent Beta-distributed random variables represented by the subjective opinion ω_X and ω_Y, the *sum* of X and Y ($\omega_X \boxplus^\beta \omega_Y$) is defined as the Beta-distributed random variable Z such that: $\mu_Z = \mu_{X+Y} = \mu_X + \mu_Y$ and $\sigma^2_Z = \sigma^2_{X+Y} = \sigma^2_X + \sigma^2_Y$.

$\omega_Z = \omega_X \boxplus^\beta \omega_Y$ can then be obtained as discussed in Section 2.2, taking (15) into consideration. The same applies for the following operators as well.

Let us now define the product operator between two independent Beta-distributed random variables X and Y as the Beta-distributed random variable Z such that $\mu_Z = \mu_{XY}$ and $\sigma^2_Z = \sigma^2_{XY}$.

Definition 3 (Product)**.** Given X and Y independent Beta-distributed random variables represented by the subjective opinion ω_X and ω_Y, the *product* of X and Y ($\omega_X \boxtimes^\beta \omega_Y$) is defined as the Beta-distributed random variable Z such that: $\mu_Z = \mu_{XY} = \mu_X\ \mu_Y$ and $\sigma^2_Z = \sigma^2_{XY} = \sigma^2_X(\mu_Y)^2 + \sigma^2_Y(\mu_X)^2 + \sigma^2_X\sigma^2_Y$.

Finally, let us define the conditioning-division operator between two independent Beta-distributed random variables X and Y, represented by subjective opinions ω_X and ω_Y, as the Beta-distributed random variable Z such that $\mu_Z = \mu_{\frac{X}{Y}}$ and $\sigma^2_Z = \sigma^2_{\frac{X}{Y}}$.

Definition 4 (Conditioning-Division)**.** Given $\omega_X = \langle b_X, d_X, u_X, a_X\rangle$ and $\omega_Y = \langle b_Y, d_Y, u_Y, a_Y\rangle$ subjective opinions such that X and Y are Beta-distributed random variables, $Y = \mathbf{A}(\mathcal{I}(\boldsymbol{E} = \boldsymbol{e})) = \mathbf{A}(\mathcal{I}(q \wedge \boldsymbol{E} = \boldsymbol{e})) \oplus \mathbf{A}(\mathcal{I}(\neg q \wedge \boldsymbol{E} = \boldsymbol{e}))$, with $\mathbf{A}(\mathcal{I}(q \wedge \boldsymbol{E} = \boldsymbol{e})) = X$. The *conditioning-division* of X by Y ($\omega_X \boxslash^\beta \omega_Y$) is defined as the Beta-distributed random variable Z such that:[5]

$$\mu_Z = \mu_{\frac{X}{Y}} = \mu_X \mu_{\frac{1}{Y}} \simeq \frac{\mu_X}{\mu_Y} \tag{20}$$

and

$$\sigma^2_Z \simeq (\mu_Z)^2(1-\mu_Z)^2 \cdot \cdot \left(\frac{\sigma^2_X}{(\mu_X)^2} + \frac{\sigma^2_Y - \sigma^2_X}{(\mu_Y - \mu_X)^2} + \frac{2\sigma^2_X}{\mu_X(\mu_Y - \mu_X)}\right) \tag{21}$$

We can now define a new aProbLog parametrisation similar to Definition 1 operating with our newly defined operators $\boxplus^\beta$, $\boxtimes^\beta$, and $\boxslash^\beta$.

