Managing Uncertainty and Vagueness in Description Logics, Logic Programs and Description Logic Programs

Umberto Straccia

ISTI-CNR, Via G. Moruzzi 1, I-56124 Pisa, Italy
straccia@isti.cnr.it

Guide to ReasoningWeb 2008 Summer School

Abstract. Managing uncertainty and/or vagueness is starting to play an important role in Semantic Web representation languages. Our aim is to overview basic concepts on representing uncertain and vague knowledge in current Semantic Web ontology and rule languages (and their combination).

1 Introduction

The management of uncertainty and/or vagueness is an important issue whenever the real world information to be represented is of imperfect nature, which likely occurs in Semantic Web tasks. In this work we overview the relevant work in the context of Description Logics [6], Logic Programs [143] and their combination. This work should act as a reference/citation guide to the relevant literature, and, thus, we keep the formal level to a minimum.

2 Uncertainty and Vagueness Basics

There has been a long-lasting misunderstanding in the literature of artificial intelligence and uncertainty modeling, regarding the role of probability/possibility theory and vague/fuzzy theory. A clarifying paper is [64]. We recall here salient notes, which may clarify the role of these theories for the inexpert reader.

A standard example that points out the difference between degrees of uncertainty and degrees of truth is that of a bottle [64]. In terms of binary truth values, a bottle is viewed as full or empty. But if one accounts for the quantity of liquid in the bottle, one may e.g. say that the bottle is “half-full”. Under this way of speaking, “full” becomes a fuzzy predicate [294] and the degree of truth of “the bottle is full” reflects the amount of liquid in the bottle. The situation is quite different when expressing our ignorance about whether the bottle is either full or empty (given that we know that only one of the two situations is the true one). Saying that the probability that the bottle is full is 0.5 does not mean that the bottle is half full.

We recall that under uncertainty theory fall all those approaches in which statements rather than being either true or false, are true or false to some probability or possibility
(for example, “it will rain tomorrow”). That is, a statement is true or false in any world, but we are “uncertain” about which world to consider as the right one, and thus we speak about e.g. a probability distribution or a possibility distribution over the worlds. For example, we cannot exactly establish whether it will rain tomorrow or not, due to our incomplete knowledge about our world, but we can estimate to which degree this is probable, possible, and necessary.

As for the main differences between probability and possibility theory, the probability of an event is the sum of the probabilities of all worlds that satisfy this event, whereas the possibility of an event is the maximum of the possibilities of all worlds that satisfy the event. Intuitively, the probability of an event aggregates the probabilities of all worlds that satisfy this event, whereas the possibility of an event is simply the possibility of the “most optimistic” world that satisfies the event. Hence, although both probability and possibility theory allow for quantifying degrees of uncertainty, they are conceptually quite different from each other. That is, probability and possibility theory represent different facets of uncertainty.

On the other hand, under vagueness/fuzziness theory fall all those approaches in which statements (for example, “the tomato is ripe”) are true to some degree, which is taken from a truth space. That is, an interpretation maps a statement to a truth degree, since we are unable to establish whether a statement is completely true or false due to the involvement of vague concepts, such as “ripe”, which only have an imprecise definition. For example, we cannot exactly say whether a tomato is ripe or not, but rather can only say that the tomato is ripe to some degree. Usually, such statements involve so-called vague/fuzzy predicates [294].

Note that all vague/fuzzy statements are truth-functional, that is, the degree of truth of every statement can be calculated from the degrees of truth of its constituents, while uncertain statements cannot be a function of the uncertainties of their constituents [63]. More concretely, in probability theory, only the negation is truth-functional (see Eq. 1), while in possibility theory, only the disjunction resp. conjunction is truth-functional in possibilities resp. necessities of events (see Eq. 2). Furthermore, mathematical fuzzy logics are based on truly many-valued logical operators, while uncertainty logics are defined on top of standard binary logical operators.

In the following, we illustrate a typical formalization of uncertain statements and vague statements. In the former case, we consider a basic probabilistic/possibilistic logic, while in the latter, we consider a basic many-valued logic.

2.1 Probabilistic Logic

Probabilistic logic has its origin in philosophy and logic. Its roots can be traced back to Boole in 1854 [18]. There is a wide spectrum of formal languages that have been explored in probabilistic logic, ranging from constraints for unconditional and conditional events to rich languages that specify linear inequalities over events (see especially the work by Nilsson [210], Fagin et al. [75], Dubois and Prade et al. [61,66,5,65], Frisch and Haddawy [82], and the first author [157,160,164]; see also the survey on sentential probability logic by Hailperin [95]). Recently, nonmonotonic generalizations of probabilistic logic have been developed and explored; see especially [168] for an overview.
In this section, for illustrative purposes, we recall only the simple probabilistic logic described in [210].

We first define probabilistic formulas and probabilistic knowledge bases. We assume a set of basic events $\Phi = \{p_1, \ldots, p_n\}$ with $n \geq 1$. We use $\bot$ and $\top$ to denote $false$ and $true$, respectively. We define events by induction as follows. Every element of $\Phi \cup \{\bot, \top\}$ is an event. If $\phi$ and $\psi$ are events, then also $\neg \phi$, $(\phi \land \psi)$, $(\phi \lor \psi)$, and $(\phi \rightarrow \psi)$ are events. We adopt the usual conventions to eliminate parentheses. A probabilistic formula is an expression of the form $\phi \geq l$, where $\phi$ is an event, and $l$ is a real number from the unit interval $[0, 1]$. Informally, $\phi \geq l$ says that $\phi$ is true with a probability of at least $l$. For example, $\text{rain}_{\text{tomorrow}} \geq 0.7$ may express that it will rain tomorrow with a probability of at least 0.7. Notice also that $\neg \phi \geq 1 - u$ encodes that $\phi$ is true with a probability of at most $u$. A probabilistic knowledge base $KB$ is a finite set of probabilistic formulas.

We next define worlds and probabilistic interpretations. A world $I$ associates with every basic event in $\Phi$ a binary truth value. We extend $I$ by induction to all events as usual. We denote by $I_\Phi$ the (finite) set of all worlds for $\Phi$. A world $I$ satisfies an event $\phi$, or $I$ is a model of $\phi$, denoted $I \models \phi$, iff $I(\phi) = \text{true}$. A probabilistic interpretation $Pr$ is a probability function on $I_\Phi$ (that is, a mapping $Pr : I_\Phi \rightarrow [0, 1]$) such that all $Pr(I)$ with $I \in I_\Phi$ sum up to 1. Intuitively, $Pr(I)$ is the degree to which the world $I \in I_\Phi$ is probable, that is, the probability function $Pr$ encodes our “uncertainty” about which world is the right one. The probability of an event $\phi$ in $Pr$, denoted $Pr(\phi)$, is the sum of all $Pr(I)$ such that $I \in I_\Phi$ and $I \models \phi$. The following equations are an immediate consequence of the above definitions: for all probabilistic interpretations $Pr$ and events $\phi$ and $\psi$, the following relationships hold:

\[
\begin{align*}
Pr(\phi \land \psi) &= Pr(\phi) + Pr(\psi) - Pr(\phi \lor \psi); \\
Pr(\phi \land \psi) &\leq \min(Pr(\phi), Pr(\psi)); \\
Pr(\phi \land \psi) &\geq \max(0, Pr(\phi) + Pr(\psi) - 1); \\
Pr(\phi \lor \psi) &= Pr(\phi) + Pr(\psi) - Pr(\phi \land \psi); \\
Pr(\phi \lor \psi) &\leq \min(1, Pr(\phi) + Pr(\psi)); \\
Pr(\neg \phi) &= 1 - Pr(\phi); \\
Pr(\bot) &= 0; \\
Pr(\top) &= 1. 
\end{align*}
\]

A probabilistic interpretation $Pr$ satisfies a probabilistic formula $\phi \geq l$, or $Pr$ is a model of $\phi \geq l$, denoted $Pr \models \phi \geq l$, iff $Pr(\phi) \geq l$. We say $Pr$ satisfies a probabilistic knowledge base $KB$, or $Pr$ is a model of $KB$, iff $Pr$ satisfies all $F \in KB$. We say $KB$ is satisfiable iff a model of $KB$ exists. A probabilistic formula $F$ is a logical consequence of $KB$, denoted $KB \models F$, iff every model of $KB$ satisfies $F$. We say $\phi \geq l$ is a tight logical consequence of $KB$ iff $l$ is the infimum of $Pr(\phi)$ subject to all models $Pr$ of $KB$. Notice that the latter is equivalent to $l = \sup \{r \mid KB \models \phi \geq r\}$.

