TPLP: Page 1–18.

doi:10.1017/S1471068425100161

Integrating Belief Domains into Probabilistic Logic Programs

DAMIANO AZZOLINI
University of Ferrara
(*e-mail:* damiano.azzolini@unife.it)

FABRIZIO RIGUZZI
University of Ferrara
(*e-mail:* fabrizio.riguzzi@unife.it)

THERESA SWIFT
Johns Hopkins Applied Physics Lab
(*e-mail:* theresasturn@gmail.com)

submitted 24 July 2025; revised 24 July 2025; accepted 27 July 2025

Abstract

Probabilistic Logic Programming (PLP) under the distribution semantics is a leading approach to practical reasoning under uncertainty. An advantage of the distribution semantics is its suitability for implementation as a Prolog or Python library, available through two well-maintained implementations, namely ProbLog and cplint/PITA. However, current formulations of the distribution semantics use point-probabilities, making it difficult to express epistemic uncertainty, such as arises from, for example, hierarchical classifications from computer vision models. Belief functions generalize probability measures as non-additive capacities and address epistemic uncertainty via interval probabilities. This paper introduces interval-based *Capacity Logic Programs* based on an extension of the distribution semantics to include belief functions and describes properties of the new framework that make it amenable to practical applications.

Keywords: statistical relational AI, inference, tabling, imprecise probability

1 Introduction

Despite the importance of probabilistic reasoning in Symbolic AI, the use of point probabilities in reasoning can be problematic. Point probabilities may implicitly assume a degree of knowledge about a distribution that is not actually available, resulting in reasoning based on unknown bias and variance. Approaches to this lack of knowledge, or *epistemic uncertainty*, often use probabilistic intervals, such as Credal Networks (Cozman, 2000) or interval probabilities (Weichselberger, 2000). One appealing approach generalizes probability theory through *belief functions* or *Dempster Shafer theory* (Shafer, 1976).

Example 1.
(Drawing Balls from an Urn). *To illustrate how belief functions address epistemic uncertainty, suppose we want to determine the probability of picking a ball of a given color from an urn, given the knowledge:*

- *30% of the balls are red*
- *10% of the balls are blue*
- *60% of the balls are either blue or yellow*

It is clear how to assign probability to red balls, but what about blue or yellow balls? At least 10% and at most 70% of the balls are blue. Similarly, between 0% and 60% of the balls are yellow. As will be explained in Section 2.1, belief functions state that the belief that a ball must be blue is 10%, while the plausibility that a ball could be blue is 70%.

Example 1 describes a situation where *aleatory* uncertainty, that is the uncertainty inherent in pulling a ball from the urn, cannot be fully specified because of epistemic uncertainty, that is limited evidence about the distribution of blue and yellow balls. Belief functions separate aleatory from epistemic uncertainty by generalizing probability distributions to *mass functions* that, in discrete domains, may assign uncertainty mass to *any* set in a domain rather than just to singleton sets. For instance in Example 1, a mass function m_{urn} would assign 0.6 to the set {`blue,yellow`}, 0.1 to {`blue`}, and 0.3 to {`red`}. Belief functions thus can be seen both as a way of handling epistemic uncertainty or incomplete evidence (Shafer, 1976), and as a generalization of probability theory (Halpern and Fagin, 1992). Mathematically, because belief functions are non-additive, they are not measures like probabilities, but Choquet *capacities* (Choquet, 1954).

Let us now introduce a more involved example on how incorporation of belief can help to address an important issue for neuro-symbolic reasoning.

Example 2.
(Reasoning about Visual Classifications). *Consider the actions of a unmanned aerial vehicle (UAV) that flies over roadways to monitor safety issues such as broken-down vehicles on the side of a road, a task that includes visual classification. As a first step, an image is sent to a neural visual model V_{mod} that locates and classifies objects within bounding boxes. A reasoner then combines this visual information with traffic reports and background knowledge to determine the proper action to take.*

To understand the relevance of belief functions to this scenario, consider that visual models like V_{mod} associate a confidence score with each object classification. Typically such scores are normalized via a softmax transformation, but even if carefully calibrated (Guo et al. 2017), the scores may not have a frequentist or subjectivist probabilistic interpretation. One way to provide a frequentist interpretation is to construct a confusion matrix M_{Cls} from V_{mod}'s test set. Given the set Cls of object types in V_{mod}, M_{Cls} provides conditional probability estimates about true classifications based on detected classifications, and vice-versa.

A common situation when using a visual model is that some of its training labels may be mapped to leaf classes of an ontology, while others may be mapped to inner classes. This is an example of Hierarchical Multi-Label Classification (Giunchiglia

https://doi.org/10.1017/S1471068425100161 Published online by Cambridge University Press

Ontology	Probability Mass Assignment
Stationary Object	0.0332
Dumpster	0.264
Kiosk	0.011
Motorized Vehicle	0.02
Passenger Vehicle	0.228
Personal Passenger Vehicle	0.0489
Chevy	0.549
Fiat	0.102
Material Transport Vehicle	0.203
Freight Truck	0.0182
Cement Truck	0.0709
Cattle truck	0.0135
...	...

Fig 1. Example of hierarchy of objects.

and Lukasiewicz, 2020) where the classes are organized in a tree and the predictions must be coherent, that is an instance belonging to a class must belong to all of its ancestors in the hierarchy. Figure 1 shows a fragment of an ontology and a thresholded confusion vector for V_{mod} representing $P(trueClass \mid detectedClass)$ for an unknown object O_{unk}. Note that V_{mod} assigns probability mass both to general levels, such as Passenger Vehicle, *and to leaf levels such as* Chevy. *To reason about this situation, a flexible model, capable of handling both aleatory and epistemic uncertainty, is needed.*

This paper shows how the distribution semantics for Probabilistic Logic Programs (PLPs) (Poole, 1993; Sato, 1995) can be extended to incorporate *belief domains* as a foundation for *Capacity Logic Programs (CaLPs)*. Specifically, our contribution is to fully develop the distribution semantics with belief domains, and based on that development,

- We show that the set of all *belief worlds* for a CaLP form a *normalized capacity* – an analog of a probability measure that is used for belief functions.
- We demonstrate measure equivalence between (pairwise-incompatible) composite choices and belief worlds; and
- We show that a CaLP also forms the basis of a normalized capacity space.

The upshot of these correspondences of belief worlds and composite choices is that the implementation of CaLPs will be possible for systems based on the distribution semantics such as ProbLog and Cplint/PITA. The structure of the paper is as follows. Section 2 presents background on belief functions and the distribution semantics. Section 3 extends the distribution semantics to include belief functions, and characterizes its properties. Section 5 surveys related work and Section 6 concludes the paper.

2 Preliminaries

Throughout this paper, we restrict our attention to discrete probability measures and capacities, both defined over finite sets, and to programs without function symbols.

https://doi.org/10.1017/S1471068425100161 Published online by Cambridge University Press

2.1 Capacities and belief functions

We recall the definition of probability measures and spaces (Ash and Doléans-Dade, 2000).

Definition 1
(Probability Measure and Space). *Let $\mathcal{X}$ be a non-empty finite set called the sample space and $\mathbb{P}(\mathcal{X})$ the powerset of $\mathcal{X}$ called the event space. A probability measure is a function $P : \mathbb{P}(\mathcal{X}) \to \mathbb{R}$ such that i) $\forall x \in \mathbb{P}(\mathcal{X}), P(x) \geq 0$, ii) $P(\mathcal{X}) = 1$, and iii) for disjoint $S_i \in \mathbb{P}(\mathcal{X}) : P(\bigcup_i \mathcal{S}_i) = \sum_i P(\mathcal{S}_i)$. A probability space is a triple $(\mathcal{X}, \mathbb{P}(\mathcal{X}), P)$.*[1]

All measures, including probability measures, have the additivity property. That is, the measure assigned to the union of disjoint events is equal to the sum of the measures of the sets. Belief and plausibility functions (Shafer, 1976) are non-additive *capacities* rather than measures (Choquet, 1954; Grabish, 2016).