Definition 5. The aProbLog parametrisation $\mathfrak{S}^\beta$ is defined as follows:

$$\begin{aligned}
&\mathcal{A}^\beta = \mathbb{R}^4_{\geqslant 0};\\
&a \oplus^\beta b = a \boxplus^\beta b;\\
&a \otimes^\beta b = a \boxtimes^\beta b;\\
&e^{\oplus^\beta} = \langle 1, 0, 0, 0.5\rangle;\\
&e^{\otimes^\beta} = \langle 0, 1, 0, 0.5\rangle;\\
&\delta^\beta(f_i) = \langle b_{f_i}, d_{f_i}, u_{f_i}, a_{f_i}\rangle \in [0,1]^4;\\
&\delta^\beta(\neg f_i) = \langle d_{f_i}, b_{f_i}, u_{f_i}, 1 - a_{f_i}\rangle;\\
&a \oslash^\beta b = a \boxslash^\beta b
\end{aligned} \tag{22}$$

As per Definition 1, also $\langle \mathcal{A}^\beta, \oplus^\beta, \otimes^\beta, e^{\oplus^\beta}, e^{\otimes^\beta}\rangle$ is not in general a commutative semiring. Means are correctly matched to projected probabilities, therefore for them $\mathfrak{S}^\beta$ actually operates as a semiring. However, for what concerns variance, the product is not distributive over addition: $\sigma^2_{X(Y+Z)} = \sigma^2_X(\mu_Y + \mu_Z)^2 + (\sigma^2_Y + \sigma^2_Z)\mu^2_X + \sigma^2_X(\sigma^2_Y + \sigma^2_Z) \neq \sigma^2_X(\mu^2_Y + \mu^2_Z) + (\sigma^2_Y + \sigma^2_Z)\mu^2_X + \sigma^2_X(\sigma^2_Y + \sigma^2_Z) = \sigma^2_{(XY)+(XZ)}$. The approximation error we introduce is therefore

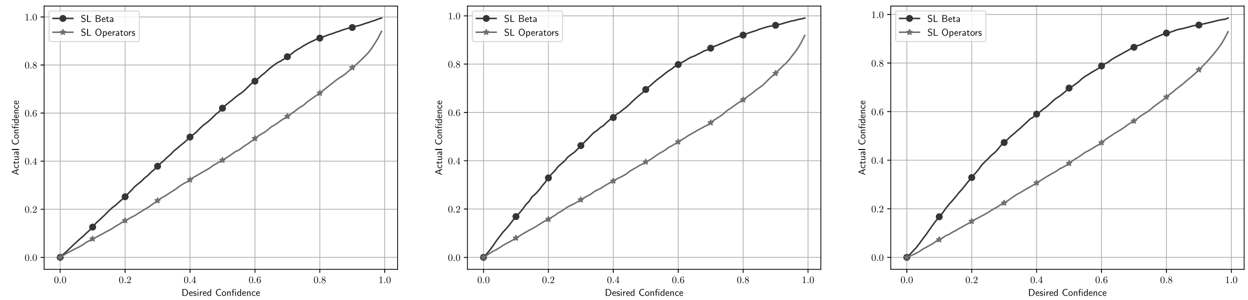
$$e(X, Y, Z) \leqslant \frac{2\mu_Y\mu_Z\sigma^2_X}{\sigma^2_X(\mu^2_Y + \mu^2_Z) + (\mu^2_X + \sigma^2_X)(\sigma^2_Y + \sigma^2_Z)} \tag{23}$$

and it minimally affects the results both in the case of low and in the case of high uncertainty in the random variables.

4 Experimental Analysis

To evaluate the suitability of using $\mathfrak{S}^\beta$ in aProbLog for uncertain probabilistic reasoning, we run an experimental analysis involving several aProbLog programs with unspecified labelling function. For each program, first labels are derived for $\mathfrak{S}_p$ by selecting the ground truth probabilities from a uniform random distribution. Then, for each label of the aProbLog program over $\mathfrak{S}_p$, we derive a subjective opinion by observing N_{ins} instantiations of the random variables comprising the aProbLog program over $\mathfrak{S}_p$ so to simulate data sparsity (Kaplan and Ivanovska 2018). We then proceed analysing the inference on specific query nodes $\boldsymbol{q}$ in the presence of a set of evidence $\boldsymbol{E} = \boldsymbol{e}$ using aProbLog with $\mathfrak{S}_{\text{SL}}$ and $\mathfrak{S}^\beta$ over the subjective opinion labels, and compare the RMSE to the actual ground truth of using aProbLog with $\mathfrak{S}_p$. This process of inference to determine the marginal Beta distributions is repeated 1000 times by considering 100 random choices for each label of the aProbLog with $\mathfrak{S}_p$, i.e. the ground truth, and for each ground truth 10 repetitions of sampling the interpretations used to derive the subjective