The main decision and optimization problems in probabilistic logic are deciding the satisfiability of probabilistic knowledge bases and logical consequences from probabilistic knowledge bases, as well as computing tight logical consequences from probabilistic knowledge bases, which can be done by deciding the solvability of a system
of linear inequalities and by solving a linear optimization problem, respectively. In particular, column generation techniques from operations research have been successfully used to solve large problem instances in probabilistic logic; see especially the work by Jaumard et al. [114] and Hansen et al. [99].

2.2 Possibilistic Logic

We next recall possibilistic logic; see especially [60]. The main syntactic and semantic differences to probabilistic logic can be summarized as follows. Syntactically, rather than using probabilistic formulas to constrain the probabilities of propositional events, we now use possibilistic formulas to constrain the necessities and possibilities of propositional events. Semantically, rather than having probability distributions on worlds, each of which associates with every event a unique probability, we now have possibility distributions on worlds, each of which associates with every event a unique possibility and a unique necessity. Differently from the probability of an event, which is the sum of the probabilities of all worlds that satisfy that event, the possibility of an event is the maximum of the possibilities of all worlds that satisfy the event. As a consequence, probabilities and possibilities of events behave quite differently from each other (see Eqs. 1 and 2). These fundamental semantic differences between probabilities and possibilities can also be used as the main criteria for using either probabilistic logic or possibilistic logic in a given application involving uncertainty. In addition, possibilistic logic may especially be used for encoding user preferences, since possibility measures can actually be viewed as rankings (on worlds or also objects) along an ordinal scale.

The semantic differences between probabilities and possibilities are also reflected in the computational properties of possibilistic and probabilistic logic, since reasoning in probabilistic logic generally requires to solve linear optimization problems, while reasoning in possibilistic logic does not, and thus can generally be done with less computational effort. Note that although possibility measures can be viewed as sets of upper probability measures [62], and possibility and probability measures can be translated into each other [57], no translations are known between possibilistic and probabilistic knowledge bases as described here.

We first define possibilistic formulas and knowledge bases. Possibilistic formulas have the form \( P\phi \geq l \) or \( N\phi \geq l \), where \( \phi \) is an event, and \( l \) is a real number from \([0,1]\). Informally, such formulas encode to what extent \( \phi \) is possibly resp. necessarily true. For example, \( P\text{rain tomorrow} \geq 0.7 \) encodes that it will rain tomorrow is possible to degree 0.7, while \( N\text{father} \rightarrow \text{man} \geq 1 \) says that a father is necessarily a man. A possibilistic knowledge base \( KB \) is a finite set of possibilistic formulas.

A possibilistic interpretation is a mapping \( \pi : \mathcal{I}_\Phi \rightarrow [0,1] \). Intuitively, \( \pi(I) \) is the degree to which the world \( I \) is possible. In particular, every world \( I \) such that \( \pi(I) = 0 \) is impossible, while every world \( I \) such that \( \pi(I) = 1 \) is totally possible. We say \( \pi \) is normalized iff \( \pi(I) = 1 \) for some \( I \in \mathcal{I}_\Phi \). Intuitively, this guarantees that there exists at least one world, which could be considered as the real one.

The possibility of an event \( \phi \) in a possibilistic interpretation \( \pi \), denoted \( \text{Poss}(\phi) \), is then defined by \( \text{Poss}(\phi) = \max \{ \pi(I) | I \in \mathcal{I}_\Phi, I \models \phi \} \) (where \( \max \emptyset = 0 \)). Intuitively, the possibility of \( \phi \) is evaluated in the most possible world where \( \phi \) is true. The dual notion to the possibility of an event \( \phi \) is the necessity of \( \phi \), denoted \( \text{Nec}(\phi) \), which is
defined by \( \text{Nec}(\phi) = 1 - \text{Poss}(\neg \phi) \). It reflects the lack of possibility of \( \neg \phi \), that is, \( \text{Nec}(\phi) \) evaluates to what extent \( \phi \) is certainly true. The following properties follows immediately from the above definitions.

for all possibilistic interpretations \( \pi \) and events \( \phi \) and \( \psi \), the following relationships hold:

\[
\begin{align*}
\text{Poss}(\phi \land \psi) & \leq \min(\text{Poss}(\phi), \text{Poss}(\psi)); \\
\text{Poss}(\phi \lor \psi) & = \max(\text{Poss}(\phi), \text{Poss}(\psi)); \\
\text{Poss}(\neg \phi) & = 1 - \text{Nec}(\phi); \\
\text{Poss}(\bot) & = 0; \\
\text{Poss}(\top) & = 1 \quad \text{in the normalized case}; \\
\text{Nec}(\phi \land \psi) & = \min(\text{Nec}(\phi), \text{Nec}(\psi)); \\
\text{Nec}(\phi \lor \psi) & \geq \max(\text{Nec}(\phi), \text{Nec}(\psi)); \\
\text{Nec}(\neg \phi) & = 1 - \text{Poss}(\phi); \\
\text{Nec}(\bot) & = 0 \quad \text{in the normalized case}; \\
\text{Nec}(\top) & = 1.
\end{align*}
\] (2)

A possibilistic interpretation \( \pi \) satisfies a possibilistic formula \( P \phi \geq l \) (resp., \( N \phi \geq l \)), or \( \pi \) is a model of \( P \phi \geq l \) (resp., \( P \phi \geq l \)), denoted \( \pi \models P \phi \geq l \) (resp., \( \pi \models P \phi \geq l \)), iff \( \text{Poss}(\phi) \geq l \) (resp., \( \text{Nec}(\phi) \geq l \)). The notions of satisfiability, logical consequence, and tight logical consequence for possibilistic knowledge bases are then defined as usual (in the same way as in the probabilistic case). We refer the reader to [60,107] for algorithms for possibilistic logic.

### 2.3 Many-Valued Logics

In the setting of many-valued logics, the convention prescribing that a proposition is either true or false is changed. A more refined range is used for the function that represents the meaning of a proposition. This is usual in natural language when words are modeled by fuzzy sets. For example, the compatibility of “tall” in the phrase “a tall man” with some individual of a given height is often graded: The man can be judged not quite tall, somewhat tall, rather tall, very tall, etc. Changing the usual true/false convention leads to a new concept of proposition, whose compatibility with a given state of facts is a matter of degree and can be measured on an ordered scale \( S \) that is no longer \( \{0, 1\} \), but e.g. the unit interval \( [0, 1] \). This leads to identifying a “fuzzy proposition” \( \phi \) with a fuzzy set of possible states of affairs; the degree of membership of a state of affairs to this fuzzy set evaluates the degree of fit between the proposition and the state of facts it refers to. This degree of fit is called degree of truth of the proposition \( \phi \) in the interpretation \( I \) (state of affairs). Many-valued logics provide compositional calculi of degrees of truth, including degrees between “true” and “false”. A sentence is now not true or false only, but may have a truth degree taken from a truth space \( S \), usually \( [0, 1] \) (in that case we speak bout Mathematical Fuzzy Logic [96]) or \( \{0, \frac{1}{n}, \frac{2}{n}, \ldots, \frac{n-1}{n}, 1\} \) for an integer \( n \geq 1 \). Often \( S \) may be also a complete lattice or a bilattice [86,80] (often used in logic programming [81]). In the sequel, we assume \( S = [0, 1] \).