CaLPs in Section 3 use belief and plausibility functions based on multiple capacity spaces or *domains*. Accordingly, we use *domain identifiers* in the following definitions.

Definition 2
(Capacity Space). *Let $\mathcal{X}$ be a non-empty finite set called the frame of discernment. A capacity is a function $\xi : \mathbb{P}(\mathcal{X}) \to [0, 1]$ such that i) $\xi(\emptyset) = 0$ and ii) for $\mathcal{A}, \mathcal{B} \subseteq \mathcal{X}$, $\mathcal{A} \subseteq \mathcal{B} \Rightarrow \xi(\mathcal{A}) \leq \xi(\mathcal{B})$. If $\xi(\mathcal{X}) = 1$, ξ is normalized. A capacity space is a tuple $(D, \mathcal{X}, \mathbb{P}(\mathcal{X}), \xi)$ where D is a string called a domain identifier.*

Belief and plausibility functions are capacities based on *mass functions*.

Definition 3
(Mass Functions). *Let $\mathcal{X}$ be a non-empty finite set and D a domain identifier. A mass function $mass : D \times \mathbb{P}(\mathcal{X}) \to \mathbb{R}$ satisfies the following properties: i) $mass(D, \emptyset) = 0$, ii) $\forall x \in \mathbb{P}(\mathcal{X}), mass(D, x) \geq 0$, and iii) $\sum_{x \in \mathbb{P}(\mathcal{X})} mass(D, x) = 1$. A set $x \subseteq \mathbb{P}(\mathcal{X})$ where $mass(D, x) \neq 0$ is a focal set.*

Definition 4
(Belief and Plausibility Capacities). *Let $\mathcal{X}$ be a non-empty finite set, mass a mass function, D a domain identifier, and $X \in \mathbb{P}(\mathcal{X})$.*

- *The belief of X is the sum of masses of all (not necessarily proper) subsets of X*

$$Belief(D, X) = \sum_{B \subseteq X} mass(D, B) \tag{1}$$

- *The plausibility of X is the sum of the masses of all sets that are non-disjoint with X, and can be stated in terms of belief. Denoting the set complement of X as $\neg X$:*

$$Plaus(D, X) = 1 - Belief(D, \neg X) \tag{2}$$

From the above definitions, it can be seen that *Belief* and *Plaus* are capacities. Also, belief and plausibility are *conjuncts*, as it is also true that $Belief(D, A) = 1 - Plaus(D, \neg A)$.

[1] Since we restrict our attention to measures over finite sets, there is no loss of generality in the use of powersets rather than σ-algebras, as any σ-algebra over a finite set is isomorphic to the powerset of a set with smaller cardinality.

https://doi.org/10.1017/S1471068425100161 Published online by Cambridge University Press

A capacity space $(D, \mathcal{X}, \mathbb{P}(\mathcal{X}), \xi)$ for which ξ is a belief or plausibility function is a *belief domain*, and $\mathbb{P}(\mathcal{X})$ is called the *belief event space* of such a space.

Example 3.
We may represent a belief domain by the predicates domain/2 which represents the frame of discernment ($\mathcal{X}$) and mass/3 which represents the mass function. The belief domain of Example 1, which we call urn1, can be represented as:

$domain(urn1, \{blue, red, yellow\})$.

$mass(urn1, \{blue\}, 0.1)$. $mass(urn1, \{red\}, 0.3)$. $mass(urn1, \{blue, yellow\}, 0.6)$.

Here, $Belief(urn1, \{red, yellow\}) = 0.3$ *and* $Plaus(urn1, \{red, yellow\}) = 1 - Belief(urn1, \{blue\}) = 0.9$.

2.2 Probabilistic logic programs (PLPs)

Among several equivalent languages for PLP under the distribution semantics, we consider ProbLog (De Raedt *et al.* 2007) for simplicity of exposition. A *ProbLog program* $\mathcal{P} = (\mathcal{R}, \mathcal{F})$ consists of a finite set $\mathcal{R}$ of (certain) logic programming rules and a finite set $\mathcal{F}$ of *probabilistic facts* of the form $p_i :: f_i$ where $p_i \in [0, 1]$ and f_i is an atom, meaning that we have evidence of the truth of each ground instantiation $f_i\theta$ of f_i with probability p_i and of its negation with probability $1 - p_i$. Without loss of generality, we assume that atoms in probabilistic facts do not unify with the head of any rule and that all probabilistic facts are independent (cf. (Riguzzi, 2022)). Given a ProbLog program $\mathcal{P} = (\mathcal{R}, \mathcal{F})$, its *grounding* ($ground(\mathcal{P})$) is defined as $(ground(\mathcal{R}), ground(\mathcal{F}))$. With a slight abuse of notation, we sometimes use $\mathcal{F}$ to indicate the set of atoms f_i of probabilistic facts. The meaning of $\mathcal{F}$ will be clear from the context.

2.2.1 The distribution semantics for ProbLog programs without function symbols

For a ProbLog program $\mathcal{P} = (\mathcal{R}, \mathcal{F})$, a grounding $(\mathcal{R}, \mathcal{F}')$, such that $\mathcal{F}' \subseteq ground(\mathcal{F})$ is called a *world*. $W_{\mathcal{P}}$ denotes the set of all worlds. Whenever $\mathcal{P}$ contains no function symbols, $ground(\mathcal{F})$ is finite, so $W_{\mathcal{P}}$ is also finite. We also assume that each ProbLog program $\mathcal{P} = (\mathcal{R}, \mathcal{F})$ is *uniformly total*, that is that each world has a total well-founded model (Przymusinski, 1989), so in each world every ground atom is either true or false. The condition that a program is uniformly total is the most general way to express that every world is associated with a stratified program, so that the distribution semantics is well-defined. We define a measure on a set of worlds, and how to compute the probability for a query.

Definition 5
(Measures on Worlds and Sets of Worlds). $\rho_{\mathcal{P}} : W_{\mathcal{P}} \to \mathbb{R}$ *is a measure on worlds such that for each* $w \in W_{\mathcal{P}}$

$$\rho_{\mathcal{P}}(w) = \prod_{p::a \in \mathcal{F} | a \in w} p \prod_{p::a \in \mathcal{F} | a \notin w} (1 - p).$$

$\mu_{\mathcal{P}} : \mathbb{P}(W_{\mathcal{P}}) \to \mathbb{R}$ *is a measure on sets of worlds as such that for each* $\omega \in \mathbb{P}(W_{\mathcal{P}})$

$$\mu_{\mathcal{P}}(\omega) = \sum_{w \in \omega} \rho_{\mathcal{P}}(w).$$

https://doi.org/10.1017/S1471068425100161 Published online by Cambridge University Press

From Definition 5, $(W_\mathcal{P}, \mathbb{P}(W_\mathcal{P}), \mu_\mathcal{P})$ is a probability space and $\mu_\mathcal{P}$ a probability measure.

Definition 6.
Given a ground atom q, define $Q : W_\mathcal{P} \to \{0,1\}$ as

$$Q(w) = \begin{cases} 1 \text{ if } w \models q \\ 0 \text{ otherwise} \end{cases} \tag{3}$$

$w \models q$ means that q is true in the well-founded model of w.

Since $Q^{-1}(\gamma) \in \mathbb{P}(W_\mathcal{P})$ for all $\gamma \subseteq \{0,1\}$, and since the range of Q is measurable, Q is a random variable. The distribution of Q is defined by $P(Q=1)$ ($P(Q=0)$ is given by $1 - P(Q=1)$) and for a ground atom q we indicate $P(Q=1)$ by $P(q)$.
Given this discussion, we compute the probability of a ground atom q called *query* as

$$P(q) = \mu_\mathcal{P}(Q^{-1}(1)) = \mu_\mathcal{P}(\{w \mid w \in W_\mathcal{P}, w \models q\}) = \sum_{w \in W_\mathcal{P} \mid w \models q} \rho_\mathcal{P}(w).$$

That is, $P(q)$ is the sum of probabilities associated with each world in which q is true.

