Introduction to Probabilistic Ontologies

Rafael Peñaloza[0000−0002−2693−5790]

IKR3 Lab, University of Milano-Bicocca, Italy
rafael.penaloza@unimib.it

Abstract. There is no doubt about it; an accurate representation of a real knowledge domain must be able to capture uncertainty. As the best known formalism for handling uncertainty, probability theory is often called upon this task, giving rise to probabilistic ontologies. Unfortunately, things are not as simple as they might appear, and different choices made can deeply affect the semantics and computational properties of probabilistic ontology languages. In this tutorial, we explore the main design choices available, and the situations in which they may be meaningful or not. We then dive deeper into a specific family of probabilistic ontology languages which can express logical and probabilistic dependencies between axioms.

1 Introduction

Ontologies, in the computer-science sense of "representations of a knowledge domain" present a prominent research area of interest within the general umbrella of Artificial Intelligence, and more precisely in Knowledge Representation. Indeed, any intelligent agent should have, in one way or another, a representation of the knowledge of its active domain, so that it is able to react to observations of its environment in an adequate manner.

Within the context of the Semantic Web, ontologies have been identified as an appropriate manner to represent, combine, and in general use distributed knowledge from different sources into specific applications. Notably, beyond the general view of knowledge inter-connectivity, many industrial players, knowledge domains, and other users are building specialised ontologies for fitting their own needs. This has led not only to a large collection of ontologies dealing with all sorts of domain areas and large repositories holding them but, more importantly, with a plethora of languages which are used to build them. Abstracting from the specific peculiarities of these languages (and to avoid the need to clarify at each point which language is taken under consideration), we will see an ontology basically as a (typically finite) set of constraints that define how different elements of the knowledge domain should relate to each other. In particular, we focus on a setting where the ontology language has an underlying logical interpretation. The use of logic-based formalisms allows us to provide formal semantics, and guarantees of correctness for entailment methods. Thus, we can speak about consequences that follow from a given ontology; informally, these are pieces of knowledge that are implicitly (rather than explicitly) encoded in the ontology.

With the success of ontology languages and the proliferation of ontologies, the usual limitations of classical logic for modelling expert knowledge come more prominently to light. One particular gap is that of representing uncertainty. Unfortunately, in many domain areas it is simply not possible to represent adequate knowledge without some level of uncertainty. A typical domain where uncertainty is commonly found is in medicine. Indeed, there are almost no certainties in medical sciences, when it comes to diagnose and cure a disease. This is because different people present different symptoms and express them in diverse manners, but also due to the limits in our capacity to observe the disease itself. Let us justify this statement with a few examples.

Suppose that a patient complains about intense abdominal pain, and very high fever. If the patient is young, and without any other known medical issues, we can expect (with a high level of certainty) that they are suffering from appendicitis; however, without additional testing and controls, this diagnosis cannot be certain. Another example appears in viral testing. Some viral infections are tested via a respiratory swab which has a limited precision, with a small possibility of error. In simple terms, a positive result in the swab provides a likelihood, but no certainty, of the infection. Finally, there exist diseases like Chronic Traumatic Encephalopathy (CTE) which cannot be diagnosed before death [21]. Indeed, although there are some signs and symptoms which may indicate the development of this ailment, the only way to know that someone suffers from CTE is to observe their brain closely.

Although medical sciences provide good and understandable examples of uncertain knowledge, uncertainty is not intrinsic to medicine only. Manufacturing processes occassionally produce defective pieces; the voting intentions of a population can be predicted by other behaviours, up to a margin of error; weather forecasts and natural disaster predictions are never fully accurate. There are countless examples which could be enumerated here.

Given the overall presence of uncertainty in these areas, it makes sense to try to include a way to represent it and use it within ontology languages. The most prominent approach for handling uncertainty is, without any doubt, probability theory. Hence, without intending to diminish any other formalisms, in this work we focus only on probabilities, with the goal of building probabilistic ontologies. From a very high level point of view, the idea is very simple: just add a probabilistic interpretation to the constraints forming the ontology, and adapt any reasoning and derivation methods to handle these probabilities. But the devil, as always, is in the details. Once we start to look closely into the formalisms, we realise that there are many choices to be made. First and foremost, how do we interpret these probabilities? Another important choice is how to combine different statements meaningfully. This is specially relevant since probabilities, in contrast to most logical formalisms, are not truth- functional; that is, one cannot compute the probability of a complex event based only on the probabilities of its components.

As there are many choices to be made about the probabilistic interpretation, several probabilistic ontology languages have been studied in the literature, many of which will be mentioned throughout this tutorial. Indeed, each classical ontology language can potentially give rise to many different probabilistic ontology languages. The field is so rich that it would be impossible to study them all in detail in a work like this. Instead, our goal is to explain the different choices, their motivation, their advantages, and their disadvantages in a manner that a user interested in representing probabilistic knowledge is able to identify the right choices. To achieve this goal, we introduce what we call the Five Golden Rules: five high-level considerations that must be kept in mind when dealing with probabilistic ontologies. These rules are by no means complete; they are not even at the same level of abstraction. They are mereley intended as a simple way to remember some of the most relevant aspects which characterise the different formalisms.

The rest of this tutorial is structured as follows. We start by providing an abstract definition of ontology languages and ontologies, which will be the base of the rest of the work. In the same section, we enumerate several examples of ontology languages, to showcase the generality of our definitions, and the scope of our work. Section 3 helps as a primer to uncertainty, and more specifically probability theory. It introduces all the notions needed for the following sections, including conditional and joint distributions, random variables, and Bayesian networks. After this general introduction, we combine both formalisms to define probabilistic ontologies in Section 4. We also briefly discuss some formalisms that are left out from our definition. The Five Golden Rules are presented in Section 5. The "rules" are given mnemonic names to help clarify their scope. The last section before the conclusions focuses on a specific probabilistic ontology language which is built by a combination of hypergraphs and databases, and different independence assumptions. It is meant as an example of a language which can be constructed from our notions, and the choices made following the golden rules.

2 Ontologies

In a very abstract sense, an ontology is merely a description of the world, or a relevant segment of it. In more practical terms, an ontology introduces all the terms of interest within a domain, and their relationships; that is, it is a representation of domain knowledge. This representation helps to understand, share, improve, and in general use the domain knowledge across applications. Depending on the domain of interest, the application at hand, and the desired properties of the description, different kinds of languages may be more or less suitable. This has motivated the introduction and study of many different knowledge representation languages.

Nowadays, the terms *ontology* or *ontology language* in the context of knowledge representation is most likely to remind us about the Web Ontology Language (OWL) from the Semantic Web [24] or, for the younger ones among us, of Knowledge Graphs (KGs). Already these two terms evoque a large variety of languages, from the different formalisms used in modern KGs [4, 14], to the existing OWL 2 profiles [54], to the many description logic [2] formalisms that are less expressive than $\mathcal{SROIQ}$ [40], the logic underlying OWL 2. Despite this variety, one can easily think of features which are not covered by any one of these languages. For example, they cannot express temporal constraints, or limit the number of objects that satisfy a given property.

For most of this tutorial, we will be interested in the problem of extending ontology languages with probabilities. Since the general underlying ideas do not depend on the specific language, we will consider an abstract notion of *ontology language* which covers the examples enumerated above, but includes also other members. To make the ideas accessible, we will instantiate them in a simple language throughout several examples. Nevertheless, the reader should keep in mind that different languages satisfy different properties. These properties, their motivation, and some of the consequences of choosing them are discussed in further detail in Section 5.

For our purposes, an ontology language is a logical language which describes a knowledge domain by imposing constraints on how the underlying terms can be interpreted. Looking at ontology languages in this general form allows us to handle all these languages (and some more) simultaneously, without having to worry about their specific properties. In this way, we can also understand how the probabilistic component affects the notions within a language. A similar general definition of ontology has been used before, in a different context [60].

Formally, an *ontology language* is a tuple $\mathfrak{L} = (\mathfrak{A}, \mathfrak{O}, \mathfrak{C}, \models)$, where $\mathfrak{A}$ is a countable set of well-formed *axioms*, $\mathfrak{C}$ is the set of *consequences*, $\mathfrak{O} \subseteq \mathscr{P}(\mathfrak{A})$ is the set of *acceptable ontologies*—where an element $\mathcal{O} \in \mathfrak{O}$ is called an *ontology*—and $\models \subseteq \mathfrak{O} \times \mathfrak{C}$ is the *entailment relation*, which will be written using an infix notation. The only constraints imposed into these components are that the class of acceptable ontologies is closed under subsets, and that the entailment relation is monotonic. In formal terms, this means that for all $\mathcal{O}, \mathcal{O}' \subseteq \mathfrak{A}$ such that $\mathcal{O} \subseteq \mathcal{O}'$ and all $c \in \mathfrak{C}$, (i) if $\mathcal{O}' \in \mathfrak{O}$, then $\mathcal{O} \in \mathfrak{O}$, and (ii) if $\mathcal{O}' \in \mathfrak{O}$ and $\mathcal{O} \models c$, then $\mathcal{O}' \models c$. When $\mathcal{O} \models c$ we say that $\mathcal{O}$ *entails* c or that c is *entailed* by $\mathcal{O}$. Although this definition does not require it, we will often assume that an ontology is finite; that is, that it contains finitely many axioms. This makes sense in the context of knowledge representation.

Some Ontology Languages

We now present some examples of ontology languages, which will help clarify our general definition and the choices made in future sections.

Graphs. One of the simplest ontology languages one can think of is that of graphs [29]. Formally, given a countable set V of *nodes* we define the ontology language $\mathfrak{L}_{\mathsf{G}} := (\mathfrak{A}_{\mathsf{G}}, \mathfrak{O}_{\mathsf{G}}, \mathfrak{A}_{\mathsf{G}}, \models)$ where $\mathfrak{A}_{\mathsf{G}} := V \times V$, $\mathfrak{O}_{\mathsf{G}} := \mathscr{P}(\mathfrak{A}_{\mathsf{G}})$, and $\models$ is the reachability relation; that is, for a graph $\mathcal{G} \subseteq \mathfrak{A}_{\mathsf{G}}$ and nodes $u, v \in V$, we have that $\mathcal{G} \models (u, v)$ iff v is reachable from u in $\mathcal{G}$.

In this case, the axioms of the ontology are the edges in the graph, and an ontology is a set of edges. Keeping in line with the assumption mentioned before, we often restrict to finite graphs. Note that for this language—and for others which we will present later—the class of axioms and the class of consequences coincide. However, their elements are interpreted in a different manner; the pair (u, v), when seen as an axiom, represents an edge in the graph, while as a consequence it represents a question about reachability. We will see in further examples that these two classes need not coincide in general.

The attentive reader may already be wondering why we have decided to use a class of acceptable ontologies $\mathfrak{O} \subseteq \mathscr{P}(\mathfrak{A})$ rather than allowing any set of axioms to be an ontology. The reason is, of course, that for some applications, we might want to avoid an overhead or other issues caused by arbitrary sets of axioms. For example, if in our application we are only interested in *acyclic* graphs, or in *trees*, we may simply restrict the class of acceptable ontologies to reflect this fact; e.g., in the former case, we can define $\mathfrak{L}_{\mathsf{AG}} := (\mathfrak{A}_{\mathsf{G}}, \mathfrak{O}_{\mathsf{AG}}, \mathfrak{A}_{\mathsf{G}}, \models)$ where $\mathfrak{A}_{\mathsf{G}}$ and $\models$ are as before, but now $\mathfrak{O}_{\mathsf{AG}} := \{\mathcal{G} \subseteq \mathfrak{A}_{\mathsf{G}} \mid \mathcal{G} \text{ is acyclic}\}$. Note that the class $\mathfrak{O}_{\mathsf{AG}}$ is closed under subsets as required by the definition of ontology languages; that is, if $\mathcal{G}$ is an acyclic graph, then every subgraph of $\mathcal{G}$ is also acyclic. We will see other examples where considering only a restricted class of acceptable ontologies may become fundamental for the effectiveness of entailment decisions.