In the many-valued logic that we consider here, many-valued formulas have the form \( \phi \geq l \) or \( \phi \leq u \), where \( l, u \in [0, 1] \) [94,96], which encode that the degree of truth
of \( \phi \) is at least \( l \) resp. at most \( u \). For example, \( \text{ripe} \_\text{tomato} \geq 0.9 \) says that we have a rather ripe tomato (the degree of truth of \( \text{ripe} \_\text{tomato} \) is at least 0.9).

Semantically, a many-valued interpretation \( \mathcal{I} \) maps each basic proposition \( p_i \) into \([0, 1]\) and is then extended inductively to all propositions as follows:

\[
\begin{align*}
\mathcal{I}(\phi \land \psi) &= \mathcal{I}(\phi) \otimes \mathcal{I}(\psi); \\
\mathcal{I}(\phi \lor \psi) &= \mathcal{I}(\phi) \oplus \mathcal{I}(\psi); \\
\mathcal{I}(\phi \rightarrow \psi) &= \mathcal{I}(\phi) \rightarrow \mathcal{I}(\psi); \\
\mathcal{I}(\neg \phi) &= \ominus \mathcal{I}(\phi),
\end{align*}
\]

where \( \otimes, \oplus, \Rightarrow, \) and \( \ominus \) are so-called combination functions, namely, triangular norms (or t-norms), triangular co-norms (or s-norms), implication functions, and negation functions, respectively, which extend the classical Boolean conjunction, disjunction, implication, and negation, respectively, to the many-valued case.

Several t-norms, s-norms, implication functions, and negation functions have been given in the literature. An important aspect of such functions is that they satisfy some properties that one expects to hold for the connectives; see Tables 1 and 2. Note that in Table 1, the two properties Tautology and Contradiction follow from Identity, Commutativity, and Monotonicity. Usually, the implication function \( \Rightarrow \) is defined as \( r \)-implication, that is, \( a \Rightarrow b = \sup \{c \mid a \otimes c \leq b\} \).

Some t-norms, s-norms, implication functions, and negation functions of various fuzzy logics are shown in Table 3 [96]. In fuzzy logic, one usually distinguishes three different logics, namely, Łukasiewicz, Gödel, and Product logic; the popular Zadeh logic is a sublogic of Łukasiewicz logic. Some salient properties of these logics are shown in Table 4. For more properties, see especially [96,212]. Note also, that a many-valued logic having all properties shown in Table 4, collapses to boolean logic, that is the truth-set can be \([0, 1] \) only.

The implication \( x \Rightarrow y = \max(1 - x, y) \) is called Kleene-Dienes implication in the fuzzy logic literature. Note that we have the following inferences: Let \( a \geq n \) and \( a \Rightarrow b \geq m \). Then, under Kleene-Dienes implication, we infer that if \( n > 1 - m \) then \( b \geq m \). Under r-implication relative to a t-norm \( \otimes \), we infer that \( b \geq n \otimes m \).

Note that implication functions and t-norms are also used to define the degree of subsumption between fuzzy sets and the composition of two (binary) fuzzy relations. A fuzzy set \( R \) over a countable crisp set \( X \) is a function \( R: X \rightarrow [0, 1] \). The degree of subsumption between two fuzzy sets \( A \) and \( B \), denoted \( A \sqsubseteq B \), is defined as \( \inf_{x \in X} A(x) \Rightarrow B(x) \), where \( \Rightarrow \) is an implication function. Note that if \( A(x) \leq B(x) \), for all \( x \in [0, 1] \), then \( A \sqsubseteq B \). Evaluates to 1. Of course, \( A \sqsubseteq B \) is defined as \( \sup_{x \in X} R(x, y) \), for every \( x \in X \) and \( y \in Y \). The composition of two fuzzy relations \( R_1: X \times Y \rightarrow [0, 1] \) and \( R_2: Y \times Z \rightarrow [0, 1] \) is defined as \( R_1 \circ R_2 \). A fuzzy relation \( R \) is transitive iff \( R(x, z) \geq (R \circ R)(x, z) \).

A many-valued interpretation \( \mathcal{I} \) satisfies a many-valued formula \( \phi \geq l \) (resp., \( \phi \leq u \)) or \( \mathcal{I} \) is a model of \( \phi \geq l \) (resp., \( \phi \leq u \)), denoted \( \mathcal{I} \models \phi \geq l \) (resp., \( \mathcal{I} \models \phi \leq u \), iff \( \mathcal{I}(\phi) \geq l \) (resp., \( \mathcal{I}(\phi) \leq u \)). The notions of satisfiability, logical consequence, and tight logical
### Table 1. Properties for t-norms and s-norms.

<table>
<thead>
<tr>
<th>Axiom Name</th>
<th>T-norm</th>
<th>S-norm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tautology / Contradiction</td>
<td>(a \otimes 0 = 0)</td>
<td>(a \oplus 1 = 1)</td>
</tr>
<tr>
<td>Identity</td>
<td>(a \otimes 1 = a)</td>
<td>(a \oplus 0 = a)</td>
</tr>
<tr>
<td>Commutativity</td>
<td>(a \otimes b = b \otimes a)</td>
<td>(a \oplus b = b \oplus a)</td>
</tr>
<tr>
<td>Associativity</td>
<td>((a \otimes b) \otimes c = a \otimes (b \otimes c))</td>
<td>((a \oplus b) \oplus c = a \oplus (b \oplus c))</td>
</tr>
<tr>
<td>Monotonicity</td>
<td>if (b \leq c), then (a \otimes b \leq a \otimes c)</td>
<td>if (b \leq c), then (a \oplus b \leq a \oplus c)</td>
</tr>
</tbody>
</table>

### Table 2. Properties for implication and negation functions.

<table>
<thead>
<tr>
<th>Axiom Name</th>
<th>Implication Function</th>
<th>Negation Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tautology / Contradiction</td>
<td>(0 \Rightarrow b = 1, a \Rightarrow 1 = 1, 1 \Rightarrow 0 = 0)</td>
<td>(\ominus 0 = 1, \ominus 1 = 0)</td>
</tr>
<tr>
<td>Antitonicity</td>
<td>if (a \leq b), then (a \Rightarrow c \geq b \Rightarrow c)</td>
<td>if (a \leq b), then (\ominus a \geq \ominus b)</td>
</tr>
<tr>
<td>Monotonicity</td>
<td>if (b \leq c), then (a \Rightarrow b \leq a \Rightarrow c)</td>
<td></td>
</tr>
</tbody>
</table>