2.2.2 Computing queries to probabilistic logic programs

Let us introduce some terminology. An *atomic choice* indicates whether or not a ground probabilistic fact $p_i :: f_i$ is selected and is represented by the pair (f_i, k_i) where $k_i \in \{0,1\}$: $k_i = 1$ indicates that f_i is selected, $k_i = 0$ that it is not. A set of atomic choices is *consistent* if only one alternative is selected for any probabilistic fact, that is the set does not contain atomic choices $(f_i, 0)$ and $(f_i, 1)$ for any f_i. A *composite choice* κ is a consistent set of atomic choices.

Definition 7
(Measure of a Composite Choice). *Given a composite choice κ, we define the function ρ_c as*

$$\rho_c(\kappa) = \prod_{(f_i,1) \in \kappa} p_i \prod_{(f_i,0) \in \kappa} (1 - p_i).$$

A *selection* σ (also called a *total composite choice*) contains one atomic choice for every probabilistic fact. From the preceding discussion, it is immediate that a selection σ identifies a world w_σ. The *set of worlds ω_κ compatible with a composite choice κ* is $\omega_\kappa = \{w_\sigma \in W_\mathcal{P} \mid \kappa \subseteq \sigma\}$. Therefore, a composite choice identifies a set of worlds. Given a set of composite choices K, the *set of worlds ω_K compatible with K* is $\omega_K = \bigcup_{\kappa \in K} \omega_\kappa$. Two sets K_1 and K_2 of composite choices are *equivalent* if $\omega_{K_1} = \omega_{K_2}$, that is, if they identify the same set of worlds. If the union of two composite choices κ_1 and κ_2 is not consistent, then κ_1 and κ_2 are *incompatible*. We define as *pairwise incompatible* a set K of composite choices if $\forall \kappa_1 \in K, \forall \kappa_2 \in K$, $\kappa_1 \neq \kappa_2$ implies that κ_1 and κ_2 are incompatible. If K is a pairwise incompatible set of composite choices, define $\mu_c(K) = \sum_{\kappa \in K} \rho_c(\kappa)$.

Given a general set K of composite choices, we can construct a pairwise incompatible equivalent set through the technique of *splitting*. In detail, if f is a probabilistic fact and κ is a composite choice that does not contain an atomic choice (f, k) for any k, the *split* of κ on f can be defined as the set of composite choices $S_{\kappa,f} = \{\kappa \cup \{(f,0)\}, \kappa \cup \{(f,1)\}\}$.

https://doi.org/10.1017/S1471068425100161 Published online by Cambridge University Press

In this way, κ and $S_{\kappa,f}$ identify the same set of possible worlds, that is $\omega_\kappa = \omega_{S_{\kappa,f}}$, and $S_{\kappa,f}$ is pairwise incompatible. It turns out that, given a set of composite choices, by repeatedly applying splitting it is possible to obtain an equivalent mutually incompatible set of composite choices.

Theorem 1
((Poole, 2000)). *Let K be a set of composite choices. Then there is a pairwise incompatible set of composite choices equivalent to K.*

Theorem 2
((Poole, 1993) Measure Equivalence for Composite Choices). *If K_1 and K_2 are both pairwise incompatible sets of composite choices such that they are equivalent, then $\mu_c(K_1) = \mu_c(K_2)$.*

Theorem 3
(Probability Space of a Program). *Let $\mathcal{P}$ be a ProbLog program without function symbols and let $\Omega_\mathcal{P}$ be*

$$\{\omega_K \mid K \text{ is a set of composite choices}\}.$$

Then $\Omega_\mathcal{P} = \mathbb{P}(W_\mathcal{P})$ and $\mu_\mathcal{P}(\omega_K) = \mu_c(K')$ where K' is a pairwise incompatible set of composite choices equivalent to K.

A composite choice κ is an *explanation* for a query q if $\forall w \in \omega_\kappa : w \models q$. If the program is uniformly total, $w \models q$ is either true or false for every world w. Moreover, a set K of composite choices is *covering* with respect to a query q if every world in which q is true belongs to ω_K. Therefore, in order to compute the probability of a query q, we can find a covering set K of explanations for q, then make it mutually incompatible and compute the probability from it. This is the approach followed by ProbLog and PITA for example (Kimmig *et al.* 2011; Riguzzi and Swift, 2011).

3 Capacity logic programs

We now present *Capacity Logic Programs (CaLPs)* that extend PLPs to include belief domains (Section 2.1).

Definition 8
(CaLP rule). *Let $\mathcal{D}$ be a set of belief domains. A belief fact has the form belief(D_k, B) where $D_k \in \mathcal{D}$, and B is an element of the belief event space of D_k. We call the value of the first argument of a belief fact the domain and the value of the second argument the belief event.*[2]

A CaLP rule has the form h :– $l_0, \ldots, l_m, p_0, \ldots, p_n, b_0, \ldots, b_o$ where h is an atom, each $l_i, i \in \{0, \ldots, m\}$ is a literal, each $p_j, j \in \{0, \ldots, n\}$ is a probabilistic fact or its negation and each $b_k, k \in \{0, \ldots, o\}$ is a belief fact or its negation. A negated belief fact \+ belief(D, B) is equal to belief$(D, \neg B)$.

Belief facts should be distinguished from the belief function *Belief/2* of Definition 4. Note that belief facts alone are sufficient to specify a program's use of a belief domain,

[2] Alternatively, the belief event could be called evidence.

https://doi.org/10.1017/S1471068425100161 Published online by Cambridge University Press

since plausibility and belief are conjuncts, so each can be computed from the other. We assume that all belief domains are function-free. To simplify presentation, we also assume belief facts are ground.

Definition 9
(Capacity Logic Program). *Let $\mathcal{F}$ be a finite set of probabilistic facts and $\mathcal{R}$ a CaLP ruleset. Let $\mathcal{D}$ be a finite set of belief domains and $\mathcal{B}$ the set of ground belief facts for all belief events of all belief domains in $\mathcal{D}$. The tuple $\mathcal{P} = (\mathcal{R}, \mathcal{F}, \mathcal{D}, \mathcal{B})$ is called a Capacity Logic Program. $\mathcal{B}$ is called the belief set of $\mathcal{P}$.*

As notation, for a set $\mathcal{B}$ of belief facts and belief domain D, $facts(D, \mathcal{B})$ designates the set of belief facts in $\mathcal{B}$ that have domain D. In addition, $doms(\mathcal{B})$ denotes the set of all domains that are domain arguments of belief facts in $\mathcal{B}$.

When used in a belief world, a set of belief facts is made *canonical* to take into account that while belief facts from different belief domains are independent, belief facts from the same belief domain are not. Informally, if set of belief facts contains $belief(D_1, B_1)$ and $belief(D_1, B_2)$, these belief facts are replaced by $belief(D_1, B_1 \cap B_2)$, to ensure all belief facts are independent.

Definition 10
(Canonicalization of a Set of Belief Facts). *The canonicalization of a set of belief facts Bel is*

$$canon(Bel) = \{belief(D, B) \mid D \in doms(Bel) \text{ and } B = \bigcap_{belief(D,B_i) \in Bel} B_i\}$$

The set $canon(Bel)$ may contain facts whose belief event is $\emptyset$. If $canon(\mathcal{B})$ contains such facts it is inconsistent, otherwise it is consistent.