[5] In the following, $\simeq$ highlights the fact that the results are obtained using the the first order Taylor approximation.



(a) (b) (c)

Figure 1: Actual versus desired significance of bounds derived from the uncertainty for Smokers & Friends with: (a) $N_{ins} = 10$; (b $N_{ins} = 50$; and (c) $N_{ins} = 100$. Best closest to the diagonal. In the figure, *SL Beta* represents aProbLog with $\mathfrak{S}^{\beta}$, and *SL Operators* represents aProbLog with $\mathfrak{S}_{\text{SL}}$.

Program	N_{ins}		$\mathfrak{S}^{\beta}$	$\mathfrak{S}_{\text{SL}}$
Friends & Smokers	10	Actual	**0.1014**	0.1514
		Predicted	0.1727	0.1178
	50	Actual	**0.0620**	0.1123
		Predicted	0.0926	0.0815
	100	Actual	**0.0641**	0.1253
		Predicted	0.1150	0.0893

Table 1: RMSE for the queried variables in the Friends & Smokers program: best results for the actual RMSE in bold.

opinion labels used in $\mathfrak{S}_{\text{SL}}$ and $\mathfrak{S}^{\beta}$ observing N_{ins} instantiations of all the variables.

Following (Kaplan and Ivanovska 2018), we judge the quality of the Beta distributions of the queries on how well its expression of uncertainty captures the spread between its projected probability and the actual ground truth probability. In simulations where the ground truths are known, such as ours, confidence bounds can be formed around the projected probabilities at a significance level of γ and determine the fraction of cases when the ground truth falls within the bounds. If the uncertainty is well determined by the Beta distributions, then this fraction should correspond to the strength γ of the confidence interval (Kaplan and Ivanovska 2018, Appendix C).

4.1 Inferences in Arbitrary aProbLog Programs

We first considered the famous Friends & Smokers problem[6] with fixed queries and set of evidence, to illustrate the behaviour between $\mathfrak{S}_{\text{SL}}$ and $\mathfrak{S}^{\beta}$. Table 1 provides the root mean square error (RMSE) between the projected probabilities and the ground truth probabilities for all the inferred query variables for N_{ins} = 10, 50, 100. The table also includes the predicted RMSE by taking the square root of the average—over the number of runs—variances from the inferred marginal Beta distributions, cf. Eq. (13). Figure 1 plots the desired and actual significance levels for the confidence intervals (best closest to the diagonal), i.e. the fractions of times the ground truth falls within confidence bounds set to capture x% of the data.

The aProbLog with $\mathfrak{S}^{\beta}$ exhibits the lowest RMSE, and is a little conservative in estimating its own RMSE, while aProbLog with $\mathfrak{S}_{\text{SL}}$ is overconfident. This reflects in Figure 1, with the results of aProbLog with $\mathfrak{S}^{\beta}$ being over the diagonal, and those of aProbLog with $\mathfrak{S}_{\text{SL}}$ being below it.