Prolog. Another well known formalism which can be considered an ontology language is Prolog. Here we briefly introduce only a restricted version to give the general idea. A (simplified) *Horn clause* is a logical implication of the form $\forall \mathbf{x}.(\exists \mathbf{y}.\varphi(\mathbf{x}, \mathbf{y}) \to p(\mathbf{x}))$ where $\mathbf{x}$ and $\mathbf{y}$ are vectors of variables, φ is a conjunction of predicates using the variables in $\mathbf{x}$ and $\mathbf{y}$, and p is a single predicate. A *fact* is a term $p(\mathbf{a})$, where p is a predicate and $\mathbf{a}$ is a vector of constant symbols. The language $\mathfrak{L}_{\mathsf{P}}$ is defined by the tuple $(\mathfrak{A}_{\mathsf{P}}, \mathfrak{O}_{\mathsf{P}}, \mathfrak{C}_{\mathsf{P}}, \models)$ where $\mathfrak{A}_{\mathsf{P}}$ is the class of all Horn clauses and facts, $\mathfrak{O}_{\mathsf{P}} := \mathscr{P}_{\mathsf{fin}}(\mathfrak{A}_{\mathsf{P}})$ is the set of all finite subsets of $\mathfrak{A}_{\mathsf{P}}$, $\mathfrak{C}_{\mathsf{P}}$ is the class of all facts, and $\models$ is the standard entailment relation for first-order logic, restricted to the simple implications involved. Note that in this case, the class of consequences is a strict subset of the class of axioms. In Prolog, we are not interested in finding other clauses which can be derived from an ontology, but only new facts which are implicitly encoded in it.

Databases. To include a formalism that is not typically considered an ontology language, but still fits within our definition, we consider (relational) databases. In this setting, we have a database, which is composed by a set of tables which satisfy a relational schema. The tables in the database can be abstracted to a set of ground terms $p(\mathbf{a})$ (or facts, as in the previous setting), where the arity and other properties of the predicate p are constrained by the schema, which is essentially a set of first-order formulas. Under this view, the set $\mathfrak{A}_{\mathsf{DB}}$ of axioms of the database ontology language $\mathfrak{L}_{\mathsf{DB}}$ contains all the possible facts and schema formulas; the class of acceptable ontologies $\mathfrak{O}_{\mathsf{DB}}$ includes all finite sets of axioms where the facts comply with the schema, the class of consequences $\mathfrak{C}_{\mathsf{DB}}$ is the set of all positive Boolean queries, and $\models$ is the standard entailment relation from databases to queries. In this case, the classes of axioms and consequences are different. Note also that we restrict $\mathfrak{C}_{\mathsf{DB}}$ to contain only positive queries. Indeed, if we allowed arbitrary queries, then the closed-world semantics of databases would violate the monotonicity condition over $\models$; that is, a query which uses negation may follow from a given database, and not follow anymore after a single fact has been added to it.

$\mathcal{HL}$. As a running example for this tutorial, we will consider a simple ontology language which combines the properties of (hyper)graphs and databases, with the semantics similar to what Prolog programs can do. The logic itself takes its form as a very inexpressive description logic (and hence the name), and also fits within some limited definitions of knowledge graphs (see e.g. [5]). Consider two disjoint sets N_C and N_I of *concept names* and *individual names*, respectively. An *inclusion axiom* is an expression of the form $S \to T$, where $S \subseteq N_C$ is a finite set of concept names and $T \in N_C$; a *fact* is of the form $A(a)$ with $A \in N_C$ and $a \in N_I$. We consider both, inclusion axioms and facts as axioms of our ontology language; that is, $\mathfrak{A}_{\mathsf{HL}}$ is the set of all inclusions and facts. As acceptable ontologies, we allow all finite subsets of $\mathfrak{A}_{\mathsf{HL}}$: $\mathfrak{O}_{\mathsf{HL}} := \mathscr{P}_{\mathsf{fin}}(\mathfrak{A}_{\mathsf{HL}})$. Intuitively, an $\mathcal{HL}$ ontology is composed of a finite hypergraph (defined by the inclusion axioms) and a finite set of facts; i.e., $\mathcal{HL}$ combines hypergraphs with data. For that reason, we will henceforth often refer to inclusion axioms as *hyperedges*, and a set of inclusion axioms as a *hypergraph*. The facts that form the data can be thought of as being separated from the hypergraph, which forms a kind of schema. As consequences, we consider the class $\mathfrak{C}_{\mathsf{HL}}$ of all facts. Variants of this inexpressive logic have been used to showcase the properties and understand the complexity of non-standard reasoning problems in ontology languages [58,59,61].

Strictly speaking, the inclusion axioms define a special kind of directed hypergraph [31], where hyperedges are formed by a set of *sources* and a single *target* node; i.e., given a set V of nodes, a hypergraph is a set of pairs of the form (U, v) where $U \subseteq V$ and $v \in V$. In such a hypergraph, a *path* from a set U of nodes to a node v is a sequence of hyperedges $(U_0, v_0), (U_1, v_1), \dots, (U_n, v_n)$ such that (i) $v_n = v$ and (ii) for each $i, 0 \leq i \leq n$ it holds that $U_i \subseteq \{v_j \mid j < i\} \cup U$. The node v is *reachable* from the set U iff there exists a path from U to v. In this setting, we say that an ontology $\mathcal{O}$ entails the fact $A(a)$ (i.e., $\mathcal{O} \models A(a)$) iff A is reachable from some set of concepts S in the hypergraph defined by $\mathcal{O}$, such that $\{B(a) \mid B \in S\} \subseteq \mathcal{O}$. For example, Figure 4 shows two hypergraphs representing $\mathcal{HL}$ ontologies. In both cases, if an ontology contains the facts $\mathsf{Nausea}(a)$ and $\mathsf{MemoryLoss}(a)$, we can derive the consequence $\mathsf{Observation}(a)$. The details on how to derive a consequence efficiently are explained better in Section 6. The choice of the language $\mathfrak{L}_{\mathsf{HL}} := (\mathfrak{A}_{\mathsf{HL}}, \mathfrak{O}_{\mathsf{HL}}, \mathfrak{C}_{\mathsf{HL}}, \models)$ as a guiding example is motivated by its simplicity, and the properties it preserves in relation to other known languages. We will make full use of all these properties in Section 6.

Others As mentioned already, there exist many other logic-based formalisms which fit into our definition of ontology languages. For the sake of inclusiveness, and to motivate the generality of our discourse, we briefly mention some of them without going into much detail. The interested reader can dive deeper into any of these formalisms consulting the references provided. Before going into less obvious formalisms, we simply specify that all classical *description logics* (DLs) [2] fall into our definition of ontology language. Typically, a DL ontology is composed of a so-called TBox, which encodes the terminological knowledge of the domain, and an ABox containing all the known facts about specific individuals. Although in most cases an ontology is simply a set of axioms of these forms, it is sometimes important to restrict the class of acceptable ontologies. For example, to guarantee decidability of its reasoning services, $\mathcal{SROIQ}$ forbids combinations of some axioms involving role names (for details, see [40]). In smaller members of the family, limiting the TBox to be *acyclic* can further reduce the complexity of reasoning as well; e.g., in $\mathcal{ALC}$ the complexity drops from ExpTime for general TBoxes [66], to PSpace for acyclic ones [3]. Orthogonally, the different logics studied under the umbrella of knowledge graphs also fit this definition [25, 39].

Another natural formalism to be considered is propositional logic. In this case, one can for example consider as axioms the set of all clauses, which makes a CNF formula—that is, a set of clauses—an ontology. As consequences one can consider other clauses including the empty clause used to decide satisfiability of a formula. These consequences can be derived or tested through standard methods from satisfiability testing [6]. It is perhaps less obvious to think of a temporal logic like LTL as an ontology language. In [3], LTL was used as an example of ontology language to show the power of the automata-based methods it proposed. In this context, an ontology is a set (seen as a conjunction) of temporal constraints. Independently, a language known as Declare was proposed for modelling constraints and requisites in a business process [52, 62]. Declare is a graphical language, whose underlying formal semantics is based on LTL; specifically, a Declare model is a set of temporal constraints which is interpreted as the conjunction of LTL formulas (over finite time bounds) which underlie the Declare constraints. There exist, obviously, many other ontology languages that fit within our setting. Our goal is not to enumerate them all, but rather to show the generality of the definitions and of the solutions and settings that will be studied later on.

Before moving on to the main topics of this tutorial, we discuss briefly the importance of the entailment relation. In general, when studying an ontology language, one is interested in developing methods which can decide whether an ontology entails a given consequence or not. Although our definition does not require so, we are usually interested in cases where this entailment relation is decidable—in fact, all the examples mentioned in this section satisfy this property—and in developing effective methods for deciding it. When possible, these methods are also expected to be optimal w.r.t. the computational complexity of the problem. When studying ontologies languages in an abstract sense as we are doing now, all these subtleties are often lost, but that does not mean that they are not important. When possible, we will try to remind the reader of these complexity issues.

3 Uncertainty

A commonly mentioned limitation of ontology languages—specially within the context of knowledge representation—is their inability to model or handle uncertainty. Indeed, notice that we use the axioms in an ontology as absolute information, and the consequences follow (or not) from these axioms. This leaves no space for statements which are not completely certain. When dealing with natural knowledge domains, and in particular when the knowledge is built with the help of different knowledge experts and automated tools, it becomes impossible to avoid uncertainty in some axioms of every form.

Consider for example a medical application. In a very simple scenario, one could try to model the knowledge that patients showing a running nose and complaining of feeling light-headed have a common cold; for instance through an $\mathcal{HL}$ axiom like $(\{\mathsf{RunNose}, \mathsf{LightHead}\}, \mathsf{Cold})$. This rule would be correct most of the time, but ignores the fact that there exist many different maladies—some of them potentially serious—who share these same symptoms, and require additional interventions. Limiting the knowledge to this simple case could be very problematic in practice. It also ignores other potential symptoms, and additional information by patients. It would be much better to weaken the rule to express that these symptoms are *likely* associated to a common cold (with an appropriate measure of what the term "likely" means, as we will discuss later).

Importantly, uncertainty is *not* unique to medicine. As argued in the introduction, we are in fact surrounded by it. From weather reports, to potentially defective pieces of equipment, to accidents and vehicle traffic, we continuously consider and handle uncertain scenarios as any intelligent agent with automated reasoning should. Consider for instance a package delivery company which needs to schedule shipments and guarantee delivery times. All the factors just enumerated would need to be taken into account to minimize the risk of delays. Hence, it is fundamental to be able to include this uncertainty in a formal and manageable manner into our ontology languages.

Before going forward, it is worth to discuss briefly what we mean by *uncertainty*. Uncertainty is used to handle scenarios that happen or not but, due to a lack of knowledge and other factors, we do not know which is the case. That is, the outcome is precise, but our knowledge about it is not. The typical example for uncertainty is the toss of a coin: it will land in heads or not, but before performing the experiment one cannot know which one is. On the other hand, uncertainty is not about imprecise or other kinds of imperfect knowledge. For example, notions which have no precise definition such as "tall" or "close" do not express uncertainty, but rather imprecision. It is not that we do not know whether someone is tall or not, but rather that we cannot precisely define when a person is tall and when they are not.

There exist many different formalisms which can be used for measuring and handling uncertainty. Perhaps the two most prominent ones are possibility theory [30] and probability theory, where probabilities tend to be better known by average users, even though our intuitive understanding of probabilities often tends to be wrong. Since this tutorial is about probabilistic ontologies, we will consider only probability theory from now on. For completeness, before entending ontology languages with probabilities, we briefly introduce the main notions of probability theory. For a gentle introduction to the topic, we suggest [69].

3.1 Basics of Probability Theory

Probability theory measures the likelihood of occurrence of potentially interconnected events of which we do not have full certainty. The space forming these events is called the sample space. Although in the context of logic-based knowledge representation, one usually considers only *discrete* sample spaces, it makes sense to try to understand the general view on probability theory to be able to model also more complex scenarios. Nonetheless, we will try to preserves the intuition of discrete spaces as much as possible, to aid the understanding of readers who are not fully familiar with the topic.

Consider a non-empty, potentially infinite set Ω of *outcomes*, which is called the *sample space*. A *σ-algebra* over Ω is a class Σ of subsets of Ω such that $\Omega \in \Sigma$ and Σ is closed under set complementation and countable unions; that is, for all $X \subseteq \Omega$, if $X \in \Sigma$, then $\Omega \setminus X \in \Sigma$ as well; and for any countable sequence of sets $X_0, X_1, \ldots \subseteq \Omega$ the union $\bigcup_{n\in\mathbb{N}} X_n \in \Sigma$. If Σ is a σ-algebra over Ω, we call the pair (Ω, Σ) a *probability space*, and the elements of Σ are called *events*. From now on, we consider an arbitary but fixed probability space (Ω, Σ). For example, when we throw a die, the sample space is $\Omega := \{1, \ldots, 6\}$. A potential σ-algebra over this set is the powerset $\mathscr{P}(\Omega)$. Hence, for instance, the set $\{2, 4, 6\}$ is an event in this probability space which refers to the case where the roll of the die lands on an even number.