Example 4.
(Canonicalization of Sets of Belief Facts). *To further motivate canonicalization, consider the belief domain urn1 of Example 3 and consider a belief set containing belief(urn1,{blue}) and belief(urn1,{blue,yellow}). From an evidentiary perspective (cf. (Shafer, 1976)) the belief set contains evidence both that a blue ball was chosen and that a blue or yellow ball was chosen. Clearly, the evidence of a blue ball also provides evidence that a blue or yellow ball was chosen. Canonicalization takes the intersection of these two belief events, retaining only belief(urn1,{blue}). Alternatively, consider a belief set Bel_1 that contains both belief(urn1,{blue}) and belief(urn1,{red}). Since a ball drawn from an urn cannot be both blue and red, the belief of $belief(urn1, (\{blue\} \cap \{red\}))$ is 0 and $canon(Bel_1)$ is inconsistent.*

Definition 11
(Belief World). *For a ground CaLP $\mathcal{P} = (\mathcal{R}, \mathcal{F}, \mathcal{D}, \mathcal{B})$, a belief world $w = (\mathcal{R}, \mathcal{F}', \mathcal{D}, \mathcal{B}')$ is such that $\mathcal{F}' \subseteq \mathcal{F}$, and $\mathcal{B}' \subseteq \mathcal{B}$ is a consistent canonical belief set that contains a belief fact for every domain in $\mathcal{D}$. $W_{\mathcal{P}}^{\mathcal{D}}$ denotes the set of all belief worlds.*

Since $\mathcal{D}$ is a finite set of function-free belief domains, $W_{\mathcal{P}}^{\mathcal{D}}$ is finite. Also, for any world $(\mathcal{R}, \mathcal{F}', \mathcal{D}, \mathcal{B}') \in W_{\mathcal{P}}^{\mathcal{D}}$, $\mathcal{B}$ contains exactly one belief fact for every domain in $\mathcal{D}$: by Definition 11, $\mathcal{B}'$ contains at least one belief fact for a given $D_i \in \mathcal{D}$, while canonicalization ensures that there is at most one belief fact for D_i.

https://doi.org/10.1017/S1471068425100161 Published online by Cambridge University Press

As defined in Section 2.2, CaLPs are assumed to be uniformly total: that each world has a two-valued well-founded model (i.e., that each world gives rise to a stratified program). Thus, to extend the semantics of Section 2.2 it needs to be ensured that the well-founded model can be properly computed when belief facts are present, and for this the step of *completion* is used. Suppose w's belief set contained *belief(urn1,*{*blue*}*)* as the (unique) belief fact for the belief domain *urn1* of Example 3. Because $belief(urn1, \{blue\})$ provides evidence for $belief(urn1, \{blue, yellow\})$, a positive literal $belief(urn1, \{blue, yellow\})$ in a rule should succeed.

Definition 12
(Well-Founded Models for CaLP Programs). *Let* $w = (\mathcal{R}, \mathcal{F}, \mathcal{D}, \mathcal{B})$ *be a belief world. The completion of* $\mathcal{B}$, $comp(\mathcal{B})$ *is the smallest set such that if* $belief(\mathcal{D}, B_1) \in \mathcal{B}$, *and* B_2 *is an element of the belief event space of* D *such that* $B_1 \subseteq B_2$, *then* $B_2 \in comp(\mathcal{B})$. *The well-founded model of* w, $WFM(w)$ *is the well-founded model of* $\mathcal{R} \cup \mathcal{F} \cup comp(\mathcal{B})$. *For a ground atom* q, $w \models q$ *if* $q \in WFM(w)$.

Thus a belief set $\mathcal{B}$ must be canonical before being used to construct a world w. On the other hand, the completion of $\mathcal{B}$ is used when computing $WFM(w)$ and ensures rules requiring weak evidence (such as that a ball is yellow or blue) will be satisfied in a belief world that provides stronger evidence (such that a ball is blue).

Belief domains can be seen as a basis for imprecise probability (Halpern and Fagin, 1992), so the capacity of a belief world is an interval containing both belief and plausibility. Accordingly, circumflexed products and sums represent interval multiplication and addition. In the following definition, β_{prb} computes the portion of the capacity due to probabilistic facts (similar to Definition 5), and β_{blf} the portion due to belief facts.

Definition 13
(Capacity of a Belief World). $\beta_{prb}, \beta_{blf}, \beta : W^{\mathcal{P}}_{\mathcal{P}} \to \mathbb{R}^2$ *are capacities such that for* $w = (\mathcal{R}, \mathcal{F}, \mathcal{D}, \mathcal{B})$

$$\beta_{prb}(w) = \widehat{\prod}_{p::a \in w}[p, p] \widehat{\prod}_{p::a \notin w}[(1-p), (1-p)]$$

$$\beta_{blf}(w) = \widehat{\prod}_{belief(D,B) \in \mathcal{B}}[Belief(D, B), Plaus(D, B)]$$

$$\beta(w) = \beta_{prb}(w) \widehat{\times} \beta_{blf}(w)$$

The restriction that a world's belief set contain a belief fact for each belief domain need not affect the capacity of a world, since the fact may indicate trivial evidence for the entire discernment frame, for which the belief and plausibility are both 1. For instance, {*blue, red, yellow*} in the *urn*1 domain has a belief of 1.

To assign a non-additive capacity to a *set* of belief worlds, the super-additivity of belief and plausibility capacities must be addressed .[3] The following definition provides one step in this process.

Definition 14.
Let $\mathcal{S}$ *be a set of belief facts, and* $D \in doms(\mathcal{S})$. *The upper belief domain probability is* $BP^{\uparrow}(D, \mathcal{S}) = [1, 1]$ *if* $facts(D, \mathcal{S})$ *is empty; and*

[3] A capacity ξ is super-additive if for two sets A, B $\xi(A \cup B)$ may be greater than $\xi(A) + \xi(B)$.

https://doi.org/10.1017/S1471068425100161 Published online by Cambridge University Press

$[Belief(D, \bigcup B_i), Plaus(D, \bigcup B_i)]$ *for* $B_i \in facts(D, \mathcal{S})$ *otherwise*

Next, a set $\mathcal{B}$ of belief worlds is partitioned by grouping into cells worlds in $\mathcal{B}$ that have the same set of probabilistic facts. Within each cell, the probabilistic facts are factored out and the belief events for each domain are then unioned, leading to a single world and unique capacity for each partition cell. Finally the capacities of all cells are added. This process is necessary to avoid counting the mass of probabilistic facts more than once when the capacities of belief worlds are summed.

Definition 15
(Capacities of Sets of Belief Worlds). *Let* $\mathcal{P} = (\mathcal{R}, \mathcal{F}, \mathcal{D}, \mathcal{B})$. *The partition function* $DomPrtn(\mathcal{S}) : \mathbb{P}(W_{\mathcal{P}}^{\mathcal{D}}) \to \mathbb{P}(\mathbb{P}(W_{\mathcal{P}}^{\mathcal{D}}))$, *given* $\mathcal{S} \in \mathbb{P}(W_{\mathcal{P}}^{\mathcal{D}})$ *produces the domain partition of* $\mathcal{S}$*: the coarsest partition such that for each* $S_i \in DomPrtn(\mathcal{S})$:

$$(\mathcal{R}, \mathcal{F}_j, \mathcal{D}, \mathcal{B}_j) \in S_i \text{ and } (\mathcal{R}, \mathcal{F}_k, \mathcal{D}, \mathcal{B}_k) \in S_i \Rightarrow \mathcal{F}_j = \mathcal{F}_k.$$

Using the domain partition, we define $probfacts(S_i)$ *as the unique* $\mathcal{F}'$ *for any* $w = (\mathcal{R}, \mathcal{F}', \mathcal{D}, \mathcal{B}')$ *s.t.,* $w \in S_i$. *The capacity* β_{prtn} *for* $S_i \in DomPrtn(\mathcal{S})$ *is*

$$\beta_{prtn}(S_i) = \beta_{prb}(probfacts(S_i)) \widehat{\times} \widehat{\prod}_{(\mathcal{R},\mathcal{F},\mathcal{D},\mathcal{B}) \in \mathcal{S}_i; D \in doms(\mathcal{B})} BP^{\uparrow}(D, \mathcal{B}). \tag{4}$$