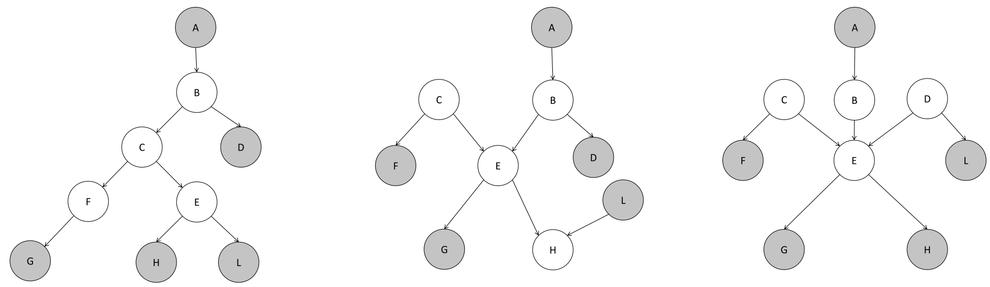
4.2 Inferences in aProbLog Programs Representing Single-Connected Bayesian Networks

We compared our approach against the state-of-the-art approaches for reasoning with uncertain probabilites—Subjective Bayesian Network (Ivanovska et al. 2015; Kaplan and Ivanovska 2016; 2018), Credal Network (Zaffalon and Fagiuoli 1998), and Belief Network (Smets 1993)—in the case that is handled by all of them, namely single connected Bayesian networks. We considered three networks proposed in (Kaplan and Ivanovska 2018) that are depicted in Figure 2: from each network, we straightforwardly derived a aProbLog program.

As before, Table 2 provides the root mean square error (RMSE) between the projected probabilities and the ground truth probabilities for all the inferred query variables for N_{ins} = 10, 50, 100, together with the RMSE predicted by taking the square root of the average variances from the inferred marginal Beta distributions. Figure 3 plots the desired and actual significance levels for the confidence intervals (best closest to the diagonal).

Table 2 shows that aProbLog with $\mathfrak{S}^{\beta}$ shares the best performance with the state-of-the-art Subjective Bayesian Networks—in terms of actual RMSE—for Net1, and in two out of three cases of Net2 (all of them from a practical standpoint). This is clearly a significant achievement considering that Subjective Bayesian network is the state-of-the-art approach when dealing only with single connected Bayesian Networks with uncertain probabilities, while aProbLog with

[6] https://dtai.cs.kuleuven.be/problog/tutorial/basic/05_smokers.html



(a) (b) (c)

Figure 2: Network structures tested where the exterior gray variables are directly observed and the remaining are queried: (a) Net1, a tree; (b) Net2, singly connected network with one node having two parents; (c) Net3, singly connected network with one node having three parents.

	N_{ins}		$\mathfrak{S}^{\beta}$	$\mathfrak{S}_{\text{SL}}$	SBN	GBT	Credal
Net1	10	A	**0.1505**	0.2078	**0.1505**	0.1530	0.1631
		P	0.1994	0.1562	0.1470	0.0868	0.2009
	50	A	0.0555	0.0895	0.0555	0.0619	**0.0553**
		P	0.0950	0.0579	0.0563	0.0261	0.0761
	100	A	**0.0766**	0.1182	**0.0766**	0.0795	0.0771
		P	0.1280	0.0772	0.0763	0.0373	0.1028
Net2	10	A	**0.1387**	0.2089	**0.1387**	0.1416	0.1459
		P	0.2031	0.1662	0.1391	0.1050	0.1849
	50	A	0.0537	0.0974	0.0537	0.0561	**0.0528**
		P	0.1002	0.0671	0.0520	0.0342	0.0683
	100	A	0.0730	0.1229	**0.0726**	0.0752	0.0728
		P	0.1380	0.0863	0.0725	0.0482	0.0949
Net3	10	A	0.1566	0.2111	**0.1534**	0.1554	0.1643
		P	0.1935	0.1517	0.1467	0.0832	0.1964
	50	A	0.0697	0.0947	**0.0548**	0.0584	**0.0548**
		P	0.0926	0.0602	0.0553	0.0242	0.0720
	100	A	0.0879	0.1242	0.0745	0.0776	**0.0743**
		P	0.1232	0.0798	0.0743	0.0347	0.0973

Table 2: RMSE for the queried variables in the various networks: A stands for Actual, P for Predicted. Best results for the Actual RMSE in bold.