A *probability measure* is a function $P : \Sigma \to [0, 1]$ such that (i) $P(\Omega) = 1$, and (ii) if $\{X_0, X_1, \ldots\} \subseteq \Sigma$ is a countable collection of pairwise disjoint events, then $P(\bigcup_{n\in\mathbb{N}} X_i) = \sum_{n\in\mathbb{N}} P(X_i)$. In words, these conditions state that the event of all possible outcomes has probability 1—which is intuitively understood as being certain—and that the probability of the union disjoint events can be computed by adding the probabilities of each of the events. Based on these two conditions, we can derive all of the well-known properties of probabilities. For example, that for every two events $X, Y \in \Sigma$, it holds that $P(\Omega \setminus X) = 1 - P(X)$ and $P(X \cup Y) = P(X) + P(Y) - P(XY)$.[1]

Note that probabilities are defined over events, which are sets of outcomes, and not over individual outcomes in general. This is specially important when the sample space is uncountable, as it avoids the need to provide a probability value to each possible outcome. If such an assignment was required, then only a countable amount of outcomes could receive a positive probability. A simpler case is obtained when Ω is at most countable. In that case we can assume without loss of generality that the σ-algebra is simply $\mathscr{P}(\Omega)$; the class of all subsets of Ω. Moreover, it suffices to define the *probability mass function*; that is, a probability $P(\omega)$ for each outcome $\omega \in \Omega$. This implicitly yields the measure

[1] As is standard in probability theory, we use concatenation to denote intersection. That is, XY stands for the event $X \cap Y$.

$P(X) = \sum_{\omega \in X} P(\omega)$ for all $X \subseteq \Omega$. This is called a *discrete probability measure*. Returning to our die example, since the sample space is finite, we can define a discrete probability measure by assigning a probability to each outcome. If the die is fair, we can assign $P(i) = 1/6$ for each $i, 1 \leq i \leq 6$. In that case, we can compute the probability of observing an even number as

$$P(\{2,4,6\}) = P(2) + P(4) + P(6) = 3/6.$$

From the point of view of discrete probabilities, it is easy to give an intuitive reading of the notion of an event. Recall that an event is a set of outcomes, and that in a discrete probability measure, the probability of the event is just the sum of the probabilities of the outcomes it contains. This means that when we speak about the probability of an event, we are in fact measuring the likelihood of observing at least one of its outcomes.

For most notions used in ontologies and in particular in this tutorial, it suffices to consider discrete probabilities. Hence, we suggest any reader who is not familiar with probabilities or measure theory to think of this case mainly. The continuous case becomes relevant in some very specific applications and settings only.

3.2 Conditional Probabilities

A very important notion in probability theory is that of *conditioning*, which can be thought of as the adaptation of logical implication to probabilities. Intuitively, the conditional probability of X *given* Y (in symbols $P(X \mid Y)$) expresses the likelihood of observing the event X under the assumption that the event Y holds. Note that assuming the truth of Y immediately removes the possibility of many outcomes and events; in essence, any outcome that does not belong to Y is forbidden, as it contradicts the assumption that one of the outcomes in Y holds. When conditioning, we redistribute the probability of the remaining events, preserving their proportionality within the new probability space.

Formally, conditioning over an event Y defines the new probability space of outcomes in Y $(Y, \Sigma_{|Y})$ where $\Sigma_{|Y} = \{X \cap Y \mid X \in \Sigma\}$. As said before, the probability of the new events is considered to remain proportional, within the remaining set of outcomes. Hence, $P_{|Y}(X \cap Y) = P(X \cap Y)/P(Y)$. Note, however, that this definition creates a probability distribution over a new space, but we would rather prefer to be able to speak about the events of the original space. That is why the *conditional probability given* Y is defined, for every event X, as $P(X \mid Y) := P_{|Y}(XY) = P(XY)/P(Y)$. In particular this means that for any two events X, Y, $P(XY) = P(X \mid Y)P(Y)$ holds. Considering again the example with the die, if $Y = \{2,4,6\}$ is the event of observing an even number, and $X = \{1,2,3\}$ refers to observing at most a 3, then

$$P(X \mid Y) = P(XY)/P(Y) = P(\{2\})/P(Y) = \frac{1/6}{1/2} = 1/3.$$

In this case, $P(X) = P(\overline{X}) = {}^1\!/_2$ but $P(X \mid Y) = {}^1\!/_3$ and $P(\overline{X} \mid Y) = {}^2\!/_3$.[2] That is, conditioning over another event may increase or decrease the probability of a given event.

We say that two events X, Y are *independent* iff the occurrence of one does not affect the probability of the other one. More formally, X and Y are independent iff $P(XY) = P(X)P(Y)$. Note that this means that for independent events, $P(X \mid Y) = P(X)$ and $P(Y \mid X) = P(Y)$. For example, the events $X = \{2, 4, 6\}$ and $Y = \{1, \ldots, 4\}$ are independent. Indeed, $XY = \{2, 4\}$ and we can see that $P(XY) = {}^1\!/_3 = {}^3\!/_6 \cdot {}^4\!/_6 = P(X)P(Y)$. We extend this notion of *conditional independence* as follows. The events X and Y are *conditionally independent given* Z iff $P(XY \mid Z) = P(X \mid Z)P(Y \mid Z)$. This extension is very natural, but requires that the conditioning random variable is preserved through all the elements of the equation. Through a simple computation, it is easy to verify that for any three events X, Y, Z it holds that $P(XYZ) = P(X \mid YZ)P(Y \mid Z)P(Z)$. This idea can be easily generalised to any finite set of events. Together with conditional independence, equations of this kind will be fundamental in Section 3.4.

A cautious reader would have already detected an anomaly in the definition of conditional probabilities: the conditional probability given Y is only well-defined in case $P(Y) > 0$. This anomaly can be understood from our intuitive interpretation of probabilities. An event with probability 0 is one which is almost impossible to be observed, and hence assuming it to be true is akin to assuming a contradiction as the premise of a classical logical implication. In logic, this is solved by stating that everything follows from a contradiction; in probabilities, we simply disallow that special case, to guarantee that the properties of a probability distribution still hold after conditioning.

An important property of conditioning, which does not apply for classical implications is its reversibility: with enough information, we can change the conditioning and the conditioned events. Specifically, recall from the definition of conditioning that $P(X \mid Y)P(Y) = P(XY)$, but then, obviously, it is also the case that $P(Y \mid X)P(X) = P(XY)$. From these two equations we can deduce that $P(X \mid Y)P(Y) = P(Y \mid X)P(X)$. This equation, commonly known as Bayes' rule when expressed as

$$P(X \mid Y) = \frac{P(Y \mid X)P(X)}{P(Y)}$$

allows us to reverse a conditioning statement if we know the probabilities of each of the events separatedly.

3.3 Boolean Random Variables and Joint Distributions

It is often convenient to think of events X as *Boolean random variables*. Although the notion of a random variable is much more complex, it suffices for the purpose of this tutorial to equate events and random variables. More precisely, we will

[2] Again, as usual in probability theory, $\overline{X}$ denotes the event $\Omega \setminus X$; that is, the negation of X.

often speak about the Boolean random variable X to refer to a given event X. The reason is that, seen as a (Boolean) random variable, X can have two states: $\mathtt{t}$ when X holds, and $\mathtt{f}$ otherwise; that is, $\mathtt{f}$ denotes the fact that the event $\Omega \setminus X$ holds. Then X defines a probability distribution over its two states. For brevity, we will write X when the random variable is in state $\mathtt{t}$, and $\overline{X}$ when it is in state $\mathtt{f}$. From now on, whenever we speak about a random variable, we refer to a Boolean random variable in this sense.

Given a finite collection of random variables $X_1, \dots, X_n$, we can build the *joint probability distribution*, which assigns a probability to each possible combination of states of the variables. Note that the specific distribution depends on the events underlying the random variables, and their probabilistic dependencies. Still, some general operations can be applied over the joint distribution. One of the most important is *marginalization*, which allows to remove a random variable from the distribution. Specifically,

$$\begin{aligned} P(X_2, \dots, X_n) &= P(X_1, X_2, \dots, X_n) + P(\overline{X}_1, X_2, \dots, X_n) \\ &= P(X_2, \dots, X_n \mid X_1) P(X_1) + P(X_2, \dots, X_n \mid \overline{X}_1) P(\overline{X}_1). \end{aligned}$$

3.4 Bayesian Networks

Note that when building the joint probability distribution of n random variables, there is no specific pattern that the assigned probabilities should show. In the worst case, the description of the full joint probability distribution would require to provide a value for each of the 2^n different combinations of states of the random variables; that is, for all the variants of making some variables $\mathtt{t}$ and some $\mathtt{f}$. This exponential growth becomes easily unmanageable after a few tens of variables are incorporated. For that reason, different approaches have been proposed for describing joint probability distributions more succinctly. A very prominent such approach, which exploits some conditional independence assumptions, is Bayesian networks [27, 57].

In a nutshell, Bayesian networks use a graphical representation, through a directed acyclic graph (DAG) to express some conditional independence relationships between random variables (which appear as nodes in the DAG), which allow to limit the explicit probabilistic dependencies required to express the full joint probability. Formally, a *Bayesian network* is a pair $\mathcal{B} = (\mathcal{G}, \Phi)$ where $\mathcal{G}$ is a DAG whose nodes represent Boolean random variables, and Φ is a collection of conditional probability distribution tables, containing exactly one table for each node X given its parents $\pi(X)$ on G. Note that each of the tables in Φ is still exponential in the number of parents $\pi(X)$. However, assuming that the maximum in-degree in $\mathcal{G}$ is bounded, these tables are more easily handled than a full joint probability distribution table.

The graphical structure of the DAG $\mathcal{G}$ encodes an underlying conditional independence assumption between the random variables; namely, that every node is independent of all its non-successors given its parents. In other words, if the state of all the parent nodes $\pi(X)$ is known, then knowledge about any other

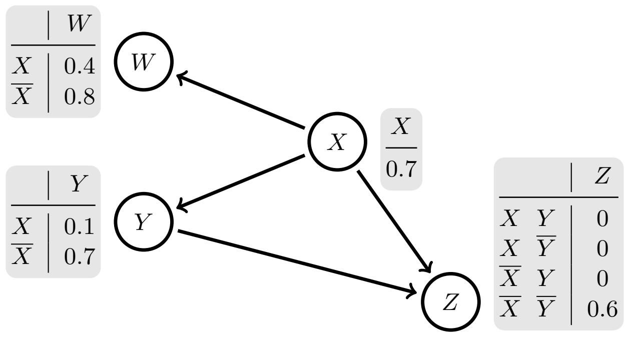

Fig. 1. A very simple Bayesian network

variable Y that is not reachable from X does not affect the likelihood of observing X itself; in formulas $P(X \mid Y\pi(X)) = P(X \mid \pi(X))$. Under this assumption, the whole joint probability distribution can be factorised as the product of all the conditional distributions appearing in Φ. That is,

$$P(\mathbf{X}) = \prod_{X \in \mathbf{X}} P(X \mid \pi(X))$$

where $\mathbf{X}$ is the set of all nodes from $\mathcal{G}$.

A very simple example of a Bayesian network is depicted in Figure 1. In this case, the graph has four variables, and each node is depicted together with its conditional probability table. Note that describing the same joint probability distribution as a table would require 16 rows. Even at the scale of this example, we are saving almost half the space. The graphical structure expresses that the variable W is conditionally independent of both Y and Z given X. That is, if we know the state of X, any further information on Y on Z will not affect the probability of W. We can compute the probability of each possible state; for example,

$$\begin{aligned} P(W, X, \overline{Y}, \overline{Z}) &= P(W \mid X)P(X)P(\overline{Y} \mid X)P(\overline{Z} \mid X\overline{Y}) \\ &= 0.4 \cdot 0.7 \cdot 0.9 \cdot 1 = 0.252. \end{aligned}$$

Probabilistic inferences over BNs have been thoroughly studied in the literature. The simplest inference corresponds to the update of the probability distribution given some evidence; that is, computing the posterior distribution under the observation of the state of some variables of the network. In a nutshell, once we know the state of a given variable, we can use the rules of conditional probability to update our beliefs about the rest of the network, through a sort of message-passing mechanism. The same approach can also be used to compute the marginals, which in BNs is called *variable elimination.* Recall that to obtain the marginal w.r.t. a given variable X, it suffices to compute the posterior w.r.t. to each of the two possible states of X, as well as the prior probabilities of these states. Hence, it suffices to assume that we are given evidence of each of the states, and propagate this information through the network. There are many other inferences studied in the context of BNs; for instance, to find whether there are instantiations of a subset of $\mathbf{X}$ which yield a given probability of observing another variable (with or without evidence). We refer the interested reader to [27] for further details.

Importantly, although in the general case making any of these inferences is computationally hard [22, 46, 65], under some conditions on the structure of the underlying graph they remain tractable. For example, if the graph is a tree, or closely resembles a tree (formally, if it has a bounded *treewidth*), then there are efficient methods for propagating evidence and deriving the posterior distribution, even reducing the complexity of inferences to linear time. For example, the DAG in Figure 1 is not a tree, but the only situation violating the tree condition is a join of two paths in a terminating node; thus, it has a low treewidth. From these, other inferences become also simpler [28].