Finally, the capacity for $\mathcal{S}$ *is*

$$\xi^{\mathcal{B}}(\mathcal{S}) = \widehat{\sum}_{S_i \in DomPrtn(\mathcal{S})} \beta_{prtn}(S_i). \tag{5}$$

Example 5.
(Computing the Capacity for a Set of Belief Worlds). *Consider a new domain urn*2 *analogous to urn*1 *of Example 3*:

domain(urn2,{green,orange,purple})
mass(urn2,{green},0.1) mass(urn2,{orange},0.3) mass(urn2,{green,purple},0.6)
Let CaLP $\mathcal{P} = (\mathcal{R}, \mathcal{F}, \mathcal{D}, \mathcal{B})$ *where* $\mathcal{F} = \{p_1 :: f_1, p_2 :: f_2\}$ *and* $\mathcal{D} = \{urn1, urn2\}$, *along with a set* $\mathcal{S}$ *of 3 worlds of* $\mathcal{P}$ *with the following fact sets and belief domains.*
$w_1 : \mathcal{F}_1 = \{p_1\}, \mathcal{B}_1 = \{belief(urn1, \{blue\})\}$ $w_2 : \mathcal{F}_2 = \{p_2\}, \mathcal{B}_2 = \{belief(urn2, green)\}$
$w_3 : \mathcal{F}_3 = \{p_1\}, \mathcal{B}_3 = \{belief(urn1, \{yellow\})\}$
Following Definition 15, $DomPrtn(\mathcal{S})$ *partitions* $\mathcal{S}$ *into* $\{w_1, w_3\}$ *and* $\{w_2\}$. *To compute* $\beta_{prtn}(\{w_1, w_3\})$, $BP^{\uparrow}$ *is calculated for each belief domain giving*

$$[Belief(urn1, \{blue, yellow\}), Plaus(urn1, \{blue, yellow\}]) = [0.7, 0.7]$$

for the domain urn1 and $[1, 1]$ *for the domain urn*2. *Next,* $\beta_{prb}(\{w_1, w_3\})$ *is calculated as* $[p_1, p_1] \widehat{\times} [(1 - p_2), (1 - p_2)]$ *and multiplied with* $[0.7, 0.7]$. *The result is then summed with* $\beta_{prtn}(\{w_2\})$.

One might ask why Definition 15 uses $BP^{\uparrow}$ to combine beliefs of the same domain rather than Dempster's Rule of Combination. Dempster's rule was designed to combine independent evidence to determine a precise combined belief. This approach is useful for some purposes but has been criticized for many others (cf. (Jøsang, 2016)). When combining CaLP worlds, epistemic commitment for a given domain is reduced by $BP^{\uparrow}$ in a manner similar to combining CaLP explanations (Section 4).

https://doi.org/10.1017/S1471068425100161 Published online by Cambridge University Press

Since the range of $\xi^{\mathcal{B}}$ is $\mathbb{R}^2$, we designate $proj_B, proj_P : \mathbb{R}^2 \to \mathbb{R}$ as projection functions for the belief and plausibility portions, respectively. The following theorem states that capacity spaces formed using belief worlds and the capacity function $\xi^{\mathcal{B}}$ have belief and plausibility capacities whose value is 1 for the entire belief event space.

Theorem 4
Let $\mathcal{P} = (\mathcal{R}, \mathcal{F}, \mathcal{D}, \mathcal{B})$ *be a CaLP. Then,* $(W^{\mathcal{D}}_{\mathcal{P}}, \mathbb{P}(W^{\mathcal{D}}_{\mathcal{P}}), proj_B(\xi^{\mathcal{B}}))$ *and* $(W^{\mathcal{D}}_{\mathcal{P}}, \mathbb{P}(W^{\mathcal{D}}_{\mathcal{P}}), proj_P(\xi^{\mathcal{B}}))$ *are normalized capacity spaces.*

Proof.
From Definitions 2 and 15 it is clear that $\xi^{\mathcal{B}}$ is a capacity. To show that $\xi^{\mathcal{B}}$ is normalized, consider that $\xi^{\mathcal{B}}$ is a function composition, where the first function, $DomPrtn(W^{\mathcal{D}}_{\mathcal{P}})$ returns the domain partition of $W^{\mathcal{D}}_{\mathcal{P}}$, while $\beta_{prb}(probfacts(S_i))$ is the separate probability of $\mathcal{F}$. Consider a cell $\mathcal{S}_i \in DomPrtn(\mathcal{S})$. From equation (4)

$$\beta_{prtn}(S_i) = \beta_{prb}(probfacts(S_i)) \widehat{\times} \widehat{\prod}_{(\mathcal{R},\mathcal{F},\mathcal{B}) \in \mathcal{S}_i; D \in doms(\mathcal{B})} BP^{\uparrow}(\mathcal{D}, B)$$

Every world in $\mathcal{S}_i$ contains the same set of probabilistic facts, $(probfacts(S_i))$, but differs in belief facts. Let D be a belief domain in $\mathcal{D}$. Since $\mathcal{S}_i$ is a partition of $W^{\mathcal{D}}_{\mathcal{P}}$, and because all belief domains in all worlds are normalized, S_i contains a world with every belief event of D. Thus $BP^{\uparrow}(D, B) = [1, 1]$ so that

$$\beta_{prtn}(S_i) = \beta_{prb}(probfacts(S_i)) \widehat{\times} \widehat{\prod}_{(\mathcal{R},\mathcal{F},\mathcal{B}) \in \mathcal{S}_i; D \in doms(\mathcal{B})} [1, 1] = \beta_{prb}(probfacts(S_i))$$

Thus, the value of $\beta_{prtn}(\mathcal{S}_i)$ relies only on the probabilistic facts that were used for the partition. Since $W^{\mathcal{D}}_{\mathcal{P}}$ contains all possible worlds, $\xi^{\mathcal{B}}(W^{\mathcal{D}}_{\mathcal{P}})$ reduces to the sum of all subsets of probabilistic facts of $\mathcal{P}$, which is 1. □

Definition 16
(cf. Section 2.2, Definition 6). *Given a ground atom* q *in* $ground(\mathcal{P})$ *we define* $Q_{\mathcal{B}} : W^{\mathcal{D}}_{\mathcal{P}} \to \{0, 1\}$ *as*

$$Q_{\mathcal{B}}(w) = \begin{cases} 1 \text{ if } w \vDash q \text{ (}Definition\ 12\text{)} \\ 0 \text{ otherwise} \end{cases} \tag{6}$$

By definition $Q^{-1}_{\mathcal{B}}(\gamma) \in \mathbb{P}(W^{\mathcal{D}}_{\mathcal{P}})$ for $\gamma \subseteq \{0, 1\}$, and since the range of $Q_{\mathcal{B}}$ is measurable, $Q_{\mathcal{B}}$ is a capacity-based random variable.

Given the above discussion, we compute $[belief(q), plaus(q)]$ for a ground atom q as

$$[belief(q), plaus(q)] = \xi^{\mathcal{B}}(Q^{-1}_{\mathcal{B}}(1)) = \xi^{\mathcal{B}}(\{w \mid w \in W^{\mathcal{D}}_{\mathcal{P}}, w \vDash q\}).$$

4 Computing queries to capacity logic programs

We extend the definition of atomic choice from Section 2.2.2 to CaLPs as follows.