$\mathfrak{S}^{\beta}$ can also handle much more complex problems. Net3 results are slightly worse due to approximations induced in the floating point operations used in the implementation: the more the connections of a node in the Bayesian network (e.g. node E in Figure 2c), the higher the number of operations involved in (7). A more accurate code engineering can address it. Consistently with Table 1, aProbLog with $\mathfrak{S}^{\beta}$ has lower RMSE than with $\mathfrak{S}_{\text{SL}}$ and it underestimates its predicted RMSE, while aProbLog with $\mathfrak{S}_{\text{SL}}$ overestimates it.

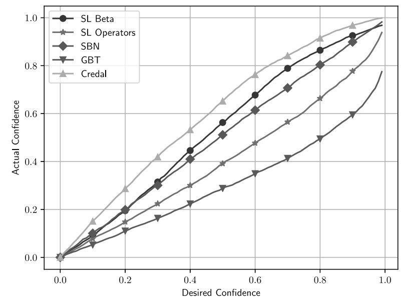
From visual inspection of Figure 3, it is evident that aProbLog with $\mathfrak{S}^{\beta}$ performs best in presence of high uncertainty ($N_{ins} = 10$). In presence of lower uncertainty, instead, it underestimates its own prediction up to a desired confidence between 0.6 and 0.8, and overestimate it after. This is due to the fact that aProbLog computes the conditional distributions at the very end of the process and $\mathfrak{S}^{\beta}$ relies, in (21), on the assumption that X and Y are uncorrelated. However, since the correlation between X and Y is inversely proportional to $\sqrt{\sigma_X^2 \sigma_Y^2}$, the lower the uncertainty, the less accurate our approximation.

5 Conclusion

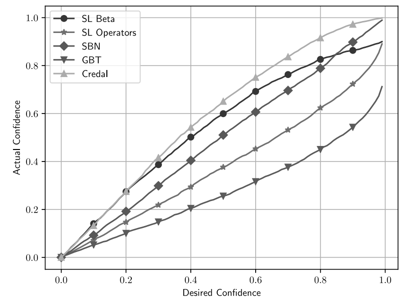
We enabled the aProbLog approach to probabilistic logic programming to reason in presence of uncertain probabilities represented as Beta-distributed random variables. Other extensions to logic programming can handle uncertain probabilities by considering intervals of possible probabilities (Ng and Subrahmanian 1992), similarly to the Credal network approach we compared against in Section 4; or by sampling random distributions, including ProbLog itself and cplint (Alberti et al. 2017) among others. Our approach does not require sampling or Monte Carlo computation, thus being significantly more efficient.

Our experimental section shows that the proposed operators outperform the standard subjective logic operators and they are as good as the state-of-the-art approaches for uncertain probabilities in Bayesian networks while being able to handle much more complex problems. Moreover, in presence of high uncertainty, which is our main research focus, the approximations we introduce in this paper are minimal, as Figures 3a, 3d, and 3g show, with the results of aProbLog with $\mathfrak{S}^{\beta}$ being very close to the diagonal.

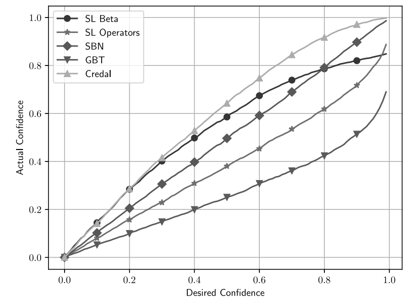
As part of future work we will (1) provide a different characterisation of the variance in (21) taking into consideration the correlation between X and Y; (2) test the boundaries of our approximations to provide practitioners with pragmatic assessments and assurances; and (3) introduce an expectation-maximisation (EM) algorithm for learning labels representing Beta-distributed random variables with partial interpretations and compare it against the LFI algorithm (Gutmann, Thon, and De Raedt 2011) for ProbLog.