4 Probabilistic Ontologies

We have so far introduced a general notion of ontology languages, and probabilities as a prominent theory for handling uncertainty. We now want to merge them together to be able to build and use probabilistic ontologies. At first sight, this looks as a pretty straightforward task: simply take an ontology language, and allow for probabilistic statements. However, once we try to fill in the technical details, we see that things are not as easy as they seem, and the road is paved with subtleties.

There is in fact a large space of possibilities from which to choose when extending an ontology language with probabilities. This is attested by the many different probabilistic ontology languages that can be found in the literature (see e.g. [48] for a slightly outdated survey focusing on the Semantic Web, and [19] for a more recent tutorial on probabilistic knowledge bases from a different perspective). It is very important, then, that one is aware of these choices and why they are made, lest we obtain unintended consequences from our knowledge. Before considering these choices, we introduce formally what we mean by a probabilistic ontology in the context of this work.

Definition 1 (probabilistic ontology). *Let $\mathfrak{L} = (\mathfrak{A}, \mathfrak{O}, \mathfrak{C}, \models)$ be an ontology language. A* probabilistic ontology *is a pair $(\mathcal{O}, P)$ where $\mathcal{O} \in \mathfrak{O}$ is an ontology and $P : \mathcal{O} \to [0, 1]$ is a partial function assigning a* probability degree *to some axioms in the ontology.*

Note that probabilistic ontologies naturally generalise classical ones. Indeed, the ontology $\mathcal{O}$ can be equivalently seen as the probabilistic ontology $(\mathcal{O}, P_\emptyset)$ where $P_\emptyset$ is the empty partial function, which assigns a value to *none* of the elements of $\mathcal{O}$.

Intuitively, a probabilistic ontology is just an ontology where some of the axioms are labelled with a value in the interval $[0, 1]$, which will be interpreted as their probability; that is, for a given axiom $\alpha \in \mathfrak{A}$, $P(\alpha)$ expresses the probability of α being true. Importantly, the assignment of probabilities P is a partial function, which means that not all axioms are required to get such a degree. This is fundamental to allow for cases where parts of the ontology are required to be certain. For example, in some probabilistic DLs one may expect to have only uncertain data, but certain terminological knowledge; that is, only the ABox assertions may be assigned a probability degree. Another prominent formalism that makes this restriction is ProbLog [63], a probabilistic variant of Prolog which allows facts, but not rules, to be assigned a probability. Similarly, in probabilistic databases the data tuples are uncertain, but the database schema is not [68].

One could think it is more natural to make the probabilistic assignment a total function, which assigns a probability degree to *all* the elements of $\mathcal{O}$, and using the degree 1 to express certainty. While this corresponds to the usual, intuitive understanding of probabilities—where probability 1 means certainty—it does not correspond to the mathematical properties of probabilities and, depending on the semantics chosen, may in fact produce different results. Indeed, formally speaking a probability 1 only means *almost certainty*, while probability 0 is assigned to *almost impossible* events.

Excluded Formalisms

Through our chosen notion of probabilistic ontologies, where some axioms are labelled with a real number between 0 and 1, we have in fact excluded many formalisms which have been—or could be—considered as probabilistic ontology languages as well. Prominently, it leaves out all formalisms based on log-linear interpretations of probabilities [9, 55]. In those formalisms, probabilities are not assigned directly, but are rather implicitly encoded in *weights*, which can be arbitrarily large real numbers. The resulting probability is proportional to the weights used, but through a logarithmic factor, which makes it difficult to interpret the original weights. We could easily extend Definition 1 to allow these weights, but decided not to do so for clarity and conciseness of the presentation.

When trying to identify the likelihood of an axiom to hold, we often encounter the problem of finding a precise number which can be used without damaging the accuracy of the results. While the probability assigned by the function P can be interpreted as a lower or upper bound for the actual probability, in some scenarios it may become important to specify both bounds of a probability interval [16,23]. Expressing these bounds requires at least two labels per axiom, and hence does not directly fit into Definition 1. Of course, it would not be difficult to allow two functions P_L and P_U expressing the lower and upper bounds. We chose not to do so to avoid the notational overhead that it induces, but most of what will be discussed henceforth applies for this setting as well.

An important consideration is that our notion of probabilistic ontologies is intended to express uncertain knowledge, but not knowledge about uncertainty. The difference, which may seem small at first sight, is clearly observable from our definition of a probabilistic ontology: it is the knowledge (expressed through axioms) which is assigned a probability degree, but these degrees are not directly

Table 1. The five golden rules

	The Rules
1	Use probabilities
2	Use the right probabilities
3	To count or not to count
4	Understand the numbers
5	Be careful with independence

used within the axioms. For example, through a probabilistic ontology (based on an adequate ontology language) one can express that a person with the flu has a probability 0.8 of showing fever, but cannot express that if one has a parasitic infection with probability 0.5, then one will show fever with probability 0.8. The reason for our exclusion is that in the latter case the probabilities are an intrinsic element of the ontology language; the axioms themselves can express them already. In other words, probabilities are fundamental in the construction of the underlying ontology language. Thus, languages capable of expressing those statements (see e.g. [33, 34, 49]) fall into the basic definition of an ontology language from Section 2, rather than on the probabilistic extension of Definition 1.

Now that we have a syntactic definition of a probabilistic ontology language, we need to know how interpret these probabilistic ontologies and, more importantly, how to decide whether—and with which probability—an entailment follows from them. In addition, thinking about probabilities introduces new reasoning problems which should also be studied in these languages. As hinted already, depending on the requirements of the language and its inferences, there exist many different semantics which one could choose for interpreting the probabilities, each with a particular application scenario. Rather than enumerating them all directly, we prefer to provide five directives (the *Five Golden Rules*) which should be kept in mind when choosing the semantics, along with examples and the reasoning behind them. These examples will help the reader find probabilistic ontology languages that fall into our definition, in contrast to the excluded ones enumerated above.

5 The Five Golden Rules

Here are the five golden rules to consider when designing, choosing, or using a probabilistic ontology or probabilistic ontology language. For ease of consultation, they are also enumerated in Table 1.

5.1 Use probabilities

This is the fundamental rule, and one who puzzles researchers and practitioners who approach the area of imperfect knowledge representation for the first time. As we have defined them, probabilistic ontologies assign a real number in $[0, 1]$

to some of the axioms. As probabilities are commonly taught at schools, and we often hear probabilistic terminology in our daily lives, it is natural to associate any annotations that fall within this interval the meaning of probabilities, and this is often done in practice. However, there exist many other measures which use the same scale, but have nothing to do with probabilities or even with uncertainty.

Recall first that probabilities are a measure of *uncertainty*, meaning that they refer to events which either hold or not, and our lack of knowledge is only about which of these two possibilities holds. This goes in contrast with notions like *vagueness* or *graded truths*, where there may exist many different intermediate truth degrees, often also expressed with values in the unit interval. For example, uncertainty can express that there is a $2/3$ chance that a rolled die will fall in a number smaller or equal to 4—it either does, or does not—but cannot express imprecise notions such as the concept of *nearness* or *tallness.* The latter are notions which have no precise definition, or clear-cut separations between membership and non-membership (e.g., there is no specific height at which a person starts being tall, and under which they are not). These are studied in fuzzy and many-valued logics [7,8,10,11,32,35,67]. In these logics, it is possible to express a fact like $\mathsf{Tall}(\mathsf{a}) : 0.8$ which should be interpreted as saying that a is tall with a degree 0.8. Although it may be tempting to interpret this degree as a probability, it would be a great mistake. To mention just one of the many differences between these approaches, fuzzy logics are truth-functional, while probabilities are not. That is, in fuzzy logic from the degrees of φ and ψ, one can compute the degree of $\varphi \wedge \psi$, but in probability theory this is not possible unless we know their conditional dependencies. Hence, before we can choose the appropriate semantics for our approach, we need to make sure that we are dealing with uncertainty, and not with some other kind of imperfect knowledge.

After we have confirmed that we are dealing with uncertainty, we still need to verify that probabilities are the right formalism for it. Another formalism for dealing with uncertainty, which was already mentioned before, is possibility theory [30]. While it shares many properties with probability theory, there are some important differences which make the two formalisms incompatible. Without going into too many details, in possibility theory one is more interested in a qualitative expression of the degrees of uncertainty, which is implicitly encoded by the ordering of the degrees used, rather than in the specific quantitative interpretation of these values. More specifically, in possibility theory an axiom with a degree 0.9 should only be considered *more likely* than one with degree, say 0.8, but without any specific reference to *how much more likely* it is. In addition, in possibility theory the negations are also handled differently, through an additional measure called *necessity.*

After we are convinced that probabilities are indeed the formalism that serves our purposes, we still need to make sure that we understand what these probabilities mean, and how they were obtained. Answering these questions is the focus of the remaining golden rules.

5.2 Use the right probabilities

When we are dealing with probabilistic knowledge, and in particular in scenarios where constraints may be shared among several individuals, it is fundamental to know how probabilities and uncertainty relate to each other and, more importantly, the scope of the uncertainty. Consider the following two statements.

1. The probability of a flight being delayed is 0.3
2. The probability of flight ZZ1900 being delayed tomorrow is 0.3

The first one can be understood as expressing that, from the whole universe of available flights, about 30% suffer delays. The second statement, however, cannot be interpreted this way; in fact, we cannot speak about the universe of "tomorrows," as there is only one such day, nor of flights ZZ1900 on that day, as again only one exists. Instead, it should be understood as expressing that from all the possible scenarios for the day (weighted according to their plausibility) about 30% involve a delay of flight ZZ1900.[3]

This difference in interpretations was already noticed by Halpern [36, 37], who divided probabilistic logics into two main types. Type 1 (or *statistical*) probabilities are those that interpret statements about proportions of the world; hence, give a statistical flavour in the most basic sense of the word. Type 2 (or *subjective*) probabilities, on the other hand, interpret the statements using possible worlds; they usually express a degree of belief, which is applied to every single individual without taking into account the rest of the population.

Just as they express different kinds of views on the probability statement, they need to be interpreted differently in the semantics of a probabilistic ontology. In case of statistical probabilities, the usual semantics of the underlying classical formalism need to be overhauled. Indeed, as these statistical statements express the proportion of population elements which satisfy a given property, they implicitly assert a negation as well. In the first example above, saying that 30% of the flights are delayed means that there are also some flights (in fact, 70% of them) which are *not* delayed. This, and the semantic relationships between properties may produce new issues not foreseen in the original semantics. The usual approach to handle these issues is to partition the class of all domain individuals according to *types* (i.e., the properties they satisfy) and verify that the proportional cardinality of each partition is compatible with the associated probabilities. Some existing formalisms which use this semantics in different forms are [17, 41–43, 45, 47, 56].

The semantics of subjective probabilities are often more intuitive both in terms of understanding, and in the development of reasoning methods. In the multiple-world semantics, which are at the bottom of subjective probabilities, one builds situations in which the axiom is "believed" to be true, and others in which it is believed to be false. Each of these situations is then treated as a classical ontology containing all axioms which are assumed to hold, with its

[3] Alternatively, read the second statement as "the probability of rain tomorrow is 0.3."

usual semantics and entailment relation.[4] At this point, every consequence can be given a probability which is based on the probabilities of the subontologies where the entailment holds. The only problem is to identify such probabilities, which depend on the relationships between axioms, and might not be completely obvious. This issue will be discussed in more detail in Section 5.5. For a more general discussion on the differences between probabilistic interpretations, see e.g. [38, 44].

Another important choice to make when dealing with subjective probabilities arises from the difference between an *open-world* and a *closed-world* interpretation of the probabilities and the axioms in the ontology. The simplest and most natural view is the open-world, in which an axiom which was removed from the ontology in one of the possible worlds may still hold if it is entailed by the remaining axioms. In that sense, the probability associated to the axiom serves only as a lower bound, which could be surpassed in case of knowledge redundancy or further information. This view is the one described briefly in the previous paragraph, and is shared by many formalisms including [20, 63, 64]. The closed-world view, on the other hand, requires that in a situation where an axiom α does not hold, this axiom must be interpreted as false. This guarantees that axioms have exactly the probability assigned to them, but may introduce many other issues. First and foremost, the ontology may become inconsistent in the presence of redundancy. Second, it may require increasing the expressivity of the underlying ontology language in ways in which existing methods may stop working. An example of a probabilistic ontology language following this view are probabilistic databases [68].