### Table 3. Combination functions of various fuzzy logics.

<table>
<thead>
<tr>
<th>Łukasiewicz Logic</th>
<th>Gödel Logic</th>
<th>Product Logic</th>
<th>Zadeh Logic</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a \otimes b)</td>
<td>(\max(a + b - 1, 0))</td>
<td>(\min(a, b))</td>
<td>(a \cdot b)</td>
</tr>
<tr>
<td>(a \oplus b)</td>
<td>(\min(a + b, 1))</td>
<td>(\max(a, b))</td>
<td>(a + b - a \cdot b)</td>
</tr>
<tr>
<td>(a \Rightarrow b)</td>
<td>(\min(1 - a + b, 1))</td>
<td>(1) if (a \leq b) (\ominus) otherwise</td>
<td>(\min(1, b/a)) (\max(1 - a, b))</td>
</tr>
<tr>
<td>(\ominus a)</td>
<td>(1 - a)</td>
<td>(1) if (a = 0) (\ominus) otherwise</td>
<td>(1) if (a = 0) (\ominus) otherwise (1 - a)</td>
</tr>
</tbody>
</table>

### Table 4. Some additional properties of combination functions of various fuzzy logics.

<table>
<thead>
<tr>
<th>Property</th>
<th>Łukasiewicz Logic</th>
<th>Gödel Logic</th>
<th>Product Logic</th>
<th>Zadeh Logic</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x \otimes x = 0)</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>(x \oplus x = 1)</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>(x \otimes x = x)</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>(x \oplus x = x)</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>(\ominus x = x)</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>(x \Rightarrow y = \ominus x \oplus y)</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>(\ominus (x \Rightarrow y) = x \otimes y)</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>(\ominus (x \otimes y) = x \oplus y)</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>(\ominus (x \oplus y) = \ominus x \otimes y)</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>
consequence for many-valued knowledge bases are then defined in the standard way (in the same way as in the probabilistic case). We refer the reader to [93,94,96] for algorithms for many-valued logics.

3 Managing Imperfect Knowledge in Semantic Web Languages

3.1 The case of Description Logics

Probabilistic Uncertainty and Description Logics. Although there are several previous approaches to probabilistic description logics without semantic web background, P-SHOIN(D) [87,170,174] (see also [178]) is the most expressive probabilistic description logic, both in terms of the generalized classical description logic and in terms of the supported forms of terminological and assertional probabilistic knowledge. The syntax of the probabilistic description logic P-SHOIN(D) uses the notion of a conditional constraint from [160] to express probabilistic knowledge in addition to the axioms of SHOIN(D). Its semantics is based on the notion of lexicographic entailment in probabilistic default reasoning [162,166], which is a probabilistic generalization of the sophisticated notion of lexicographic entailment by Lehmann [134] in default reasoning from conditional knowledge bases. Due to this semantics, P-SHOIN(D) allows for expressing both terminological probabilistic knowledge about concepts and roles, and also assertional probabilistic knowledge about instances of concepts and roles. It naturally interprets terminological and assertational probabilistic knowledge as statistical knowledge about concepts and roles and as degrees of belief about instances of concepts and roles, respectively, and allows for deriving both statistical knowledge and degrees of belief. As an important additional feature, it also allows for expressing default knowledge about concepts (as a special case of terminological probabilistic knowledge), which is semantically interpreted as in Lehmann’s lexicographic default entailment [134].

Roughly, in it we have conditional constraints of the form \( (\psi | \phi)[l, u] \), where \( \psi, \phi \) are concepts, \( l \) and \( u \) are reals from \([0, 1]\). Informally, \((\psi | \phi)[l, u]\) encodes that the probability of \( \psi \) given \( \phi \) lies between \( l \) and \( u \). A PTBoxes, PBoxes, and probabilistic knowledge bases as follows: (i) A PTBox \( PT = (T, P) \) consists of a classical (description logic) knowledge base \( T \) and a finite set of conditional constraints \( P \); (ii) A PBox \( P \) is a finite set of conditional constraints; and (iii) a probabilistic knowledge base \( KB = (T, P, (P_o)_{o \in I_P}) \) relative to \( I_P \) consists of a PTBox \( PT = (T, P) \) and one PBox \( P_o \) for every probabilistic individual \( o \in I_P \). The meaning of a conditional constraint \((\psi | \phi)[l, u]\) depends on whether it belongs to \( P \) or to \( P_o \) for some probabilistic individual \( o \in I_P \):

- Each \((\psi | \phi)[l, u]\) in \( P \) informally encodes that “generally, if an object belongs to \( \phi \), then it belongs to \( \psi \) with a probability in \([l, u]\)”. For example, \((\exists R. \{ o \} | \phi)[l, u]\) in \( P \), where \( o \in I_C \) and \( R \in R_A \), encodes that “generally, if an object belongs to \( \phi \), then it is related to \( o \) by \( R \) with a probability in \([l, u]\)”.
- Each \((\psi | \phi)[l, u]\) in \( P_o \), where \( o \in I_P \), informally encodes that “if \( o \) belongs to \( \phi \), then \( o \) belongs to \( \psi \) with a probability in \([l, u]\)”. For example, \((\exists R. \{ o' \} | \phi)[l, u]\) in \( P_o \), where \( o \in I_P \), \( o' \in I_C \), and \( R \in R_A \), expresses that “if \( o \) belongs to \( \phi \), then \( o \) is related to \( o' \) by \( R \) with a probability in \([l, u]\)”.


The main reasoning problems in P-$SHOIN(D)$ are summarized by the following decision and computation problems are (i) to decide whether a probabilistic knowledge base $KB = (T, P, (P_o)_{o \in I_P})$ is consistent; and (ii) compute the tightest bounds $l, u \in [0, 1]$ such that $KB \models (\psi|\phi)[l, u]$.

Note that if the chosen classical description logic allows for decidable knowledge base satisfiability, then also the main reasoning tasks in the probabilistic extension are all decidable. (see [170,174] for further details).

There are already implementations of its predecessor P-$SHOQ(D)$ (see [202]) and of a probabilistic description logic based on probabilistic default reasoning as in [162,166]. Recently, the Pronto system ¹, claims to have implemented P-$SHOIN(D)$.

Other approaches. Other approaches to probabilistic description logics can be classified according to the generalized classical description logics, the supported forms of probabilistic knowledge, the underlying probabilistic semantics, and the reasoning techniques.

One of the earliest approaches to probabilistic description logics is due to Heinsohn [100], who presents a probabilistic extension of the description logic $ALC$, which allows to represent terminological probabilistic knowledge about concepts and roles, and which is based on the notion of logical entailment in probabilistic logics, similar to [210,5,82,160]. Heinsohn [100], however, does not allow for assertional (classical or probabilistic) knowledge about concept and role instances. The main reasoning problems are deciding the consistency of probabilistic terminological knowledge bases and computing logically entailed tight probability intervals. Heinsohn proposes a sound and complete global reasoning technique based on classical reasoning in $ALC$ and linear programming, as well as a sound but incomplete local reasoning technique based on the iterative application of local inference rules.

Another early approach to probabilistic description logics is due to Jaeger [112], who also proposes a probabilistic extension of the description logic $ALC$, which allows for terminological probabilistic knowledge about concepts and roles, and assertional probabilistic knowledge about concept instances, but does not support assertional probabilistic knowledge about role instances (but he mentions a possible extension in this direction). The entailment of terminological probabilistic knowledge from terminological probabilistic knowledge is based on the notion of logical entailment in probabilistic logic, while the entailment of assertional probabilistic knowledge from terminological and assertional probabilistic knowledge is based on a cross-entropy minimization relative to terminological probabilistic knowledge. The main reasoning problems are terminological probabilistic consistency and inference, which are solved by linear programming, and assertional probabilistic consistency and inference, which are solved by an approximation algorithm.

The recent work by Dürig and Studer [67] presents a further probabilistic extension of $ALC$, which is based on a model-theoretic semantics as in probabilistic logics, but which only allows for assertional probabilistic knowledge about concept and role instances, and not for terminological probabilistic knowledge. The paper also explores independence assumptions for assertional probabilistic knowledge. The main reasoning

problem is deciding the consistency of assertional probabilistic knowledge, but neither an algorithm nor a decidability result is given.