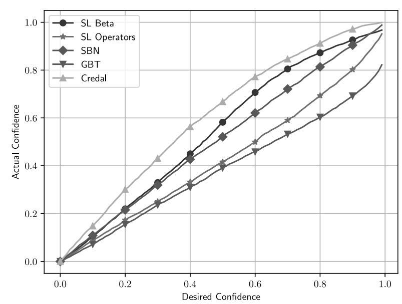
(a)



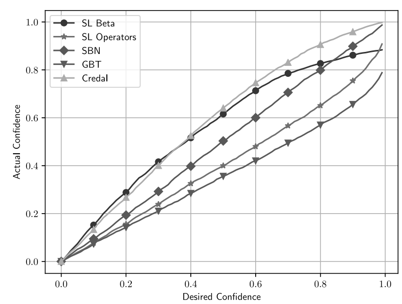
(b)



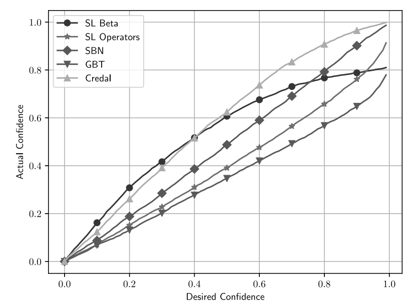
(c)



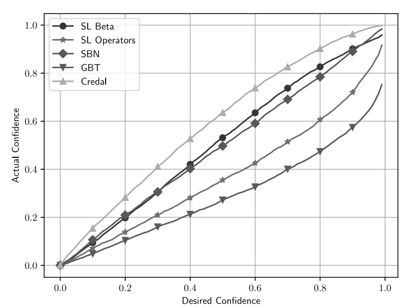
(d)



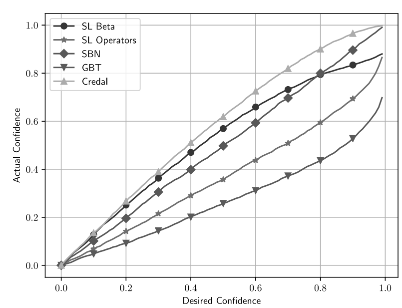
(e)



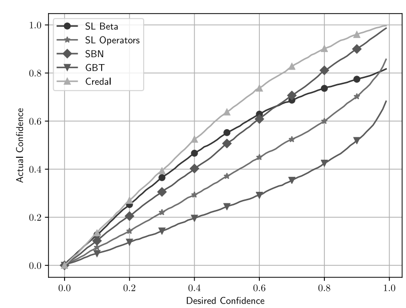
(f)



(g)



(h)



(i)

Figure 3: Actual versus desired significance of bounds derived from the uncertainty for: (a) Net1 with $N_{ins} = 10$; (b) Net1 with $N_{ins} = 50$; (c) Net1 with $N_{ins} = 100$; (d) Net2 with $N_{ins} = 10$; (e) Net2 with $N_{ins} = 50$; (f) Net2 with $N_{ins} = 100$; (g) Net3 with $N_{ins} = 10$; (h) Net3 with $N_{ins} = 50$; (i) Net3 with $N_{ins} = 100$. Best closest to the diagonal. In the figure, *SL Beta* represents aProbLog with $\mathfrak{S}^{\beta}$, and *SL Operators* represents aProbLog with $\mathfrak{S}_{\text{SL}}$.