Importantly, the differences between probabilistic approaches are not always clear-cut, and may not correspond to the first intuition, or the name chosen by those who developed them. For example, ProbDeclare [51], the probabilistic extension of Declare, was developed to describe business process information extracted through a statistical analysis—each constraint is associated with the proportion of traces where it has been satisfied. However, its semantics is based on PLTL_f^0 [50], which uses a multiple-world semantics and may thus seem to represent subjective probabilities. Another example are the so-called open-world probabilistic databases [18], which apply an open-world view on the facts that *do not* appear in the database (i.e., they are not assumed impossible, but simply less likely than a given bound), but a closed-world for those that do appear—in the semantics, when a fact is chosen not to belong to the current world, then they are impossible in the given interpretation.[5]

5.3 To count or not to count

The next consideration arises mainly from the difference between facts and general knowledge. While we have included them all into the name *axiom*, many

[4] Note that by definition of ontology languages, any subset of an ontology is also an ontology; hence this construction works without issues.

[5] The actual semantics of open-world databases is more complex than this, but we wanted to provide just a basic intuition.

```
unreliable(X) <- part-of(X,Y), unreliable(Y).
part-of(computer,chip).
part-of(chip,core).
```

Fig. 2. A Prolog program with one rule and two facts.

ontology languages make a clear distinction between them. In DLs one separates the TBox (terminological knowledge) from the ABox (the facts); databases distinguish the facts from the schema, and Prolog separates facts from rules. Even in our small language $\mathcal{HL}$, one could think of distinguishing the graph from the facts.

Facts can be either used, or not, to derive some consequences from the ontology. But more general knowledge, like rules or TBox axioms have a very different behaviour: they speak about all possible instantiations, and hence may require multiple applications within a single derivation. This behaviour is not problematic in classical ontology languages because entailments do not count; that is, they do not take into account the number of times an axiom may have to be used to derive a consequence. With probabilities, things might change. One must make a choice whether the semantics should count or not the number of applications. This issue is easier explained through examples.

Consider first a simple production scenario where the knowledge includes the statement that any component with an unreliable part is itself unreliable, together with a part-of relationship between structures. This can be expressed e.g., in Prolog through one rule and several facts as shown in Figure 2. In this scenario it makes sense to weaken the rule into a probabilistic statement expressing that the rule holds with probability, say 0.9. That is, that components are slightly resilient to errors in their parts. Suppose that we know that the `core` is unreliable. What should be the probability of the `computer` being unreliable as well? Intuitively, a first rule application should tell us that the `chip` is unreliable with some probability less than 1, and hence, unreliability of `computer` holds with a lower probability than 0.9; assuming independence, we expect this probability to be $0.81 = 0.9 \cdot 0.9$. Thus, the fact that the rule needed to be applied more than once to deduce unreliability of `computer` affects the probability of this conclusion. Under the multiple-world semantics of Problog, one builds several classical Prolog programs, where the rule holds in some of them, and does not hold in others. Note that in every world that makes the rule true, the consequence `unreliable(computer)` holds, and in all others, it does not hold, yielding a probability 0.9 to this consequence. Hence this semantics would produce an unexpected result, simply because the multiple-world semantics does not take into account the number of times the rule is applied. For such a scenario, other semantics would be more meaningful.

Consider now a hierarchical structure expressing weather conditions, with constraints stating that cold and wet weather produces fog, and that in cold weather, urban areas have an increase in smog. Moreover, in situations of fog and smog, there is low visibility. All this can be expressed through the $\mathcal{HL}$ axioms $(\{\mathsf{Cold}, \mathsf{Humid}\}, \mathsf{Fog})$, $(\{\mathsf{Cold}, \mathsf{Urban}\}, \mathsf{Smog})$, and $(\{\mathsf{Fog}, \mathsf{Smog}\}, \mathsf{LowVis})$. Suppose, moreover, that we know for sure that milano is an urban area with high humidity, and we predict with a 90% probability that it will be cold: $\mathsf{Humid(milano)} : 1$, $\mathsf{Urban(milano)} : 1$, $\mathsf{Cold(milano)} : 0.9$. If we are interested in predicting the probability of low visibility, we could try to chain our axioms to the facts, and deduce first a 90% probability for each, fog and smog, and then we need to compute a probability of low visibility based on these two. If we assumed independence, we would get a probability of 0.81 of low visibility. This result has several issues, starting from the fact that fog and smog are not independent (they both depend on the cold factor),[6] but one can already see that the probability of 0.81 is a result of "counting" twice the fact that the probability of cold in Milano is 0.9. Even though this fact is used in two parts of the derivation, it is a mistake to consider the use of both applications as different. All depends on one single fact. In this case, the appropriate action is to consider a semantics that does not take into account the number of applications of an axiom. The multiple-world semantics would fit perfectly well in it.

5.4 Understand the numbers

One of the main criticisms that probabilistic logics receive from other areas of knowledge representation is about the source of the probabilistic knowledge. More specifically, how do we compute the probabilities with which the axioms are labelled? This is a very valid criticism, and is at the heart of the applicability of these formalisms.

In many cases, we can easily justify the use of a probability as a statistical measure of what is expected to be observed. For example, a probabilistic rule under the statistical semantics may express that some proportion of the population satisfies a given property. This proportion may be computed through a full census, or approximated through different sampling techniques. Still, in some other cases the use of arbitrary numbers as probabilities—or as the expressed probabilities—is not really justified. One prominent example is the automatic population of probabilistic databases [53]. In this setting, natural language processing (NLP) techniques are applied to large corpora to extract some facts, and sometimes more advanced knowledge, which is written in the text. An advantage of using NLP is that tools often return a weight associated to each fact, which expresses the confidence on the fact. This measure can be seen as a probability, in the sense that it describes a proportion of successes in the underlying tool for sentences of the kind extracted. However, it does not refer to the likelihood of the piece of knowledge itself. In other words, the tool provides a measure of its certainty that it read the sentence correctly from the corpus, but cannot judge the correctness of this text. To put it bluntly, a well-written but falsity ridden text would provide facts with a higher confidence value than a well-researched

[6] We will come back to this issue in Section 5.5.

piece with grammatical errors. This is natural, since it is not the scope of NLP to do fact checking, but only to process the written text.

Other kinds of probabilities are more problematic. For example, suppose that one is trying to model the subjective belief that a sport expert has on the likelihood of success of a specific team in a championship. As humans, we are not very good at providing adequate probabilities, and will often default back to very specific numbers which we observe continuously: 0.99, 0.9, maybe 0.8. It is very unlikely that someone would measure their confidence as, say, 0.935. Even statistical methods should be taken with a grain of salt. For example, sampling methods—which were used to justify probabilities just one paragraph above—do not really yield any specific probability which can be used directly. Rather, they can provide estimators, some of which are more likely than others, and perhaps some certainty value or confidence interval around them. If one does not understand the meaning of these notions and values, one will easily fall into a rabbit hole of misleading or erroneous interpretations of probabilistic entailments. However, this is a topic that falls outside the scope of this tutorial, and we will not go into further details.

5.5 Be careful with independence

The fifth golden rule refers to the notion of probabilistic independence. As we have seen throughout this section, independence is an important assumption, which is used very often for probabilistic reasoning. The reason is very simple: independence provides a flavour of truth-functionality which is not usually present in probability measures. To be more precise, for two arbitrary events X, Y, knowing $P(X)$ and $P(Y)$ does not suffice in general for knowing $P(XY)$. However, if X and Y are independent, then we know that $P(XY) = P(X)P(Y)$. Since one does not usually have enough information to compute the probability of the intersection of two events, and even if it is known, computing it may require more costly operations, one can consider independence as a simplifying assumption. Unfortunately, this assumption is often overlooked in formalism descriptions, and may not be very realistic in the first place.

We have already seen in Section 5.3 an example where an independence assumption yields a problematic result. An important aspect of that example is that the probabilistically dependent elements are in fact hidden from the user, and the consequence being asked. Recall that in the example we apply two different axioms that require the same fact ($\mathsf{Cold}(\mathsf{milano})$) to be derived. The conclusions of these axioms then made no further reference to this fact, but could be further combined to derive more facts. A user querying for the probability of low visibility may have no idea that the probability of cold has any influence on it, even if they know that low visibility depends on the presence of fog and smog.

To try to convince the reader of the importance of analysing the independence assumption—despite its ubiquity in probabilistic formalisms—we present a simple example. Suppose that we are extracting information about an individual whose first name is *Andrea*. Andrea is a typical male name in Italy, but a typical female name in other parts of the world. Without further information, we cannot be certain about the gender of this individual. Suppose that one analysis concludes that there is a 50% chance that Andrea is *male*, while another one concludes that they have a 50% chance of being *female*. Note that, from our meta-knowledge perspective, these two statements are perfectly consistent with each other. They are so even if we include the constraint that Male and Female are disjoint classes. However, the independence assumption would yield the conclusion that Andrea is in the intersection of Male and Female with probability 0.25 (or yield an inconsistency together with the disjointness axiom) which is clearly wrong.

Still, one should not have the impression that it is necessarily wrong to use an independence assumption. Depending on the application, it may indeed be the right choice to make for different reasons. For example, when it is impossible to obtain a better joint probability distribution between the axioms, or in cases where representing it would be too costly. A prominent formalism where this assumption is used are tuple-independent probabilistic databases [68]. As with classical databases, in the probabilistic variant one expects to handle huge tables efficiently. Obtaining and representing a full joint probability distribution for all the tuples in the database would be prohibitive, and removing the independence assumption results in slower derivation methods. Moreover, as only the data is probabilistic and the goal is to answer queries, the multiple use of a single database tuple in different parts of the derivation is seldom an issue. Hence, while formally speaking the independence assumption is not well justified, its use allows for a practical solution. However, one should always keep in mind the limitations of the assumption, specially when extending the formalism to other uses.

6 A Specific Language

After we have seen some of the most important decisions to make when dealing with a probabilistic ontology language, we now put those notions in practice by defining a probabilistic extension of $\mathcal{HL}$ and studying some of its properties. This probabilistic ontology language can be seen as a new member of the family of Bayesian DLs which have been studied in recent years [12, 20]. It can also be seen as a special case of ProbLog, with a simpler targetted syntax for expressing probabilistic rules and conditional probabilistic statements. Note that this is, in part, an artificial example for showcasing the properties and choices in the construction of ontology languages. In practice one would likely use a different language capable of expressing more complex properties.

Recall that in $\mathcal{HL}$, axioms are either hyperedges or atomic facts. In our case, we require that the probabilistic assignment of a probabilistic ontology is total; that is, every axiom in the ontology receives a probability degree. Formally, a probabilistic $\mathcal{HL}$ ontology is a finite set of $\mathcal{HL}$ axioms, where each axiom is assigned a probability degree in $[0, 1]$. For the semantics, we will choose a multiple-world approach akin to subjective probabilities, under an open-world point of view. In particular, in our approach the number of times an axiom is used to derive a consequence is irrelevant. However, we do not use the standard independence assumption, but rather assume independence only in some elements of the ontology.

Recall that we can see an $\mathcal{HL}$ ontology as being formed of two parts: a hypergraph, and a data store. Our assumption, as in many ontology languages, will be that the hypergraph will be rather small in comparison to the size of the data. Hence, we may include assumptions to manipulate the data efficiently, but can invest more computational resources dealing with the hypergraph. For this reason, we will assume tuple independence on the data (as commonly done in probabilistic databases), but provide a joint probability distribution on the hyperedges. As it will be prohibitive to express this distribution extensionally for moderately large hypergraphs,[7] we use a more compact encoding through a Bayesian network. Interestingly, this representation will allow us to express also some logical dependencies between the axioms. Formally, to achieve this, we will need to make the general notion of a probabilistic ontology slightly more precise, adding a component to the tuple.

A *probabilistic* $\mathcal{HL}$ *ontology* is a tuple $(\mathcal{O}, P, \mathcal{B}, B)$ where $\mathcal{O}$ is an $\mathcal{HL}$ ontology, $P : \mathcal{O} \to [0, 1]$ is the *probability assignment function*, $\mathcal{B} = (\mathcal{G}, \Phi)$ is a Bayesian network with $\mathcal{G} = (V, E)$, and $B : \mathcal{O} \to \mathscr{P}(V)$ is the *context function*, which maps every hyperedge in $\mathcal{O}$ to a set of random variables in the BN $\mathcal{B}$. In essence, the context function associates each hyperedge in $\mathcal{O}$ with a class of states in the BN $\mathcal{B}$. Intuitively, this assignment expresses that the hyperedge α is true whenever all the variables in $B(\alpha)$ are made true in the BN. We require that $P, \mathcal{B}, B$ are *consistent* in the sense that $P(\alpha) = P_{\mathcal{B}}(B(\alpha))$ holds for all hyperedges α.