Jaeger’s recent work [113] focuses on interpreting probabilistic concept subsumption and probabilistic role quantification through statistical sampling distributions, and develops a probabilistic version of the guarded fragment of first-order logic. The semantics is different from the semantics of all the other probabilistic description logics in this paper, since it is based on probability distributions over the domain, and not on the more commonly used probability distributions over a set of possible worlds. The paper proposes a sound Gentzen-style sequent calculus for the logic, but it neither proves the completeness of this calculus nor decidability in general.

Koller et al.’s work [126] presents the probabilistic description logic P-CLASSIC, which is a probabilistic generalization (of a variant) of the description logic CLASSIC. Similar to Heinsohn’s work [100], it allows for encoding terminological probabilistic knowledge about concepts, roles, and attributes (via so-called p-classes), but it does not support assertional (classical or probabilistic) knowledge about instances of concepts and roles. However, in contrast to [100], its probabilistic semantics is based on a reduction to Bayesian networks. The main reasoning problem is to determine the exact probabilities for conditionals between concept expressions in canonical form. This problem is solved by a reduction to inference in Bayesian networks. As an important feature of P-CLASSIC, the above problem can be solved in polynomial time, when the underlying Bayesian network is a polytree. Note that a recent implementation of P-CLASSIC is described in [116].

Closely related work by Yelland [292] proposes a probabilistic extension of a description logic close to \(FL\), whose probabilistic semantics is also based on a reduction to Bayesian networks, and it applies this approach to market analysis. The approach allows for encoding terminological probabilistic knowledge about concepts and roles, but it does not support assertional (classical or probabilistic) knowledge about instances of concepts and roles. Like in Koller et al.’s work [126], the main reasoning problem is to determine the exact probabilities for conditionals between concepts, which is solved by a reduction to inference in Bayesian networks.

**Probabilistic Web Ontology Languages.** The literature contains several probabilistic generalizations of web ontology languages. Many of these approaches focus especially on combining the web ontology language OWL with probabilistic formalisms based on Bayesian networks.

In particular, da Costa [28], da Costa and Laskey [29], and da Costa et al. [30] suggest a probabilistic generalization of OWL, called PR-OWL, whose probabilistic semantics is based on multi-entity Bayesian networks (MEBNs). The latter are a Bayesian logic that combines first-order logic with Bayesian networks. Roughly speaking, PR-OWL represents knowledge as parameterized fragments of Bayesian networks. Hence, it can encode probability distributions on the interpretations of an associated first-order theory as well as repeated structure.

In [55,56], Ding et al. propose a probabilistic generalization of OWL, called BayesOWL, which is based on standard Bayesian networks. BayesOWL provides a set of rules and procedures for the direct translation of an OWL ontology into a Bayesian net-
work, and it also provides a method for incorporating available probability constraints when constructing the Bayesian network. The generated Bayesian network, which preserves the semantics of the original ontology and which is consistent with all the given probability constraints, supports ontology reasoning, both within and across ontologies, as Bayesian inferences. In [215,56], Ding et al. also describe an application of the BayesOWL approach in ontology mapping.

In closely related work, Mitra et al. [196] describe an implemented technique, called OMEN, to enhance existing ontology mappings by using a Bayesian network to represent the influences between potential concept mappings across ontologies. More concretely, OMEN is based on a simple ontology model similar to RDF Schema. It uses a set of meta-rules that capture the influence of the ontology structure and the semantics of ontology relations, and matches nodes that are neighbors of already matched nodes in the two ontologies.

Yang and Calmet [289] present an integration of the web ontology language OWL with Bayesian networks, called OntoBayes. The approach makes use of probability and dependency-annotated OWL to represent uncertain information in Bayesian networks. The work also describes an application in risk analysis for insurance and natural disaster management. Pool and Aikin [217] also provide a method for representing uncertainty in OWL ontologies, while Fukushige [84] proposes a basic framework for representing probabilistic relationships in RDF. Nottelmann and Fuhr [211] present two probabilistic extensions of variants of OWL Lite, along with a mapping to locally stratified probabilistic Datalog.

Another important work is due to Udrea et al. [278], who present a probabilistic generalization of RDF, which allows for representing terminological probabilistic knowledge about classes and assertional probabilistic knowledge about properties of individuals. They provide a technique for assertional probabilistic inference in acyclic probabilistic RDF theories, which is based on the notion of logical entailment in probabilistic logic, coupled with a local probabilistic semantics. They also provide a prototype implementation of their algorithms.

An important application for probabilistic ontologies (and thus probabilistic description logics and ontology languages) is especially information retrieval. In particular, Subrahmanian’s group [277,109] explores the use of probabilistic ontologies in relational databases. They propose to extend relations by associating with every attribute a constrained probabilistic ontology, which describes relationships between terms occurring in the domain of that attribute. An extension of the relational algebra then allows for an increased recall (which is the proportion of documents relevant to a search query in the collection of all returned documents) in information retrieval. In closely related work, Mantay et al. [184] propose a probabilistic least common subsumer operation, which is based on a probabilistic extension of the description logic $\mathcal{ALN}$. They show that applying this approach in information retrieval allows for reducing the amount of retrieved data and thus for avoiding information flood. Another closely related work by Holli and Hyvönen [102,103] shows how degrees of overlap between concepts can be modeled and computed efficiently using Bayesian networks based on RDF(S) ontologies. Such degrees of overlap indicate how well an individual data item matches the query concept, and can thus be used for measuring the relevance in information re-
Possibilistic Uncertainty and Description Logics. Similar to probabilistic extensions of description logics, possibilistic extensions of description logics have been developed by Hollunder [107]; Dubois et al. [59] and more recently in [221].

A possibilistic axiom is of the form $P \alpha \geq l$ or $N \alpha \geq l$, where $\alpha$ is a classical description logic axiom, and $l$ is a real number from $[0, 1]$. A possibilistic RBox (resp., TBox, ABox) is a finite set of possibilistic axioms $P \alpha \geq l$ or $N \alpha \geq l$, where $\alpha$ is an RBox (resp., TBox, ABox) axiom. A possibilistic knowledge base $KB = (R, T, A)$ consists of a possibilistic RBox $R$, a possibilistic TBox $T$, and a possibilistic ABox $A$. The semantics is a straightforward extension from the propositional case to the FOL case.

The main reasoning problems related to possibilistic description logics are deciding whether a possibilistic knowledge base is satisfiable, deciding whether a possibilistic axiom is a logical consequence of a possibilistic knowledge base, and computing the tight lower and upper bounds entailed by a possibilistic knowledge base for the necessity and the possibility of a classical description logic axiom. As shown by Hollunder [107], deciding logical consequences, and thus also deciding satisfiability and computing tight lower and upper bounds can be reduced to deciding logical consequences in classical description logics. A recent implementation of reasoning in possibilistic description logics using KAON2\(^2\) is reported in [223,222].

We recall that Liau and Yao [141] report on an application of possibilistic description logics in information retrieval. More concretely, they define a possibilistic generalization of the description logic $\mathcal{ALC}$ and show that it can be used in typical information retrieval problems, such as query relaxation, query restriction, and exemplar-based retrieval. Possibilistic description logics can also be used for handling inconsistencies in ontologies [223,222]. Another important application of possibilistic description logics is the representation of user preferences in the Semantic Web. For example, the recent work by Hadjali et al. [91] shows that possibilistic logic can be nicely used for encoding user preferences in the context of databases.

Vagueness and Description Logics. There are several extensions of description logics and ontology languages using the theory of fuzzy logic. They can be classified according to (a) the description logic resp. ontology language that they generalize, (b) the allowed fuzzy constructs, (c) the underlying fuzzy logics, and (d) their reasoning algorithms.