Definition 17
(Capacity Atomic Choice (CaLPs)). *For a CaLP* $\mathcal{P}^{\mathcal{B}} = (\mathcal{R}, \mathcal{F}, \mathcal{D}, \mathcal{B})$ *a capacity atomic choice is either:*

https://doi.org/10.1017/S1471068425100161 Published online by Cambridge University Press

- *An atomic Bayesian choice of ground probabilistic fact* $p :: f$, *represented by the pair* (f, k) *where* $k \in \{0, 1\}$. $k = 1$ *indicates that* f *is selected,* $k = 0$ *that it is not.*
- *An atomic belief choice of a ground belief fact belief(*D,B*) where* D *is a belief domain in* $\mathcal{D}$ *and* B *is a belief event in* D. *The choice is represented as belief(*D,B*).*

Atomic belief choices with different belief domains are considered independent choices, much as atomic Bayesian choices. However, atomic belief choices with the same belief domain are *not* independent: so fitting them into the distribution semantics requires care.

Given a set $\mathcal{S}$ of atomic choices, let $Bay(\mathcal{S})$ be the subset of $\mathcal{S}$ that contains only the atomic Bayesian choices of $\mathcal{S}$ and let $Bel(\mathcal{S})$ be the subset of $\mathcal{S}$ that contains only the atomic belief choices of $\mathcal{S}$. Note that $Bel(\mathcal{S})$ is a set of belief facts, so that definitions of Section 3 can be used directly or with minimal change. Then $\mathcal{S}$ is consistent if both $Bay(\mathcal{S})$ and $Bel(\mathcal{S})$ are consistent (cf. Definition 10). A *capacity composite choice* κ is a consistent set of atomic choices for a CaLP $\mathcal{P}^{\mathcal{B}} = (\mathcal{P}, \mathcal{F}, \mathcal{D}, \mathcal{B})$. Given a capacity composite choice κ, let $doms(\kappa)$ be $\{D \mid belief(D, B) \in \kappa\}$.

Definition 18
(Measure for a Single Capacity Composite Choice). *Given a capacity composite choice* κ *we define* $\rho_{\mathcal{B}}^{Comp}$ *as*

$$\rho_{\mathcal{B}}^{Comp}(\kappa) = \rho_{\mathcal{P}^{\mathcal{B}}}(\kappa) \widehat{\times} \widehat{\prod}_{belief(D,Elt) \in canon(Bel(\kappa)); \mathcal{D} \in doms(\kappa)} BP^{\uparrow}(D, Elt)$$

where $\rho_{\mathcal{P}^{\mathcal{B}}}$ *is the measure for composite choices from Definition 7.*

The *set of worlds* ω_κ *compatible with a capacity composite choice* κ is

$$\omega_\kappa = \{w_\sigma = (\mathcal{R}, \mathcal{F}, \mathcal{D}, \mathcal{B}) \in W_{\mathcal{P}}^{\mathcal{D}} \mid Bay(\kappa) \subseteq \mathcal{F} \text{ and } canon(Bel(\kappa)) \subseteq \mathcal{B}\}.$$

With this definition of worlds compatible with a capacity composite choice, the definitions of equivalence of two sets of capacity composite choices, of incompatibility of two atomic choices and of a pairwise incompatible set of capacity composite choices are the same as for probabilistic logic programs.

If K is a pairwise incompatible set of capacity composite choices, define $\xi_c(K) = \sum_{\kappa \in K} \rho_{\mathcal{B}}^{Comp}(\kappa)$. Given a general set K of capacity composite choices, we can construct a pairwise incompatible equivalent set through the technique of *splitting*. In detail, (1) if f is a probabilistic fact and κ is a capacity composite choice that does not contain an atomic Bayesian choice (f, k) for any k, the *split* of κ on f, $S_{\kappa,f}$, is defined as for probabilistic logic programs; (2) if D is a domain not appearing in $doms(\kappa)$ and B a belief event of D, the *split* of κ on $belief(D, B)$, is defined as the set of capacity composite choices $S_{\kappa,D,B} = \{\kappa \cup \{belief(D, B)\}, \kappa \cup \{belief(D, \neg B)\}\}$. In this way, κ and $S_{\kappa,f}$ ($S_{\kappa,D,B}$) identify the same set of possible worlds, that is $\omega_\kappa = \omega_{S_{\kappa,f}}$ ($\omega_\kappa = \omega_{S_{\kappa,D,B}}$) and $S_{\kappa,f}$ ($S_{\kappa,D,B}$) are pairwise incompatible.

As for probabilistic logic programs, given a set of capacity composite choices, by repeatedly applying splitting it is possible to obtain an equivalent mutually incompatible set of composite choices (Poole, 2000) and the analogous of Theorem 1 can be proved.

Theorem 5
Let K *be a finite set of capacity composite choices. Then there is a pairwise incompatible set of capacity composite choices equivalent to* K.

https://doi.org/10.1017/S1471068425100161 Published online by Cambridge University Press

Proof.
Given a set of capacity composite choices K, there are two possibilities to form a new set K' of capacity composite choices so that K and K' are equivalent:

1. **Removing dominated elements**: if $\kappa_1, \kappa_2 \in K$ and $\kappa_1 \subset \kappa_2$, let $K' = K \setminus \{\kappa_2\}$.
1. **Splitting elements**: if $\kappa_1, \kappa_2 \in K$ are compatible (and neither is a superset of the other), there is a Bayesian choice $(f, k) \in \kappa_1 \setminus \kappa_2$ or a belief choice $\mathit{belief}(D, B) \in \kappa_1 \setminus \kappa_2$ We replace κ_2 by the split of κ_2 on f. Let $K' = K \setminus \{\kappa_2\} \cup S_{\kappa_2, f}$.

In both cases, $\omega_K = \omega_{K'}$. If we repeat this two operations until neither is applicable, we obtain a splitting algorithm that terminates because K is finite. The resulting set K' is pairwise incompatible and is equivalent to the original set. □

Theorem 6
(Capacity Equivalence for Capacity Composite Choices). *If K_1 and K_2 are both pairwise incompatible sets of capacity composite choices such that they are equivalent, then* $\xi_c(K_1) = \xi_c(K_2)$.

Proof.
The proof of this theorem is analogous to that of Theorem 2 given by Poole (1993). That proof hinges on the fact that the probabilities of $(f_i, 0)$ and $(f_i, 1)$ sum to 1. We must similarly prove that the belief and plausibility of $\mathit{belief}(D, B)$ and $\mathit{belief}(D, \neg B)$ sum to 1. Let us prove it for the belief: the first component of $\rho_{\mathcal{B}}^{Comp}(\{\mathit{belief}(D, B)\})$ is $\mathit{Belief}(D, B) = \sum_{X \subseteq B} \mathit{mass}(D, X)$ while of $\rho_{\mathcal{B}}^{Comp}(\{\mathit{belief}(D, \neg B)\})$ is $\mathit{Belief}(D, \neg B) = \sum_{X \subseteq \neg B} \mathit{mass}(D, X)$. Clearly the two sets $\{X \mid X \subseteq B\}$ and $\{X \mid X \subseteq \neg B\}$ partition the set of focal sets and, since the masses of focal sets sums to 1, $\mathit{Belief}(D, B) + \mathit{Belief}(D, \neg B) = 1$. For plausibility the reasoning is similar. □

Theorem 7
(Normalized Capacity Space of a Program). *Let $\mathcal{P}^{\mathcal{B}} = (\mathcal{R}, \mathcal{F}, \mathcal{D}, \mathcal{B})$ be a CaLP without function symbols and let $\Omega_{\mathcal{P}^{\mathcal{B}}} = \{\omega_K \mid K$ is a set of capacity composite choices$\}$. Then $\Omega_{\mathcal{P}^{\mathcal{B}}} = \mathbb{P}(W_{\mathcal{P}}^{\mathcal{D}})$ and $\xi^{\mathcal{B}}(\omega_K) = \xi_c(K')$ where K' is a pairwise incompatible set of capacity composite choices equivalent to K.*

Proof.
The fact that $\Omega_{\mathcal{P}^{\mathcal{B}}} \subseteq \mathbb{P}(W_{\mathcal{P}}^{\mathcal{D}})$ is due to the fact that $\mathbb{P}(W_{\mathcal{P}}^{\mathcal{D}})$ contains any possible set of worlds. In the other direction, given a set of worlds $\omega = \{w_1, \ldots, w_n\}$, we can build the set of capacity composite choices $K = \{\kappa_1, \ldots, \kappa_n\}$ where κ_i is built so that $\omega_{\kappa_i} = \{w_i\}$: let $w_i = (\mathcal{R}, \mathcal{F}, \mathcal{D}, \mathcal{B})$, for every probabilistic fact f, if $f \in \mathcal{F}$ then $(f, 1) \in \kappa_i$, and if $f \notin \mathcal{F}$ then $(f, 0) \in \kappa_i$; for every belief fact $\mathit{belief}(D, B) \in \mathit{canon}(\mathcal{B})$ then $\mathit{belief}(D, B) \in \kappa_i$.