For example, suppose that we are interested in modelling and reasoning about the knowledge from a medical application where different co-occurring diseases are being followed. In that case, the BN from Figure 1 represents the joint probability distribution of the diseases W, X, Y, Z. For example, variable Z could stand for CTE and W for Alzheimer's. We prefer to keep the names abstract as much as possible to avoid misunderstanding caused by existing knowledge from the user, given the artificial nature of this simplified example.[8] Consider then the probabilistic $\mathcal{HL}$ ontology $(\mathcal{O}_{\mathsf{exa}}, P_{\mathsf{exa}}, \mathcal{B}_{\mathsf{exa}}, B_{\mathsf{exa}})$ where $\mathcal{B}_{\mathsf{exa}}$ is the BN from Figure 1 and $\mathcal{O}_{\mathsf{exa}}$,$P_{\mathsf{exa}}$, and B_{exa} are represented in Table 2. The hypergraph models general relationships between findings, which hold only on specific contexts. For example, in the context $\{X\}$, any patient with halucinations presents also nausea, while the converse relation (that nausea implies halucinations) only holds within the context $\{W, X\}$. Note that this means that the presence of the latter axiom necessarily implies the presence of the former. The probabilities associated to these axioms are given by the probability distribution in $\mathcal{B}_{\mathsf{exa}}$. In fact, $P_{\mathcal{B}_{\mathsf{exa}}}(\{X\}) = 0.7$. An important example is the last hyper-

[7] Recall that the extensional description of a joint probability distribution requires exponential space on the number of variables involved.

[8] It should go without saying, but this is only an example and should in no way be considered medical advice of any kind.

Table 2. An example probabilistic $\mathcal{HL}$ ontology. Note that the context function B_{exa} applies only to the hyperedges.

B_{exa}	P_{exa}	$\mathcal{O}_{\mathsf{exa}}$
$\{Y\}$	0.28	$(\{\mathsf{Pain}\}, \mathsf{Tachycardia})$
$\{W, Y\}$	0.196	$(\{\mathsf{Tachycardia}, \mathsf{Nausea}\}, \mathsf{Observation})$
$\{W, X\}$	0.28	$(\{\mathsf{Nausea}\}, \mathsf{Halucination})$
$\{X\}$	0.7	$(\{\mathsf{Halucination}\}, \mathsf{Nausea})$
$\{W, Z\}$	0.0432	$(\{\mathsf{Halucination}, \mathsf{MemoryLoss}\}, \mathsf{Dementia})$
$\emptyset$	1	$(\{\mathsf{Dementia}\}, \mathsf{Observation})$

P_{exa}	$\mathcal{O}_{\mathsf{exa}}$	P_{exa}	$\mathcal{O}_{\mathsf{exa}}$
0.5	$\mathsf{Dementia}(p1)$	0.8	$\mathsf{Pain}(p1)$
0.4	$\mathsf{Halucination}(p2)$	1	$\mathsf{Pain}(p2)$
0.9	$\mathsf{Nausea}(p3)$	0.7	$\mathsf{MemoryLoss}(p3)$

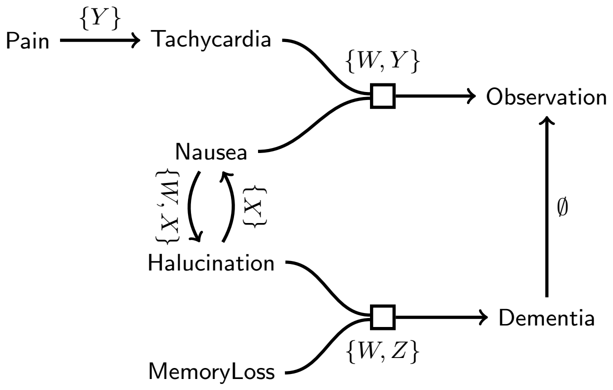

Fig. 3. The hypergraph portion of a probabilistic $\mathcal{HL}$ ontology. Each hyperedge is labelled with the set of RVs from the BN $\mathcal{B}_{\mathsf{exa}}$ given by the context function.

edge $(\{\mathsf{Dementia}\}, \mathsf{Observation})$. It is associated to the empty context $\emptyset$, which expresses that it must always hold; all the RVs in $\emptyset$ are always true. Hence, it is also assigned the probability 1. In the lower part of the table we observe a few facts expressing findings present in three different patients. We assume that these finding were obtained or confirmed through imprecise indirect tests and are thus associated to a likelihood. For an easier reading, the hypergraph portion of this ontology, together with its context mapping, is depicted in Figure 3. If we ignore for a second the probabilities, from the ontology $\mathcal{O}_{\mathsf{exa}}$ we can derive that all three patients are under observation (i.e., $\mathsf{Observation}(p_i)$ holds for $i = 1, 2, 3$). For instance, p_3 presents nausea, which implies halucinations. The latter, together with memory loss implies dementia, which itself implies observation. What we are now interested in now is taking also the probabilistic knowledge into account.

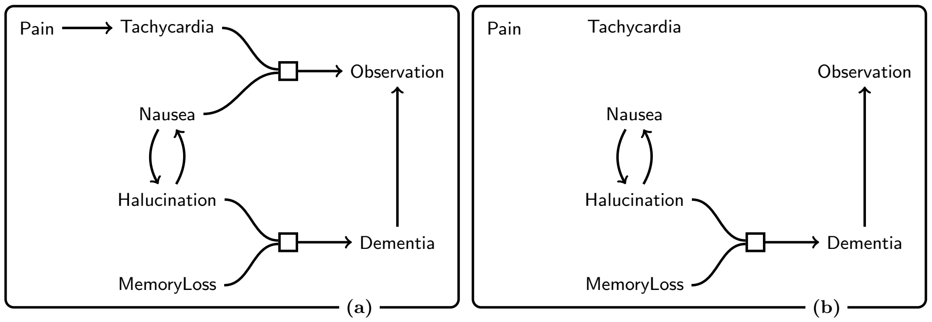

Fig. 4. The hypergraphs from the classical $\mathcal{HL}$ ontologies defined by the possible worlds (a) $w_1 = (\mathsf{Fact}(\mathcal{O}_{\mathsf{exa}}), \{W, X, Y, Z\})$ and (b) $w_2 = (\mathsf{Fact}(\mathcal{O}_{\mathsf{exa}}), \{W, X, Z\})$.

As mentioned already, we use a possible-worlds semantics to interpret the probabilities. Specifically, let $\mathsf{Fact}(\mathcal{O}) \subseteq \mathcal{O}$ be the set of all facts in $\mathcal{O}$, and let V the set of nodes in the BN $\mathcal{B}$. The set of *possible worlds* is $\mathscr{P}(\mathsf{Fact}(\mathcal{O})) \times \mathscr{P}(V)$; i.e., a pair containing a set of facts and a set of nodes. Each possible world $w = (\mathcal{F}, \mathbf{X})$ is assigned a probability defined as $P(w) := \prod_{\alpha \in \mathcal{F}} P(\alpha) \cdot P_{\mathcal{B}}(\mathbf{X})$. Note that for assigning the probability of a possible world, we are considering all facts as mutually independent, and independent of the hypergraph; however, the axioms in the hypergraph are not independent, but their relationship is expressed through the BN $\mathcal{B}$. In our example, these assumptions are meaningful: the probabilities in facts are obtained through tests which are assumed to be independent, but the hyperedges are probabilistically dependent according to the relationships expressed in $\mathcal{B}_{\mathsf{exa}}$.

Each possible world defines a (classical) $\mathcal{HL}$ ontology in the obvious manner. Specifically, the possible world $w = (\mathcal{F}, \mathbf{X})$ defines the $\mathcal{HL}$ ontology

$$\mathcal{O}(w) := \mathcal{F} \cup \{\alpha \in \mathcal{O} \setminus \mathsf{Fact}(\mathcal{O}) \mid B(\alpha) \subseteq \mathbf{X}\}.$$

In words, $\mathcal{O}(w)$ contains all the facts expressed by w and all the hyperedges that comply with the set $\mathbf{X}$ in their context mapping. Entailment in each ontology $\mathcal{O}(w)$ is defined exactly as in the classical case. For example, two possible worlds for our example probabilistic $\mathcal{HL}$ ontology are $w_1 = (\mathsf{Fact}(\mathcal{O}_{\mathsf{exa}}), \{W, X, Y, Z\})$ and $w_2 = (\mathsf{Fact}(\mathcal{O}_{\mathsf{exa}}), \{W, X, Z\})$. These possible worlds define the classical $\mathcal{HL}$ ontologies depicted in Figure 4, where the facts are all those from Table 2. In the ontology defined by w_2 we can no longer derive the fact $\mathsf{Observation}(p_2)$. Perhaps surprisingly at first sight, the probability of both possible worlds is 0. In fact, a simple observation of the BN in Figure 1 shows that any world which contains Z and either X or Y must have probability 0.

Recall that our semantics allows us to express a logical dependency between hyperedges: if α, β are two hyperedges such that $B(\alpha) \subseteq B(\beta)$, then for every possible world w it holds that if $\beta \in \mathcal{O}(w)$ then also $\alpha \in \mathcal{O}(w)$. This property is useful when trying to unify knowledge from different sources, giving a related probability to axioms provided from the same or related sources. In technical terms, it is also useful for maintaining the same syntax for $\mathcal{HL}$ axioms. Recall that hyperedges have only one target node, but it is not unrealistic to want to express that elements with some properties belong to the conjunction of two classes; e.g., that a father is a male parent. In classical $\mathcal{HL}$ this is normalised by introducing two separate axioms: fathers are male $((\{\mathsf{Father}\}, \mathsf{Male}))$, and fathers are parents $((\{\mathsf{Father}\}, \mathsf{Parent}))$, but in the probabilistic variant this normalisation would be incorrect unless we can express that both axioms must co-appear at all times.

Given a consequence $c \in \mathfrak{C}$, we define its *probability* w.r.t. $(\mathcal{O}, P, \mathcal{B}, B)$ as

$$P(c) := \sum_{w \in \mathscr{P}(\mathsf{Fact}(\mathcal{O})) \times \mathscr{P}(V), \mathcal{O}(w) \models c} P(w).$$

That is, we sum the probability of all possible worlds which entail the consequence c. Importantly, this semantics follows an open-world view of the possible world semantics: if a hyperedge or a fact is implicitly entailed by other axioms in the ontology, it might hold also in worlds where it is not explicitly required. This means that in general the probability assigned by the function P is only a lower bound, which may increase due to other dependencies. In our running example, we can see that $P(\mathsf{Observation}(p_1)) = 0.5$ because $(\{\mathsf{Dementia}\}, \mathsf{Observation})$ holds in all possible worlds, but $\mathsf{Dementia}(p_1)$ only in worlds with probability 0.5. Similarly, $P(\mathsf{Observation}(p_3)) = 0$ because for nausea and memory loss to cause dementia, both X and Z should hold which, as explained before, can only happen with probability 0. We leave as an exercise to the reader the computation of $P(\mathsf{Observation}(p_2))$

In the construction of $\mathcal{HL}$ and its semantics, we have made several design choices. All these design choices limit the applicability of our logic to some specific scenarios, and follow the guidelines from Section 5. Before continuing the technical details of this language, we expand on these choices and argue about its application scenario. First of all, we repeat that we assume that the hypergraph is proportionally much smaller than the data. Hence, to allow effective reasoning, we need to be more careful about the resources involved in handling the data, than in those handling the hypergraph (reachability) task. For that reason, we have chosen to use tuple-independence within the data, which will allow us to handle the relationships between data efficiently. Although this assumption is not justified in all possible datasets, there exist situations like, e.g., when facts are being produced by sensors in a complex machinery or when individuals are being sampled, where the independence assumption is justified. On the other hand, a full independence assumption between hyperedges is particularly excessive due to the limited expressivity of the language. Indeed, inexpressive languages usually require more than one axiom to encode a single constraint. For example, suppose that we do not only want to diagnose, but also describe and explain diseases in our ontology. Then, following the example from the beginning of Section 3, we would like to express that a common cold usually presents a running nose and light-headedness; that is, an axiom which would look like $(\mathsf{Cold}, \{\mathsf{RunNose}, \mathsf{LightHead}\})$. However, this is not expressible in $\mathcal{HL}$. Our only choice is to divide the knowledge into two axioms

$(\{\mathsf{Cold}\}, \mathsf{RunNose}), (\{\mathsf{Cold}\}, \mathsf{LightHead})$ but, clearly, these two axioms cannot be considered independent; in fact, they should always appear together because they form an atomic piece of knowledge. In our setting, both axioms will share a common context which guarantees this logical dependency. Finally, we have chosen to interpret the probabilities following the multiple-world semantics. This means that we are considering hyperedges as absolute, if they hold, but we are just not certain about the truth of the edge. This case commonly arises if the edges are being mined or extracted through automated methods from e.g., text or other web resources. In our case, we use the contextual interpretation commonly followed by Bayesian ontology languages. In this setting, we see a valuation of the variables in the BN as an uncertain context, and associate certain axioms to each context. That is, our knowledge is fully certain given the context in which it is applied, but the context itself may be uncertain. For example, we may know that in the context of osteoporosis, the bone mineral density is low, but we may not know (before a precise diagnosis) whether a patient has osteoporosis or not.