In general, fuzzy DLs allow expressions of the form $(a; C, n)$, stating that $a$ is an instance of concept $C$ with degree at least $n$, that is the FOL formula $C(a)$ is true to degree at least $n$ (it is straightforward to map DL expressions into FOL formulae). Similarly, $(C_1 \sqsubseteq C_2, n)$ and $(R_1 \sqsubseteq R_2, n)$ state vague subsumption relationships. Informally, $(C_1 \sqsubseteq C_2, n)$ dictates that the FOL formula $\forall x. C_1(x) \rightarrow C_2(x)$ is always

\(^2\)http://kaon2.semanticweb.org/
true to degree at least \( n \) (note that in mathematical fuzzy logic, the universal quantification \( \forall x \) is interpreted as \( \inf_x \), and similarly, \( \exists x \) is interpreted as \( \sup_x \) and, that not always \( \neg \forall \) is the same as \( \exists \neg \), –this is true only for Zadeh logic and Łukasiewicz logic).

In addition to the standard problems of deciding the satisfiability of fuzzy knowledge bases, deciding the satisfiability of concepts relative to fuzzy knowledge bases, and deciding logical consequences of fuzzy axioms from fuzzy knowledge bases, two other important reasoning problems are the best truth value bound problem and the best satisfiability bound problem, that is (i) to determine the tightest bound \( n \in [0, 1] \) of an axiom \( \alpha \), denoted \( \text{glb}(KB, \alpha) \), and defined as \( \text{glb}(KB, \alpha) = \sup \{ n \mid KB \models (\alpha, n) \} \); and (ii) to determine \( \text{glb}(KB, C) = \sup_I \sup_{x \in \Delta I} \{ C_I(x) \mid I \models KB \} \) (intuitively, among all models \( I \) of \( KB \), we determine the maximal degree of truth that the concept \( C \) may have over all individuals \( x \in \Delta I \)).

The first work about fuzzy DLs is due to Yen [293], who proposes a fuzzy extension of a very restricted sublanguage of \( \mathcal{ALC} \), called \( \mathcal{FL}^{-} \) [19,135]. The work includes fuzzy terminological knowledge, but no fuzzy assertional knowledge, and it is based on Zadeh logic. It already informally talks about the use of fuzzy modifiers and fuzzy concrete domains. Though, the unique reasoning facility, the subsumption test, is a crisp yes/no questioning. Tresp and Molitor [275] consider a more general extension of fuzzy \( \mathcal{ALC} \). Like Yen, they also allow for fuzzy terminological knowledge along with a special form of fuzzy modifiers (which are a combination of two linear functions), but no fuzzy assertional knowledge, and they assume Zadeh logic as underlying fuzzy logic. The work also presents a sound and complete reasoning algorithm testing the subsumption relationship using a linear programming oracle.

Another fuzzy extension of \( \mathcal{ALC} \) is due to Straccia [249,251,257,262,271], who allows for both fuzzy terminological and fuzzy assertional knowledge, but not for fuzzy modifiers and fuzzy concrete domains, and again assumes Zadeh logic as underlying fuzzy logic. Straccia [249,251] also introduces the best truth value bound problem and provides a sound and complete reasoning algorithm based on completion rules. In [250], Straccia reports a four-valued variant of fuzzy \( \mathcal{ALC} \). In the same spirit, Hölldobler et al. [104,105] extend Straccia’s fuzzy \( \mathcal{ALC} \) with concept modifiers of the form \( f_m(x) = x^\beta \), where \( \beta > 0 \), and present a sound and complete reasoning algorithm (based on completion rules) for the graded subsumption problem.

Straccia’s works [253,261,267] are essentially as [251], except that now the set of possible truth values is a complete lattice rather than \([0, 1]\).

Sanchez and Tettamanzi [232,233,234] consider a fuzzy extension of the description logic \( \mathcal{ALC}_Q \) (without assertional component) under Zadeh logic, and they start addressing the issue of a fuzzy semantics of quantifiers. Essentially, fuzzy quantifiers allow to state sentences such as \( \text{FaithfulCustomer} \cap \text{(Most)} \text{buys_LOWCalorie- Food} \) encoding “the set of all individuals that mostly by low calorie food”. An algorithm is presented, which calculates the satisfiability interval for a fuzzy concept.

Hájek [97,98] considers a fuzzy extension of the description logic \( \mathcal{ALC} \) under arbitrary t-norms. He provides in particular algorithms for deciding whether \( (C \subseteq D, 1) \) is a tautology and whether \( (C \subseteq D, 1) \) is satisfiable, which are based on a reduction to the propositional BL logic for which a Hilbert-style axiomatization exists [96] (but see
also [98] for the complexity of rational Pavelka logic, and see [17] for some complexity results on reasoning in fuzzy description logics.

Straccia [252] provides a translation of fuzzy $\mathcal{ALC}$ (with general concept inclusion axioms) into classical $\mathcal{ALC}$. The translation is modular, and thus expected to be extendable to more expressive fuzzy description logics as well. The main idea is to translate a fuzzy assertion of the form $(a: C,n) \geq$ into a crisp assertion $a: C$ with the intended meaning “$a$ is an instance of $C$ to degree at least $n$”. Then, concept inclusion axioms are used to correctly relate the $C_n$’s. For example, $C_{0.7} \sqsubseteq C_{0.6}$ is used to encode that whenever an individual is an instance of $C$ to degree at least 0.7, then it is also an instance of $C$ to degree at least 0.6. The translation is at most quadratic in the size of the fuzzy knowledge base. Note that the translation does not yet work in the presence of fuzzy modifiers and fuzzy concrete domains. Bobillo et al. [13] extend the approach to a variant of fuzzy $\mathcal{SHOIN}$. The idea has further been considered in the works [139,140], which essentially provide a crisp language in which expressions of, e.g., the form $a: \forall R_{0.8}.C_{0.9}$ are allowed, with the intended meaning “if $a$ has an $R$-successor to degree at least 0.8, then this successor is also an instance of $C$ to degree at least 0.9”. The idea has further been extended to a distributed variant of fuzzy description logics in [151]. A mapping to classical DLs under Łukasiewicz semantics has been provided in [16] for the fuzzy DL $\mathcal{ALCHOI}$.

In [180], a fuzzy extension (based on Zadeh logic) of CARIN [136] is provided, which combines fuzzy description logics with non-recursive Horn rules.

Other extensions of fuzzy description logics concern their integration with fuzzy logic programs, which however goes beyond the scope of the present paper (see, e.g., [267,263,261,169,280]). An interesting extension is due to Kang et al. [44], who extends fuzzy description logics by comparison operators, e.g., to state that “Tom is taller than Tim”. Another interesting extension is proposed by Dubois et al. [59], who combine fuzzy description logics with possibility theory. Essentially, since $(a: C,n) \geq$ is Boolean (either an interpretation satisfies it or not), we can build on top of it an uncertainty logic, which is based on possibility theory in [59].