To prove $\xi^{\mathcal{B}}(\omega_K) = \xi_c(K')$, it is enough to select as K' the set K built as above. □

The definitions of explanations and covering sets of explanations for a query q are the same as for PLP. As a result, systems like ProbLog and Cplint/PITA can be extended to inference over CaLPs: first a covering set of explanations is found using a conversion to propositional logic (ProbLog (Kimmig *et al.* 2011)) or a program transformation plus tabling and answer subsumption (PITA (Riguzzi and Swift, 2011)), then knowledge compilation is used to convert the explanations into an intermediate representation

https://doi.org/10.1017/S1471068425100161 Published online by Cambridge University Press

(d-DNNF for ProbLog and BDD for PITA) that makes them pairwise incompatible and allows the computation of the belief/plausibility.

Example 6.
To illustrate these topics, consider `r_indep` *that uses belief domains urn1 and urn2.*
`r_indep:-belief(urn1,{blue}). r_indep:- belief(urn2,{orange}).`
Since the belief domains (from Examples 3 and 5) are independent, a covering and pairwise incompatible set of explanations for `r_indep` *is*
$K = \{\ \{belief(urn1, \{blue\}), belief(urn2, \neg\{orange\})\},$
$\{belief(urn1, \neg\{blue\}), belief(urn2, \{orange\})\}$
$\{belief(urn1, \{blue\}), belief(urn2, \{orange\})\}\}$
whose capacity is:

$$\xi_c(K) = ([0.1, 0.7] \widehat{\times} [0.7, 0.7]) \ \widehat{+}\ ([0.3, 0.9] \widehat{\times} [0.3, 0.3]) \ \widehat{+}\ ([0.1, 0.7] \widehat{\times} [0.3, 0.3]) = [0.19, 0.97]$$

Next, consider `r_dep`*, both of whose rules both use the urn1 belief domain*
`r_dep:- belief(urn1,{blue}). r_dep:- belief(urn1,{red})`
The capacity of `r_dep` *is:*
$\xi_c(\{belief(urn1, \{blue\}), belief(urn1, \{red\})\}) = \xi_c(\{belief(urn1, \{blue, red\})\}) = [0.4, 1]$.

Towards a transformation based implementation.

We now sketch an implementation of inference in CaLPs using a PITA-like transformation (Riguzzi and Swift, 2011). We designate as PITAC a new transformation of a CaLP into a normal program. The normal program produced contains calls for manipulating BDDs via the `bddem` library (https://github.com/friguzzi/bddem):

- *zero/1*, *one/1*, *and/3*, *or/3* and *not/2*: Boolean operations between BDDs;
- *get_var_n*$(R, S, Probs, Var)$: returns an integer indexing a variable with $|Probs|$ values and parameters *Probs* (a list) associated to clause R with grounding S;
- *equality*$(+Var, +Value, -BDD)$: returns a BDD representing the equality $Var = Value$, that is that variable with index *Var* is assigned *Value* in the BDD;
- $mc(+BDD, -P)$: returns the model count of the formula encoded by *BDD*.

PITAC differs from PITA because it associates each ground atom with three BDDs, for the probability, belief, and plausibility, respectively, instead of one. The transformation for an atom a and a variable D, $PITAC(a, D)$, is a with the variable D added as the last argument. Variable D will contain triples (PrD, BeD, PlD) with the three BDDs.

A probabilistic fact $C_r = h : \Pi$, is transformed into the clause
$PITAC(C_r) = PITAC(h, D) \leftarrow get\_var\_n(r, S, [\Pi, 1 - \Pi], Var), equality(Var, 1, D)$.
A belief fact $C_r = belief(\mathcal{D}, B_i)$, is transformed into the clause
$PITAC(C_r) = PITAC(h, D) \leftarrow get\_var\_n(r, S, [\Pi_1, \ldots, \Pi_n], Var), equality(Var, i, D)$.
where the program contains the set of facts $\{mass(\mathcal{D}, B_j, \Pi_j)\}_{j=1}^{n}$. The transformation for clauses is the same as for PITA.

https://doi.org/10.1017/S1471068425100161 Published online by Cambridge University Press

In order to answer queries, the goal *cap(Goal,Be,Pl)* is used, which is defined by

$$cap(Goal, Be, Pl) \leftarrow add\_arg(Goal, D, GoalD),$$
$$(call(GoalD) \rightarrow DD = (PrD, BeD, PlD),$$
$$mc(PrD, Pr), mc(BeD, Bel), mc(PlD, Pla), Be\ is\ Pr \cdot Bel, Pl\ is\ Pr \cdot Pla;$$
$$Be = 0.0, Pl = 0.0).$$

where $add\_arg(Goal, D, GoalD)$ returns $PITAC(Goal, D)$ in $GoalD$.

This sketch must be refined by taking into account also the need for canonicalization and partitioning.

5 Related work

Among the possible semantics for probabilistic logic programs, such as CP-logic (Meert *et al.* 2010), Bayesian Logic Programming (Kersting and De Raedt, 2001), CLP(BN) (Costa *et al.* 2003), Prolog Factor Language (Gomes and Costa, 2012), Stochastic Logic Programs (Muggleton *et al.* 1996), and ProPPR (Wang *et al.* 2015), none of them, to the best of our knowledge, consider belief functions.

Other semantics, still not considering belief functions, target Answer Set Programming (Brewka et al, 2011), such as LPMLN (Lee and Yang, 2017), P-log (Baral *et al.* 2009), and the credal semantics (CS) (Cozman and Mauá, 2017). The latter deserves more attention: given an answer set program extended with probabilistic facts, the probability of a query q is computed as follows. First worlds are computed, as in the distribution semantics. Each world w now is an answer set program which may have zero or more stable models. The atom q can be present in no models of w, in some models, or in every model. In the first case, w makes no contribution to the probability of q. If q is present in *some* but not all models of w, $P(w)$ is computed as in Equation 5, and contributes to the *upper probability* of q. If instead the query is present in *every* model of w, $P(w)$ contributes to both the upper *and lower probability*. That is, a query no longer has a point-probability but is represented by an interval. Furthermore, Cozman and Mauá (2017) also proved the CS is characterized by the set of all probability measures that dominate an infinitely monotone Choquet capacity.

The approach of Wan and Kifer (2009) incorporates belief functions into logic programming, but unlike ours is not based on the distribution semantics. Rather, belief and plausibility intervals are associated with rules. Within a model M, the belief of an atom A is a function of the belief intervals associated with A and the support of A in M.

6 Discussion

Let us now conclude the paper with a discussion about an extension of Example 2.