To compute the probability of a consequence c, one could simply follow the structure hinted by the definition: compute all possible worlds $w = (\mathcal{F}, \mathbf{X})$, and for each of them, compute $P(w)$ (linear in $\mathcal{F}$ but PP in the size of $\mathbf{X}$) and decide whether $\mathcal{O}(w) \models c$. This of course requires exponential time in the size of each of the components used, but by analysing each possible world independently, it is possible to limit the space usage to a polynomial bound. Although this algorithm is simple to understand and implement, it is without doubt also far from optimal in terms of the computational resources used. In order to find a better approach, we need to study deeper the properties of the language.

A slightly better approach is the following. Rather than trying to construct a classical ontology for every possible world, we produce a probabilistic ontology for each *Bayesian world*; that is, for every $\mathbf{X} \subseteq V$, but such that this ontology preserves adequate properties for probabilistic reasoning. Specifically, for each $\mathbf{X} \subseteq V$, let $\mathcal{O}(\mathbf{X}) := (\mathcal{O}', P')$ be the probabilistic ontology where

$$\mathcal{O}' := \mathsf{Fact}(\mathcal{O}) \cup \{\alpha \in \mathcal{O} \setminus \mathsf{Fact}(\mathcal{O}) \mid B(\alpha) \subseteq \mathbf{X}\}$$

and P' is the restriction of P to $\mathsf{Fact}(\mathcal{O})$. In a nutshell, $\mathcal{O}(\mathbf{X})$ includes the hypergraph that is compatible with the choice $\mathbf{X}$, but preserves the probabilistic facts as they are. Note that according to our semantics, these probabilistic facts are a probabilistic database with tuple independence. The hypergraph in $\mathcal{O}(\mathbf{X})$ still entails some implicit facts which are not explicitly encoded in the database, and need to be handled. Given a consequence c, we can compute its probability $P_{\mathbf{X}}(c)$ w.r.t. $\mathcal{O}(\mathbf{X})$ by asking a simple UCQ over the probabilistic database $(\mathsf{Fact}(\mathcal{O}), P')$, where the information of the hypergraph is encoded through a backwards propagation. For example, if $\mathbf{X} = \{W, X, Y, Z\}$ (see Figure 4 (a)) and we are interested in the consequence $\mathsf{Observation}(p_3)$, we can construct the union of conjunctive queries by traversing the hyperedges backwards, as shown in Table 3. The first four lines (before the separation) are those from the UCQ built by $\mathbf{X} = \{W, X, Z\}$ (Figure 4 (b)). For more details see [1, 15]. Note that all the components of this UCQ are simple queries with only one constant, and are thus

Table 3. Construction of a UCQ through backward traversing of a hypergraph.

Observation(p_3)
Dementia(p_3)
Halucination(p_3) $\wedge$ MemoryLoss(p_3)
Nausea(p_3) $\wedge$ MemoryLoss(p_3)
Tachycardia(p_3) $\wedge$ Nausea(p_3)
Tachycardia(p_3) $\wedge$ Halucination(p_3)
Pain(p_3) $\wedge$ Nausea(p_3)
Pain(p_3) $\wedge$ Halucination(p_3)

easily manageable by modern probabilistic database systems [26]. To compute the probability of the consequence w.r.t. the original probabilistic ontology, we need to accumulate over all Bayesian worlds, taking into account their relative likelihood by means of an arithmetic sum. Indeed, it can be shown that for every consequence $c \in \mathfrak{C}$, $P(c) = \sum_{\mathbf{X} \subseteq V} P_{\mathcal{B}}(\mathbf{X}) \cdot P_{\mathbf{X}}(c)$.

Note that this second approach would still require exponential time on the size of the probabilistic ontology; at the very least, it needs to enumerate all Bayesian worlds (i.e., all subsets of V). However, this blowup is restricted to the elements of the Bayesian network, which is assumed to be the smallest component of the whole ontology. At each Bayesian world, the probability of the consequence can be computed in polynomial time. Thus, the overall complexity of computing the probability is bounded by a polynomial on the size of the database. This is called the *data complexity.* Moreover, using a probabilistic database system allows us to make slightly more advanced derivations. One simple example is that we could substitute the constant p_3 with a variable x in the UCQ generated by our model. This would return *all* individuals that are under observation (with their respective probability), without having to query them explicitly.

7 Conclusions

In this tutorial we have introduced the basic schema of probabilistic ontologies. Our goal was not to consider a specific (probabilistic) ontology language, but rather to allow for a general view which includes as many of the existing ontology languages as possible. As a side benefit, our formalisation includes languages which may not be typically considered as ontology languages, but which can still be analysed under this view.

Just as there exists a plethora of ontology languages, which have been developed to satisfy specific needs, there are also many different possibilities for extending these languages to deal with probabilities. To get a feeling about the variety of the existing languages, one can consider the survey by Lukasiewicz and Straccia [48], which is already over a decade old. If one is interested in developing—or using—a probabilistic ontology, it is important to understand the design choices behind the underlying language and their motivations, and to ensure that these coincide with the task at hand. As a mnemonic device for the main notions to consider, we have developed *The Five Golden Rules*, which are presented in Section 5. The intention of these rules is to remain informed about the properties of the formalism, and the kind of consequences one should expect from them. They are, however, far from complete; a detailed analysis of each formalism, which also takes into account the properties of the underlying ontology language is always necessary. Still, if there is only one message that the reader takes home from this work, is that **there is no one-size-fits-all solution**. Each variation has its advantages and disadvantages, and they must be balanced for the applicatoin at hand. But, importantly, all the factors need to be taken into account. We often tend to focus our attention too much on the computational complexity of a problem, which in this case means ignoring important information from our knowledge domain.

As a case scenario, we have looked in some detail into a probabilistic extension of the ontology language $\mathcal{HL}$ which combines hypergraphs with a database. When building the probabilistic extension, we considered the choices to be made, and finished with the multiple-world semantics under an open-world view. We also combined an independence assumption (for all the facts in the database) with a more complex joint probability distribution on the hyperedges, which is capable of expressing probabilistic dependencies (encoded in a Bayesian network) and logical dependencies, through a mapping from the axioms to the BN. Again, we emphasise that this language is not intended to be better (or worse) than other proposals in the literature, but rather showcase the development of a probabilistic extension. Although it is possible to make blanket statements about all probabilistic extensions with some choices—e.g., that w.r.t. subjective probabilities under an open-world view, the probability of a consequence can be reduced to multiple classical entailments—being able to look at the specific formalism will usually lead to better results. In our example language, we were able to improve the overall complexity simply by knowing how the reasoning techniques work in the classical case, and adapting them for the probabilistic scenario. In general, the analysis is not so straightforward, but it is worth the effort. The example also shows that often probabilistic ontologies require more information than just a probability assignment to axioms.

In our effort to provide a general view to probabilistic ontologies without restricting to specific languages, but at the same time trying to keep the presentation simple, we had to exclude some prominent formalisms. Some of them (for example, those described at the beginning of Section 4) could be included without major conceptual changes, but we decided against it mainly to avoid saturating the notation, and confusing the reader with elements which only become relevant in specific cases. However, there are other languages which could not be fitted for more fundamental reasons. For example, non-monotonic logics [13], where new knowledge may defeat previously derived consequences, do not fit into our definition of ontology language. They could be problematic for the way we describe the semantics of probabilistic entailments.

Probabilities are a very natural way to represent uncertainty. There are perhaps two main causes for this naturality: first, the daily bombardment of probabilistic language to which we are submitted; and the fact that we have a solid theoretical background for obtaining and reasoning about probabilities in many different contexts. But this familiarity can also be dangerous. Typical users who are not probability experts have a faulty intuition of the meaning of probabilities, and in particular do not know the subtleties of the results of statistical methods. Case in point, the use of a sampling result as a fixed probability, ignoring the underlying sampling error, and the variability of the studied phenomenon. We thus encourage all readers to go forward, use probabilities, use them profusely, but use them carefully.

References

1. Artale, A., Calvanese, D., Kontchakov, R., Zakharyaschev, M.: The DL-Lite family and relations. Journal of Artificial Intelligence Research **36**, 1–69 (2009). https://doi.org/10.1613/jair.2820
2. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press, second edn. (2007)
3. Baader, F., Hladik, J., Peñaloza, R.: Automata can show pspace results for description logics. Information and Computation **206**(9-10), 1045–1056 (2008). https://doi.org/10.1016/j.ic.2008.03.006
4. Bellomarini, L., Benedetto, D., Gottlob, G., Sallinger, E.: Vadalog: A modern architecture for automated reasoning with large knowledge graphs. Information Systems p. 101528 (2020). https://doi.org/https://doi.org/10.1016/j.is.2020.101528
5. Bianchi, F., Rossiello, G., Costabello, L., Palmonari, M., Minervini, P.: Knowledge graph embeddings and explainable AI. CoRR **abs/2004.14843** (2020), https://arxiv.org/abs/2004.14843
6. Biere, A., Heule, M.J.H., van Maaren, H., Walsh, T. (eds.): Handbook of Satisfiability, Frontiers in Artificial Intelligence and Applications, vol. 185. IOS Press (2009)
7. Bobillo, F., Straccia, U.: Reasoning within fuzzy OWL 2 EL revisited. Fuzzy Sets and Systems **351**, 1–40 (2018). https://doi.org/10.1016/j.fss.2018.03.011
8. Borgwardt, S.: Fuzzy description logics with general concept inclusions. Ph.D. thesis, Technische Universität Dresden, Germany (2014)
9. Borgwardt, S., Ceylan, İ.İ., Lukasiewicz, T.: Ontology-mediated query answering over log-linear probabilistic data. In: Proceedings of The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI'19). pp. 2711–2718. AAAI Press (2019). https://doi.org/10.1609/aaai.v33i01.33012711
10. Borgwardt, S., Peñaloza, R.: The complexity of lattice-based fuzzy description logics. Journal on Data Semantics **2**(1), 1–19 (2013). https://doi.org/10.1007/s13740-012-0013-x
11. Borgwardt, S., Peñaloza, R.: Fuzzy description logics - A survey. In: Moral, S., Pivert, O., Sánchez, D., Marín, N. (eds.) Proceedings of the 11th International Conference on Scalable Uncertainty Management (SUM 2017). LNCS, vol. 10564, pp. 31–45. Springer (2017). https://doi.org/10.1007/978-3-319-67582-4_3

12. Botha, L., Meyer, T., Peñaloza, R.: The bayesian description logic BALC. In: Proceedings of the 16th European Conference on Logics in Artificial Intelligence (JELIA'19) (2019), to appear
13. Brewka, G.: Nonmonotonic reasoning - logical foundations of commonsense, Cambridge tracts in theoretical computer science, vol. 12. Cambridge University Press (1991)
14. Calì, A., Gottlob, G., Lukasiewicz, T., Pieris, A.: Datalog+/-: A family of languages for ontology querying. In: de Moor, O., Gottlob, G., Furche, T., Sellers, A.J. (eds.) Datalog Reloaded - 1st International Workshop, Datalog 2010, Oxford, UK, March 16-19, 2010. Revised Selected Papers. Lecture Notes in Computer Science, vol. 6702, pp. 351–368. Springer (2011), http://dx.doi.org/10.1007/978-3-642-24206-9_20
15. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: Data complexity of query answering in description logics. Artificial Intelligence **195**, 335–360 (2013). https://doi.org/10.1016/j.artint.2012.10.003
16. Cano, A., Cozman, F.G., Lukasiewicz, T.: Reasoning with imprecise probabilities. International Journal of Approximate Reasoning **44**(3), 197–199 (2007). https://doi.org/10.1016/j.ijar.2006.09.001
17. Carvalho, R.N., Laskey, K.B., Costa, P.C.G.: PR-OWL - a language for defining probabilistic ontologies. International Journal of Approximate Reasoning **91**, 56–79 (2017). https://doi.org/10.1016/j.ijar.2017.08.011
18. Ceylan, İ.İ., Darwiche, A., den Broeck, G.V.: Open-world probabilistic databases. In: Baral, C., Delgrande, J.P., Wolter, F. (eds.) Proceedings of the Fifteenth International Conference on Principles of Knowledge Representation and Reasoning (KR 2016). pp. 339–348. AAAI Press (2016), http://www.aaai.org/ocs/index.php/KR/KR16/paper/view/12908
19. Ceylan, İ.İ., Lukasiewicz, T.: A tutorial on query answering and reasoning over probabilistic knowledge bases. In: d'Amato, C., Theobald, M. (eds.) Reasoning Web. Learning, Uncertainty, Streaming, and Scalability - 14th International Summer School 2018 Tutorial Lectures. LNCS, vol. 11078, pp. 35–77. Springer (2018). https://doi.org/10.1007/978-3-030-00338-8_3
20. Ceylan, İ.İ., Peñaloza, R.: The bayesian ontology language BEL. Journal of Automated Reasoning **58**(1), 67–95 (2017). https://doi.org/10.1007/s10817-016-9386-0
21. Concannon, L.G., Kaufman, M.S., Herring, S.A.: Counseling athletes on the risk of chronic traumatic encephalopathy. Sports Health **6**(5), 396–401 (2014). https://doi.org/10.1177/1941738114530958
22. Cooper, G.F.: The computational complexity of probabilistic inference using bayesian belief networks. Artificial Intelligence **42**(2-3), 393–405 (1990). https://doi.org/10.1016/0004-3702(90)90060-D
23. Cozman, F.G.: Graphical models for imprecise probabilities. International Journal of Approximate Reasoning **39**(2-3), 167–184 (2005). https://doi.org/10.1016/j.ijar.2004.10.003
24. Cuenca Grau, B., Horrocks, I., Motik, B., Parsia, B., Patel-Schneider, P., Sattler, U.: OWL 2: The next step for OWL. J. of Web Semantics **6**, 309–322 (2008)
25. Cyganiak, R., Wood, D., Lanthaler, M. (eds.): RDF 1.1 Concepts and Abstract Syntax. W3C Recommendation (25 February 2014), available at http://www.w3.org/TR/rdf11-concepts/
26. Dalvi, N.N., Suciu, D.: The dichotomy of probabilistic inference for unions of conjunctive queries. Journal of the ACM **59**(6), 30:1–30:87 (2012). https://doi.org/10.1145/2395116.2395119