We recall that usually the semantics used for fuzzy description logics is based on Zadeh logic, but where the concept inclusion is crisp, that is, $C \sqsubseteq D$ is viewed as $\forall x.C(x) \leq D(x)$. In [106,275], a calculus for fuzzy $\mathcal{ALC}$ [235] with fuzzy modifiers and simple TBoxes under Zadeh logic is reported. No indication for the BTVB problem is given. Straccia [249,251] reports a calculus for fuzzy $\mathcal{ALC}$ and simple TBoxes under Zadeh logic and addresses the BTVB problem. How the satisfiability problem and the BTVB problem can be reduced to classical $\mathcal{ALC}$, and thus can be solved by means of tools like FaCT and RACER is shown in [252]. Results providing a tableaux calculus for fuzzy $\mathcal{SHI}$ under Zadeh logic (but only allowing for a restricted form of concept inclusion axioms, which are called fuzzy inclusion introductions and fuzzy equivalence introductions), by adapting similar techniques as for the classical counterpart, are shown in [247,245]. Fuzzy general concept inclusion axioms under Zadeh logic can be managed as described in [248]. Also interesting is the work [290], which provides a tableau for fuzzy $\mathcal{SHI}$ with general concept inclusion axioms. Finally, the reasoning techniques for classical $\mathcal{SHOIN}(D)$ [108] can be extended to [251], as [247,245,244,246] already show.
On the other hand, fuzzy tableau algorithms under Zadeh semantics do not seem to be suitable to be adapted to other semantics, such as Łukasiewicz logic. Even more problematic is the fact that they are yet unable to deal with fuzzy concrete domains [254], that is the possibility to allow an explicit representation of fuzzy membership functions. Despite these negative results, recently, [255,254] report a calculus for fuzzy \( ALC(D) \) whenever the connectives, the modifiers, and the fuzzy datatype predicates are representable as bounded mixed integer linear programs (MILPs). For example, Łukasiewicz logic satisfies these conditions as well as the membership functions for fuzzy datatype predicates that we have presented in this paper. Additionally, modifiers should be a combination of linear functions. In that case, the calculus consists of a set of constraint propagation rules and an invocation to an oracle for MILP. The method has been extended to fuzzy \( SHIF(D) \) [268] (the description logic behind OWL Lite) and a reasoner, called fuzzyDL [15], has been implemented and is available at Straccia’s web page. FuzzyDL supports more features, which we do not address here. The use of MILP for reasoning in fuzzy description logics is not surprising as their use for automated deduction in many-valued logics is well-known [93,94]. Bobillo and Straccia [14] provide a calculus for fuzzy \( ALC(D) \) under product semantics.

A very recent problem for fuzzy description logics is the top-\( k \) retrieval problem. While in classical semantics, a tuple satisfies or does not satisfy a query, in fuzzy description logics, a tuple may satisfy a query to a degree. Hence, for example, given a conjunctive query over a fuzzy description logic knowledge base, it is of interest to compute only the top-\( k \) answers. While in relational databases, this problem is a current research area (see, e.g., [74,110,137]), very few is known for the case of first-order knowledge bases in general (but see [265]) and description logics in particular. The only works that we are aware of are [260,266,272], which deal with the problem of finding the top-\( k \) result over knowledge bases in a fuzzy generalization of DL-Lite [24] (note that [213,214] is subsumed by [266], though in [213,214] the storage systems is no-longer a database, but a RDF storage system).

Fuzzy logic has numerous practical applications in general (see, e.g., [125]). Related to fuzzy description logics, we point out that they have first been proposed for logic-based information retrieval [194], which originated from the idea to annotate textual documents with graded description logic sentences, which goes back to [195]. The idea has been reconsidered in [247,272,295]. In particular, (i) Zhang et al. [295] describe a semantic portal that is based on fuzzy description logics; (ii) Li et al. [138] present an improved semantic search model by integrating inference and information retrieval and an implementation in the security domain; (iii) Straccia and Visco [272] report on a multimedia information retrieval system based on a fuzzy DLR-Lite description logic, which is capable to deal with hundreds of thousands of images. D’Aquin et al. [43] provide a use case in the medical domain, where fuzzy concrete domains are used to identify tumor regions in x-ray images. Agarwal and Lamparter [1] use fuzzy description logics to improve searching and comparing products in electronic markets. They provide a more expressive search mechanism that is closer to human reasoning and that aggregates multiple search criteria to a single value (ranking of an offer relative to the query), thus enabling a better selection of offers to be considered for the negotiation. Liu et al. [142] use a fuzzy description logic to model the management part in project
selection tasks. Finally, [15] shows also how to use fuzzyDLs for e-Commerce Match-making and Semantic Fuzzy Control.

3.2 The case of Logic Programs

In logic programming, the management of imperfect information has attracted the attention of many researchers and numerous frameworks have been proposed. Addressing all of them is almost impossible, due to both the large number of works published in this field (early works date back to early 80-ties [241] ) and the different approaches proposed (see the appendix for a list of references). Like for the DL case, essentially they differ in the underlying notion of uncertainty theory and vagueness theory (probability theory, possibilistic logic, fuzzy logic and multi-valued logic) and how uncertainty/vagueness values, associated to rules and facts, are managed.

Basically [143], a logic program \( P \) is made out by a set of rules and a set of facts. Facts are atoms of the form \( P(t_1, \ldots, t_n) \), where \( t_i \) is a term (usually, a constant or a variable). In most cases, facts are ground. On the other hand rules are of the form \( A \leftarrow B_1, \ldots, B_n \), where each \( A \) and \( B_i \) is an atom. \( B_1, \ldots, B_n \) is called body, while \( A \) is called head of the rule. The intended meaning of a rules is that “if all \( B_i \) are true, then also \( A \) is true”. From a FOL perspective, a rule is just a FOL formula \( \forall x. B_1 \land \ldots \land B_n \rightarrow A \), where \( x \) are all the variables occurring in the rule. Such logic programs are called positive as no literal occurs. In case a literal occurs in the body, then we speak about normal logic programs. We may also have a disjunction of atoms in the head, and then we talk about disjunctive logic programs ([239]). In the most general setting, literals are allowed in the head as well and from a semantics point of view, the stable model semantics [85] is widely adopted.

Probabilistic Uncertainty and Logic Programs. The variety of proposals of logic programming under probability theory is huge and an description of most of them is out of the scope of this work. We describe here some groups of works.

In probabilistic generalizations of (annotated) logic programs (see [123]) based on probabilistic logic fall works such as [47,49,45,46,48,203,204,281,115], where rules have the form of annotated logic programming rules. Facts are expressions of the form \( A : \mu \), where \( \mu \) is an interval in \([0,1]\). The intended meaning of an expression \( A : [m,n] \) is “the probability of the event corresponding to \( A \) to occur (have occurred) lies in the interval \([m,n]\)”. Rules have the form \( A : \mu \leftarrow B_1 : \mu_1, \ldots, B_n : \mu_2 \), where \( \mu, \mu_i \) are intervals in \([0,1]\).

In probabilistic generalizations of logic programs based on Bayesian networks / causal models fall works such as [11,12,224,225,154,285,118,119,133,209,206,218,219,220]. Interesting is Poole’s Independent Choice Logic (ICL) approach. It is based on acyclic logic programs \( P \) under different “choices”. Each choice along with \( P \) produces a first-order model. By placing a probability distribution over the different choices, one then obtains a distribution over the set of first-order models. Roughly, rules and facts are as for classical logic programs. Additionally, there is a set \( C \) of choices of the form \( \{(A_1 : \alpha_1), \ldots, (A_n : \alpha_n)\} \), where \( A_i \) is an atom and the \( \alpha_i \) sum-up to 1. A total choice \( T_C \) is a set of atoms such that from each choice \( C_j \in C \) there is exactly one atom
A_i^j \in C_j in T_{C_j}. The probability of a query q w.r.t. to P is the sum of the probabilities p_C of total choices T_C such that P \cup T_C \models q, where p_C is the product of the \alpha_i^j, for C_j \in T_{C_j}. It is worth to note that the ICL approach generalizes Bayesian networks, influence diagrams, Markov decision processes, and normal form games.