A Subjective Logic Operators of Sum, Multiplication, and Division

Let us recall the following operators as defined in (Jøsang 2016). Let $\omega_X = \langle b_X, d_X, u_X, a_X \rangle$ and $\omega_Y = \langle b_Y, d_Y, u_Y, a_Y \rangle$ be two subjective logic opinions, then:

- the opinion about $X \cup Y$ (**sum**, $\omega_X \boxplus_{\text{SL}} \omega_Y$) is defined as $\omega_{X\cup Y} = \langle b_{X\cup Y}, d_{X\cup Y}, u_{X\cup Y}, a_{X\cup Y} \rangle$, where $b_{X\cup Y} = b_X + b_Y$, $d_{X\cup Y} = \frac{a_X(d_X - b_Y) + a_Y(d_Y - b_X)}{a_X + a_Y}$, $u_{X\cup Y} = \frac{a_X u_X + a_Y u_Y}{a_X + a_Y}$, and $a_{X\cup Y} = a_X + a_Y$;
- the opinion about $X \wedge Y$ (**product**, $\omega_X \boxtimes_{\text{SL}} \omega_Y$) is defined—under assumption of independence—as $\omega_{X\wedge Y} = \langle b_{X\wedge Y}, d_{X\wedge Y}, u_{X\wedge Y}, a_{X\wedge Y} \rangle$, where $b_{X\wedge Y} = b_X b_Y + \frac{(1-a_X)a_Y b_X u_Y + a_X(1-a_Y)u_X b_Y}{1-a_X a_Y}$, $d_{X\wedge Y} = d_X + d_Y - d_X d_Y$, $u_{X\wedge Y} = u_X u_Y + \frac{(1-a_Y)b_X u_Y + (1-a_X)u_X b_Y}{1-a_X a_Y}$, and $a_{X\wedge Y} = a_X a_Y$;
- the opinion about the division of X by Y, $X \tilde{\wedge} Y$ (**division**, $\omega_X \boxslash_{\text{SL}} \omega_Y$) is defined as $\omega_{X\tilde{\wedge}Y} = \langle b_{X\tilde{\wedge}Y}, d_{X\tilde{\wedge}Y}, u_{X\tilde{\wedge}Y}, a_{X\tilde{\wedge}Y} \rangle$ $b_{X\tilde{\wedge}Y} = \frac{a_Y(b_X + a_X u_X)}{(a_Y - a_X)(b_Y + a_Y u_Y)} - \frac{a_X(1-d_X)}{(a_Y - a_X)(1-d_Y)}$, $d_{X\tilde{\wedge}Y} = \frac{d_X - d_Y}{1-d_Y}$, $u_{X\tilde{\wedge}Y} = \frac{a_Y(1-d_X)}{(a_Y-a_X)(1-d_Y)} - \frac{a_Y(b_X + a_X u_X)}{(a_Y - a_X)(b_Y + a_Y u_Y)}$, and $a_{X\tilde{\wedge}Y} = \frac{a_X}{a_Y}$,
subject to: $a_X < a_Y$; $d_X \geqslant d_Y$; $b_X \geqslant \frac{a_X(1-a_Y)(1-d_X)b_Y}{(1-a_X)a_Y(1-d_Y)}$; $u_X \geqslant \frac{(1-a_Y)(1-d_X)u_Y}{(1-a_X)(1-d_Y)}$.

References

Alberti, M.; Bellodi, E.; Cota, G.; Riguzzi, F.; and Zese, R. 2017. cplint on SWISH: Probabilistic logical inference with a web browser. *Intelligenza Artificiale* 11(1):47–64.

Allen, T. V.; Singh, A.; Greiner, R.; and Hooper, P. 2008. Quantifying the uncertainty of a belief net response: Bayesian error-bars for belief net inference. *Artificial Intelligence* 172(4):483–513.

Anderson, R.; Hare, N.; and Maskell, S. 2016. Using a bayesian model for confidence to make decisions that consider epistemic regret. In *19th International Conference on Information Fusion*, 264–269.

Antonucci, A.; Karlsson, A.; and Sundgren, D. 2014. Decision making with hierarchical credal sets. In Laurent, A.; Strauss, O.; Bouchon-Meunier, B.; and Yager, R. R., eds., *Information Processing and Management of Uncertainty in Knowledge-Based Systems*, 456–465.