27. Darwiche, A.: Modeling and Reasoning with Bayesian Networks. Cambridge University Press (2009), http://www.cambridge.org/uk/catalogue/catalogue.asp?isbn=9780521884389
28. Dechter, R.: Bucket elimination: A unifying framework for reasoning. Artificial Intelligence **113**(1-2), 41–85 (1999). https://doi.org/10.1016/S0004-3702(99)00059-4
29. Diestel, R.: Graph Theory, Graduate texts in mathematics, vol. 173. Springer, 5th edn. (2017)
30. Dubois, D., Prade, H.: Possibility Theory - An Approach to Computerized Processing of Uncertainty. Springer (1988). https://doi.org/10.1007/978-1-4684-5287-7
31. Gallo, G., Longo, G., Pallottino, S., Nguyen, S.: Directed hypergraphs and applications. Discrete Applied Mathematics **42**(2), 177 – 201 (1993). https://doi.org/https://doi.org/10.1016/0166-218X(93)90045-P, http://www.sciencedirect.com/science/article/pii/0166218X9390045P
32. Gottwald, S.: Many-valued and fuzzy logics. In: Kacprzyk, J., Pedrycz, W. (eds.) Springer Handbook of Computational Intelligence, pp. 7–29. Springer Handbooks, Springer (2015). https://doi.org/10.1007/978-3-662-43505-2_2
33. Gutiérrez-Basulto, V., Jung, J.C., Lutz, C., Schröder, L.: A closer look at the probabilistic description logic prob-el. In: Burgard, W., Roth, D. (eds.) Proceedings of The Twenty-Fifth AAAI Conference on Artificial Intelligence (AAAI'11). AAAI Press (2011), http://www.aaai.org/ocs/index.php/AAAI/AAAI11/paper/view/3702
34. Gutiérrez-Basulto, V., Jung, J.C., Lutz, C., Schröder, L.: Probabilistic description logics for subjective uncertainty. Journal of Artificial Intelligence Research **58**, 1–66 (2017). https://doi.org/10.1613/jair.5222
35. Hájek, P.: Metamathematics of Fuzzy Logic, Trends in Logic, vol. 4. Kluwer (1998). https://doi.org/10.1007/978-94-011-5300-3
36. Halpern, J.Y.: An analysis of first-order logics of probability. In: Sridharan, N.S. (ed.) Proceedings of the 11th International Joint Conference on Artificial Intelligence (IJCAI'89). pp. 1375–1381. Morgan Kaufmann (1989), http://ijcai.org/Proceedings/89-2/Papers/084.pdf
37. Halpern, J.Y.: An analysis of first-order logics of probability. Artificial Intelligence **46**(3), 311–350 (1990). https://doi.org/10.1016/0004-3702(90)90019-V
38. Halpern, J.Y.: Reasoning about uncertainty. MIT Press (2005)
39. Hogan, A., Blomqvist, E., Cochez, M., d'Amato, C., de Melo, G., Gutierrez, C., Gayo, J.E.L., Kirrane, S., Neumaier, S., Polleres, A., Navigli, R., Ngomo, A.N., Rashid, S.M., Rula, A., Schmelzeisen, L., Sequeda, J.F., Staab, S., Zimmermann, A.: Knowledge graphs. CoRR **abs/2003.02320** (2020), https://arxiv.org/abs/2003.02320
40. Horrocks, I., Kutz, O., Sattler, U.: The even more irresistible $\mathcal{SROIQ}$. In: Doherty, P., Mylopoulos, J., Welty, C.A. (eds.) Proc. 10th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'06). pp. 57–67. AAAI Press (2006)
41. Jaeger, M.: Probabilistic reasoning in terminological logics. In: Doyle, J., Sandewall, E., Torasso, P. (eds.) Proceedings of the 4th International Conference on Principles of Knowledge Representation and Reasoning (KR 1994). pp. 305–316. Morgan Kaufmann (1994)
42. Klinov, P., Parsia, B.: Pronto: A practical probabilistic description logic reasoner. In: Bobillo, F., da Costa, P.C.G., d'Amato, C., Fanizzi, N., Laskey, K.B., Laskey, K.J., Lukasiewicz, T., Nickles, M., Pool, M. (eds.) Uncertainty Reasoning for the Semantic Web II, International Workshops URSW 2008-2010, Revised Selected Papers. LNCS, vol. 7123, pp. 59–79. Springer (2013). https://doi.org/10.1007/978-3-642-35975-0_4

43. Koller, D., Levy, A.Y., Pfeffer, A.: P-CLASSIC: A tractable probablistic description logic. In: Kuipers, B., Webber, B.L. (eds.) Proceedings of The Fourteenth National Conference on Artificial Intelligence (AAAI'97). pp. 390–397. AAAI Press (1997), http://www.aaai.org/Library/AAAI/1997/aaai97-060.php
44. Kyburg, Jr, H.E., Teng, C.M.: Uncertain Inference. Cambridge University Press (2001). https://doi.org/10.1017/CBO9780511612947
45. Laskey, K.B., da Costa, P.C.G.: Of starships and klingons: Bayesian logic for the 23rd century. In: Proceedings of the 21st Conference in Uncertainty in Artificial Intelligence (UAI'05). pp. 346–353. AUAI Press (2005)
46. Littman, M.L., Majercik, S.M., Pitassi, T.: Stochastic boolean satisfiability. Journal of Automated Reasoning **27**(3), 251–296 (2001). https://doi.org/10.1023/A:1017584715408
47. Lukasiewicz, T.: Expressive probabilistic description logics. Artificial Intelligence **172**(6-7), 852–883 (2008). https://doi.org/10.1016/j.artint.2007.10.017
48. Lukasiewicz, T., Straccia, U.: Managing uncertainty and vagueness in description logics for the semantic web. Journal of Web Semantics **6**(4), 291–308 (2008)
49. Lutz, C., Schröder, L.: Probabilistic description logics for subjective uncertainty. In: Lin, F., Sattler, U., Truszczynski, M. (eds.) Proceedings of the Twelfth International Conference on Principles of Knowledge Representation and Reasoning (KR 2010). AAAI Press (2010), http://aaai.org/ocs/index.php/KR/KR2010/paper/view/1243
50. Maggi, F.M., Montali, M., Peñaloza, R.: Temporal logics over finite traces with uncertainty. In: Proceedings of The Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI'20) (2020), to appear.
51. Maggi, F.M., Montali, M., Peñaloza, R., Alman, A.: Probabilistic business constraints. In: Proceedings of BPM-20 (2020), to appear
52. Maggi, F.M., Montali, M., Westergaard, M., van der Aalst, W.M.P.: Monitoring business constraints with linear temporal logic: An approach based on colored automata. In: Rinderle-Ma, S., Toumani, F., Wolf, K. (eds.) Proceedings of the 9th International Conference on Business Process Management)BPM 2011). LNCS, vol. 6896, pp. 132–147. Springer (2011). https://doi.org/10.1007/978-3-642-23059-2_13
53. Mintz, M., Bills, S., Snow, R., Jurafsky, D.: Distant supervision for relation extraction without labeled data. In: Su, K., Su, J., Wiebe, J. (eds.) ,Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics (ACL 2009). pp. 1003–1011. The Association for Computer Linguistics (2009), https://www.aclweb.org/anthology/P09-1113/
54. Motik, B., Cuenca Grau, B., Horrocks, I., Wu, Z., Fokoue, A., Lutz, C. (eds.): OWL 2 Web Ontology Language: Profiles. W3C Recommendation (27 October 2009), available at http://www.w3.org/TR/owl2-profiles/
55. Niepert, M., Noessner, J., Stuckenschmidt, H.: Log-linear description logics. In: Walsh, T. (ed.) Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI'11). pp. 2153–2158. IJCAI/AAAI (2011). https://doi.org/10.5591/978-1-57735-516-8/IJCAI11-359
56. Nilsson, N.J.: Probabilistic logic. Artificial Intelligence **28**(1), 71–87 (1986). https://doi.org/10.1016/0004-3702(86)90031-7
57. Pearl, J.: Probabilistic reasoning in intelligent systems - networks of plausible inference. Morgan Kaufmann series in representation and reasoning, Morgan Kaufmann (1989)

58. Peñaloza, R.: Inconsistency-tolerant instance checking in tractable description logics. LNCS, vol. 10364, pp. 215–229. Springer (2017). https://doi.org/10.1007/978-3-319-61252-2_15
59. Peñaloza, R.: Making decisions with knowledge base repairs. In: Torra, V., Narukawa, Y., Pasi, G., Viviani, M. (eds.) Proceedings of the 16th International Conference on Modeling Decisions for Artificial Intelligence (MDAI 2019). LNCS, vol. 11676, pp. 259–271. Springer (2019). https://doi.org/10.1007/978-3-030-26773-5_23
60. Peñaloza, R.: Axiom pinpointing. CoRR **abs/2003.08298** (2020), https://arxiv.org/abs/2003.08298
61. Peñaloza, R., Sertkaya, B.: Understanding the complexity of axiom pinpointing in lightweight description logics. Artificial Intelligence **250**, 80–104 (2017). https://doi.org/10.1016/j.artint.2017.06.002
62. Pesic, M., Schonenberg, H., van der Aalst, W.M.P.: DECLARE: full support for loosely-structured processes. In: Proceedings of the 11th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2007). pp. 287–300. IEEE Computer Society (2007). https://doi.org/10.1109/EDOC.2007.14
63. Raedt, L.D., Kimmig, A., Toivonen, H.: Problog: A probabilistic prolog and its application in link discovery. In: Veloso, M.M. (ed.) Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI'07). pp. 2462–2467. IJCAI (2007), http://ijcai.org/Proceedings/07/Papers/396.pdf
64. Riguzzi, F., Bellodi, E., Lamma, E., Zese, R.: Probabilistic description logics under the distribution semantics. Semantic Web **6**(5), 477–501 (2015). https://doi.org/10.3233/SW-140154
65. Roth, D.: On the hardness of approximate reasoning. Artificial Intelligence **82**(1-2), 273–302 (1996). https://doi.org/10.1016/0004-3702(94)00092-1
66. Schild, K.: A correspondence theory for terminological logics: Preliminary report. In: Mylopoulos, J., Reiter, R. (eds.) Proc. 12th Int. Joint Conf. on Artificial Intelligence (IJCAI'91). pp. 466–471. Morgan Kaufmann (1991)
67. Straccia, U.: Fuzzy semantic web languages and beyond. In: Benferhat, S., Tabia, K., Ali, M. (eds.) Proceedings of the 30th International Conference on Industrial Engineering and Other Applications of Applied Intelligent Systems (IEA/AIE 2017), Part I. LNCS, vol. 10350, pp. 3–8. Springer (2017). https://doi.org/10.1007/978-3-319-60042-0_1
68. Suciu, D., Olteanu, D., Ré, C., Koch, C.: Probabilistic Databases. Synthesis Lectures on Data Management, Morgan & Claypool Publishers (2011). https://doi.org/10.2200/S00362ED1V01Y201105DTM016
69. Tijms, H.: Understanding Probability. Cambridge University Press, 3 edn. (2012). https://doi.org/10.1017/CBO9781139206990