In the third group fall first-order generalization of probabilistic knowledge bases in probabilistic logic (based on logical entailment, lexicographic entailment, and maximum entropy entailment) and comprises works such as [156,165,163]. In these works, similarly to P-SHOIN(D), expressions are of the form (\psi|\phi)[l,u], but now \psi, \phi are formulae rather than concepts. The development of the semantics parallels to the case of P-SHOIN(D).

**Possibilistic Uncertainty and Logic Programs.** In possibilistic logic programs [58], facts are of the form (P(t_1, \ldots, t_n), N l), while rules are of the form (A \leftarrow B_1, \ldots, B_n, N l). The meaning of them is given directly by the possibilistic FOL formulae, N P(t_1, \ldots, t_n), \ge l and N (\forall x. B_1 \land \ldots \land B_n \rightarrow A) \ge l, respectively (the necessity of the formula is greater or equal than l). This basic form has been extended in [208] (which describes also an implementation) to the case of disjunctive logic programming under the stable model semantics, while [3,4,2,26] allow explicitly to deal with fuzzy sets in the language.

**Vagueness and Logic Programs.** While there is a large literature related to the management of vagueness in logic programs, there are rule forms that are general enough to cover a large amount of them (see e.g., [282,256,177]). Roughly, rules are of the form A \leftarrow f(B_1, \ldots, B_n), where A, B_i are atoms and f is a total function f : S^n \rightarrow S over a truth space S. Computationally, given an assignment/interpretation I of values to the B_i, the value of A is computed by stating that A is at least as true as f(I(B_1), \ldots, I(B_n)). The form of the rules is sufficiently expressive to encompass many approaches to many-valued logic programming. [177] provides an even more general setting as the function f may also depend on the variables occurring in the rule body. On the other hand there are also some extensions to many-valued disjunctive logic programs [188,189,259]. In some cases, e.g., [131] there is also a function g, which dictates how to aggregate the truth values in case an atom is head of several rules.

Most works deal with logic programs without negation and some may provide some technique to answer queries in a top-down manner, as e.g.,[35,123,131,282,258]. On the other hand, there are very few works dealing with normal logic programs [38,79,81,144,145,146,147,148,149,150,188,256,259,264,269,176], and little is know about top-down query answering procedures. The only exceptions are [256,264,269].

Another rising problem is the problem to compute the top-k ranked answers to a query, without computing the score of all answers. This allows to answer queries such as “find the top-k closest hotels to the conference location”. Solutions to this problem can be found in [265,270,177].

**3.3 Description Logic Programs**

Description Logic Programs [88,197,230] are a combination of description logics with logic programming. There is a large body of work on integrating rules and ontologies,
which is a key requirement of the layered architecture of the Semantic Web. Significant research efforts focus on hybrid integrations of rules and ontologies, called description logic programs (or dl-programs), which are of the form $KB = (L, P)$, where $L$ is a description logic knowledge base and $P$ is a finite set of rules involving either queries to $L$ in a loose integration (see especially [72,73,69,70,71]) or concepts and roles from $L$ as unary resp. binary predicates in a tight integration (see especially [228,229,171,197,198]). Roughly, in the loosely coupled approach, DL atoms may appear in rule bodies and act as queries to an underlying DL system, while in the tightly coupled approach the integration is more involved.

In parallel to these to approaches (loosely coupled vs. tightly coupled) there has been some works on the extension of these approaches towards the management of imperfect information: (i) under probability fall works such as [167,172,173,21,22]; (ii) under vagueness fall the works [169,176,177,267,263]; while a combination of probability and vagueness in description logic programs can be found in the work (unique so far) [175].

### A Some references related to logic programming, uncertainty and vagueness

Below a list of references and the underlying imprecision and uncertainty theory in logic programming frameworks. The list of references is by no means intended to be all-inclusive. The author apologizes both to the authors and with the readers for all the relevant works, which are not cited here.

**Probability theory:** [11,12,7,8,40,42,47,49,45,46,48]  
[224,225,154,285,83,118,119,117,133,209,130,132,155,156,158]  

**Possibilistic logic:** [3,4,2,26,58,208]

**Fuzzy set theory:** [187,9,10,20,25,27,68,101,111,124,90,89,187,201,200,216]  
[226,240,241,243,179,273,279,283,282,284,286,291]

**Multi-valued logic:** [23,34,35,36,31,32,37,38,39,41,33,54,50,53,51,52]  
[79,81,76,77,78,92,120,121,122,123,127,128,129,131]  
[144,145,146,147,148,149,150,152,153,181,183,182,185,188,189,192,192,190,191,193]  
[231,236,237,238,258,256,259,264,265,270,269,276]

**References**


36. C. V. Damásio, J. Medina, and M. Ojeda Aciego. Termination results for sorted multi-
adjoint logic programs. In Proceedings of the 10th International Conference on Information 
Processing and Management of Uncertainty in Knowledge-Based Systems, (IPMU-04), 
In D. Gabbay and P. Smets, editors, Handbook of Defeasible Reasoning and Uncertainty 
European Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR-01), 
39. C. V. Damásio and L. M. Pereira. Monotonic and residuated logic programs. In S. Ben-
ferhat and P. Besnard, editors, Symbolic and Quantitative Approaches to Reasoning with 
Uncertainty, 6th European Conference, ECSQARU 2001, Toulouse, France, September 19- 
40. C. V. Damásio and L. M. Pereira. Hybrid probabilistic logic programs as residuated logic 
41. C. V. Damásio and L. M. Pereira. Sorted monotonic logic programs and their embeddings. 
In Proceedings of the 10th International Conference on Information Processing and Man-
Verlag.
43. M. d’Aquin, J. Lieber, and A. Napoli. Towards a semantic portal for oncology using a 
description logic with fuzzy concrete domains. In E. Sanchez, editor, Fuzzy Logic and the 
44. J. L. Dazhou Kang, Baowen Xu and Y. Li. Reasoning for a fuzzy description logic with 
comparison expressions. In Proceedings of the International Workshop on Description 
45. A. Dekhtyar and M. I. Dekhtyar. Possible worlds semantics for probabilistic logic pro-
grams. In 20th International Conference on Logic Programming, volume 3132 of Lecture 
46. A. Dekhtyar and M. I. Dekhtyar. Revisiting the semantics of interval probabilistic logic 
programs. In 8th International Conference on Logic Programming and Nonmonotonic Rea-
soning (LPNMR-05), number 3662 in Lecture Notes in Computer Science, pages 330–342. 
Springer Verlag, 2005.
47. A. Dekhtyar, M. I. Dekhtyar, and V. S. Subrahmanian. Temporal probabilistic logic pro-
grams. In D. D. Schreye, editor, Logic Programming: The 1999 International Conference, 
48. A. Dekhtyar and V. Subrahmanian. Hybrid probabilistic programs. Journal of Logic Pro-
49. M. I. Dekhtyar, A. Dekhtyar, and V. S. Subrahmanian. Hybrid probabilistic programs: 
Conference on Uncertainty in Artificial Intelligence (UAI-99), pages 160–169, S.F., Cal., 
50. M. Denecker, V. Marek, and M. Truszczyński. Approximations, stable operators, well-
founded fixpoints and applications in nonmonotonic reasoning. In J. Minker, editor, Logic-
and autoepistemic logics. In A. Cohn, F. Giunchiglia, and B. Selman, editors, Proceed-


181. Z. Majkic. Coalgebraic semantics for logic programs. In 18th Workshop on (Constraint) Logic Programming (W(C)LP-05), Ulm, Germany, 2004.

183. Z. Majkic. Truth and knowledge fixpoint semantics for many-valued logic programming. In 19th Workshop on (Constraint) Logic Programming ((W(C)LP-05), number 76-87, Ulm, Germany, 2005.