De Raedt, L.; Kimmig, A.; and Toivonen, H. 2007. ProbLog: A probabilistic Prolog and its application in link discovery. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, 2462–2467.

Dempster, A. P. 1968. A generalization of bayesian inference. *Journal of the Royal Statistical Society. Series B (Methodological)* 30(2):205–247.

Eisner, J.; Goldlust, E.; and Smith, N. A. 2005. Compiling comp ling: Practical weighted dynamic programming and the dyna language. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, 281–290.

Fierens, D.; Van den Broeck, G.; Renkens, J.; Shterionov, D.; Gutmann, B.; Thon, I.; Janssens, G.; and De Raedt, L. 2015. Inference and learning in probabilistic logic programs using weighted Boolean formulas. *Theory and Practice of Logic Programming* 15(03):358–401.

Gutmann, B.; Thon, I.; and De Raedt, L. 2011. Learning the parameters of probabilistic logic programs from interpretations. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 581–596.

Ivanovska, M.; Jøsang, A.; Kaplan, L.; and Sambo, F. 2015. Subjective networks: Perspectives and challenges. In *Proc. of the 4th International Workshop on Graph Structures for Knowledge Representation and Reasoning*, 107–124.

Jøsang, A.; Hayward, R.; and Pope, S. 2006. Trust network analysis with subjective logic. In *Proceedings of the 29th Australasian Computer Science Conference-Volume 48*, 85–94.

Jøsang, A. 2016. *Subjective Logic: A Formalism for Reasoning Under Uncertainty*. Springer.

Kaplan, L., and Ivanovska, M. 2016. Efficient subjective Bayesian network belief propagation for trees. In *19th International Conference on Information Fusion*, 1300–1307.

Kaplan, L., and Ivanovska, M. 2018. Efficient belief propagation in second-order bayesian networks for singly-connected graphs. *International Journal of Approximate Reasoning* 93:132–152.

Kimmig, A.; Van den Broeck, G.; and De Raedt, L. 2011. An algebraic prolog for reasoning about possible worlds. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*, 209–214.

Kleiter, G. D. 1996. Propagating imprecise probabilities in bayesian networks. *Artificial Intelligence* 88(1):143 – 161.

Minka, T. P. 2001. Expectation propagation for approximate bayesian inference. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*, 362–369.

Moglia, M.; Sharma, A. K.; and Maheepala, S. 2012. Multi-criteria decision assessments using subjective logic: Methodology and the case of urban water strategies. *Journal of Hydrology* 452-453:180–189.

Ng, R., and Subrahmanian, V. 1992. Probabilistic logic programming. *Information and Computation* 101(2):150–201.

Poole, D. 2000. Abducing through negation as failure: stable models within the independent choice logic. *The Journal of Logic Programming* 44(1):5–35.

Sang, T.; Bearne, P.; and Kautz, H. 2005. Performing bayesian inference by weighted model counting. In *Proceedings of the 20th National Conference on Artificial Intelligence - Volume 1*, 475–481.

Sato, T., and Kameya, Y. 2001. Parameter learning of logic programs for symbolic-statistical modeling. *Journal of Artificial Intelligence Research* 15(1):391–454.

Sato, T. 1995. A statistical learning method for logic programs with distribution semantics. In *Proceedings of the 12th International Conference on Logic Programming (ICLP-95)*.

Sensoy, M.; Kaplan, L.; and Kandemir, M. 2018. Evidential deep learning to quantify classification uncertainty. In *32nd Conference on Neural Information Processing Systems (NIPS 2018)*.

Smets, P. 1993. Belief functions: The disjunctive rule of combination and the generalized Bayesian theorem. *International Journal of Approximate Reasoning* 9:1– 35.

Von Neumann, J., and Morgenstern, O. 2007. *Theory of games and economic behavior (commemorative edition)*. Princeton university press.

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