Example 7.
(Reasoning with Beliefs and Probabilities). *Suppose that the UAV from Example 2 is extended to search for stolen vehicles, alerting police when a match of sufficient strength is detected. The UAV can identify the type of a vehicle using the visual model* V_{mod} *of Example 2, although* V_{mod} *alone is not sufficient for this task. The UAV can also*

https://doi.org/10.1017/S1471068425100161 Published online by Cambridge University Press

```
stolen(Veh,VObj):- unusualType(Veh,Type),detection(VObj,VLoc),inArea(VLoc),
                   belief(V_mod,[VObj,Type]).
stolen(Veh,VObj):- rfid(VObj,DSigl,Loc),inArea(Loc),
                   targetSignal(TSig),signalMatch(DSig,TSig).
n::inArea(Loc):-   External function of last known location and time.
n::signalMatch(Sig1,Sig2):-External function to match RFIDs.
belief(Model,[Obj,Type]):- External function to match RFIDs.
```

Fig 2. UAV rules for stolen vehicles.

detect signals from strong RFID transponders but since it flies at a distance, it may detect several RFID signals in the same area, leading to uncertainty. Finally, the UAV considers only vehicles that are within a region determined by the last reported location of the vehicle and the time since the report.

Figure 2 provides rules and pseudo-code for the UAV's task. The use of belief and probabilistic facts in these rules emulates an application program's. The first rule checks whether the make of the stolen vehicle, `Veh`*, is unusual. If so, the probability that the location of the detected object* `VObj` *is within the search area is determined, and combined with the belief and plausibility that* `VObj` *is consistent with that of* `Veh`*. In the second rule, if a detected RFID signal is probabilistically in the search area, a probabilistic match is made between the signal for* `VObj` *and* `Veh`*.*

In this paper, we proposed an extension to the distribution semantics to handle belief functions and introduced the *Capacity Logic Programs* framework. We precisely characterize that framework with a set of theorems, and show how to compute the probability of queries. Since CaLPs extend PLPs, exact inference in CaLPs is #P-hard while thresholded inference is PP-hard (cf. (De Raedt et al, 2007; Cozman and Mauá, 2017)), although the upper bounds for these problems have not yet been determined. As future work, we also plan to provide a practical implementation of our framework by extending the cplint/PITA reasoner as outlined in Section 4.

Acknowledgments

This work has been partially supported by Spoke 1 "FutureHPC & BigData" of the Italian Research Center on High-Performance Computing, Big Data and Quantum Computing (ICSC) funded by MUR Missione 4 - Next Generation EU (NGEU). DA and FR are members of the Gruppo Nazionale Calcolo Scientifico – Istituto Nazionale di Alta Matematica (GNCS-INdAM).

References

Ash, R. and Doléans-Dade, C. 2000. *Probability and Measure Theory*. Harcourt/Academic Press.

Baral, C., Gelfond, M. and Rushton, N. 2009. Probabilistic reasoning with answer sets. *Theory and Practice of Logic Programming* 9, 1, 57–144.

Brewka, G., Eiter, T. and Truszczyński, M. 2011. Answer set programming at a glance. *Communications of the ACM* 54, 12, 92–103.

Choquet, G. (1954) Theory of capacities. *Annales De l'institut Fourier* 5, 131–295.

https://doi.org/10.1017/S1471068425100161 Published online by Cambridge University Press

Costa, V. S., Page, D., Qazi, M. and Cussens, J. 2003. CLP(BN): Constraint logic programming for probabilistic knowledge. In 19th International Conference on Uncertainty in Artificial Intelligence (UAI 2003), Morgan Kaufmann, 517–524.

Cozman, F. 2000. Credal networks. *Artificial Intelligence* 120, 199–233.

Cozman, F. G. and Mauá, D. D. 2017. On the semantics and complexity of probabilistic logic programs. *Journal of Artificial Intelligence Research* 60, 221–262.

De Raedt, L., Kimmig, A. and Toivonen, H. (2007) ProbLog: a probabilistic prolog and its application in link discovery, *20th International Joint Conference On Artificial Intelligence (IJCAI 2007)*, Veloso, M. M., Ed., 7, AAAI Press, 2462–2467.

Giunchiglia, E. and Lukasiewicz, T. (2020) Coherent hierarchical multi-label classification networks. In *Advances in Neural Information Processing Systems*, vol. 33. Curran Associates, Inc, 9662–9673.

Gomes, T. and Costa, V. S. (2012) Evaluating inference algorithms for the prolog factor language. In *21st International Conference On Inductive Logic Programming (ILP 2012)*, Riguzzi, F. and Železný, F., Eds. Lecture Notes in Computer Science, vol. 7842, Springer, 74–85,

Grabish, M. 2016. *Set Functions, Games, and Capacities in Decision Making*. Springer.

Guo, C., Pleiss, G., Sun, Y. and Weinberger, K. 2017. On calibration of modern neural networks. In *Proc. of the 34th International Conference on Machine Learning, ICML 2017*, PMLR.

Halpern, J. and Fagin, R. 1992. Two views of belief: belief as generalized probability and helief as evidence. *Artificial Intelligence* 54, 275–312.

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

Kersting, K. and De Raedt, L. 2001. Towards combining inductive logic programming with bayesian networks. In *11th International Conference on Inductive Logic Programming (ILP 2001)*, Springer, 118–131.

Kimmig, A., Demoen, B., De Raedt, L., Costa, V. S. and Rocha, R. 2011. On the implementation of the probabilistic logic programming language ProbLog. *Theory and Practice of Logic Programming* 11, 2–3, 235–262.

Lee, J. and Yang, Z. 2017. LPMLN, weak constraints, and P-log. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017*, Singh, S. and Markovitch, S., Eds. AAAI Press, 1170–1177.

Meert, W., Struyf, J. and Blockeel, H. (2010) CP-logic theory inference with contextual variable elimination and comparison to BDD based inference methods. In *Inductive Logic Programming*, De Raedt, L., Ed. Lecture Notes in Computer Science, vol. 5989, Springer, 96–109.

Muggleton, S., et al. 1996. Stochastic logic programs. *Advances in Inductive Logic Programming* 32, 254–264.

Poole, D. 1993. Probabilistic Horn abduction and Bayesian networks. *Artificial Intelligence* 64, 1, 81–129.

Poole, D. 2000. Abducing through negation as failure: stable models within the independent choice logic. *The Journal of Logic Programming* 44, 1–3, 5–35.

Przymusinski, T. C. 1989. Every logic program has a natural stratification and an iterated least fixed point model. In *Proceedings of the Eighth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*. PODS '89, Association for Computing Machinery, 11–21.

Riguzzi, F. 2022. *Foundations of Probabilistic Logic Programming Languages, Semantics, Inference and Learning*, 2nd ed. River Publishers.

https://doi.org/10.1017/S1471068425100161 Published online by Cambridge University Press

Riguzzi, F. and Swift, T. 2011. The PITA system: tabling and answer subsumption for reasoning under uncertainty. *Theory and Practice of Logic Programming* 11, 4–5, 433–449.

Sato, T. 1995. A statistical learning method for logic programs with distribution semantics. In *ICLP'95: Proceedings of the 12th International Conference on Logic Programming*, Sterling, L., Ed. MIT Press, 715–729.

Shafer, G. 1976. *A Mathematical Theory of Evidence*. Princeton University Press.

Wan, H. and Kifer, M. (2009) Belief logic programming: uncertainty reasoning with correlation of evidence. In *Logic Programming and Nonmonotonic Reasoning*, Springer Berlin Heidelberg, 316–328.

Wang, W. Y., Mazaitis, K., Lao, N. and Cohen, W. W. 2015. Efficient inference and learning in a large knowledge base: reasoning with extracted information using a locally groundable first-order probabilistic logic. *Machine Learning* 100, 101–126.

Weichselberger, K. 2000. The theory of interval probability as a unifying concept for uncertainty in knowledge-based systems. *International Journal of Approximate Reasoning* 24, 149–170.