In *Proceedings of the Fourteenth National Conference on Artificial Intelligence (AAAI-97)*, pages 390-397, Providence, Rhode Island, August 1997

P-CLASSIC: A tractable probabilistic description logic

Daphne Koller
Computer Science Department
Stanford University
Stanford, CA 94305-9010
koller@cs.stanford.edu

Alon Levy
AT&T Labs
600 Mountain Ave.
Murray Hill, NJ 07974
levy@research.att.com

Avi Pfeffer
Computer Science Department
Stanford University
Stanford, CA 94305-9010
avi@cs.stanford.edu

Abstract

Knowledge representation languages invariably reflect a trade-off between expressivity and tractability. Evidence suggests that the compromise chosen by description logics is a particularly successful one. However, description logic (as for all variants of first-order logic) is severely limited in its ability to express uncertainty. In this paper, we present P-CLASSIC, a probabilistic version of the description logic CLASSIC. In addition to terminological knowledge, the language utilizes *Bayesian networks* to express uncertainty about the basic properties of an individual, the number of fillers for its roles, and the properties of these fillers. We provide a semantics for P-CLASSIC and an effective inference procedure for *probabilistic subsumption*: computing the probability that a random individual in class C is also in class D. The effectiveness of the algorithm relies on independence assumptions and on our ability to execute *lifted* inference: reasoning about similar individuals as a group rather than as separate ground terms. We show that the complexity of the inference algorithm is the best that can be hoped for in a language that combines description logic with Bayesian networks. In particular, if we restrict to Bayesian networks that support polynomial time inference, the complexity of our inference procedure is also polynomial time.

1 Introduction

First-order logic has been the basis for most knowledge representation formalisms. Its basic units—individuals, their properties, and the relations between them—naturally capture the way in which people encode their knowledge. Unfortunately, it is severely limited in its ability to represent our uncertainty about the world: a fact can only be known to be true, known to be false, or neither. By contrast, most of our knowledge about the real world is not absolutely true, but only true with some degree of certainty. This limitation renders first-order logic inapplicable to a large range of real-world problems.

The fundamental step in addressing this problem was taken by Bacchus (1990) and Halpern (1990). They defined and analyzed ways in which probabilities can be added to first-order logic, and clarified the semantics of the resulting formalisms. Their work focused on probabilistic extensions of *full* first-order logic. As a consequence, these logics were shown to be highly undecidable (much more so even than first-order logic). Furthermore, they do not support a natural and compact specification of independence assumptions, which are crucial to getting nontrivial conclusions from a probabilistic knowledge base.

There has been a recent move towards integrating probabilities into less expressive subsets of first-order logic. By and large, knowledge representation formalisms based on subsets of first-order logic fall into two categories: rule-based languages, and object-centered formalisms (e.g., frame-based languages, description logics). So far, most work on probabilistic extensions has focused on augmenting rule-based languages (Goldman and Charniak 1990; Breese 1992; Poole 1993; Ngo *et al.* 1995). In this paper, we take a first step towards integrating probabilities with object-centered languages, by developing a probabilistic description logic. Our language provides the ability to describe *classes* of individuals, and to reason about the relationships between classes.

Description logics are subsets of first-order logic with equality that have been designed to model rich class hierarchies. Informally, in a description logic we begin with a set of *primitive* concepts (i.e., unary predicates) and roles (binary relations). For a given individual a, the individuals related to a by some role R are called the R-*fillers* of a. Description logic allows us to describe classes of individuals (*complex concepts*) based on their properties: the primitive concepts to which they belong, the number of R-fillers of an individual, and the properties of the fillers. Description logics support *subsumption queries*—whether one complex concept is always a subset of another, and *membership queries*—whether a particular individual is an instance of a given concept.

Several systems have been built based on description logics (e.g., CLASSIC (Brachman *et al.* 1991), LOOM (MacGregor 1988), and BACK (Petalson 1991)), and they have been used in several applications (e.g., (Wright *et al.* 1993)). In addition, several information integration systems (e.g., the Information Manifold (Levy *et al.* 1996) and SIMS (Arens *et al.* 1996)) use description logics to represent the information sources by specifying the class of individuals contained in the information source. For example, an individual might be one article in a bibliographic database. To retrieve the complete answer to a query, the system accesses each information source whose description overlaps with the description of the query.

One of the main limitations of description logics is that they can express very little about the overlap between two concepts. Given two concepts, we can infer that one subsumes the other, that they are disjoint, or that they may have a non-empty overlap. However, the degree of the overlap cannot be described or inferred. The need for such knowledge is clearly demonstrated in the information integration domain. Accessing *every* information source which potentially overlaps with our query may be prohibitively expensive. If we want the system to find a large fraction of the answers as soon as possible, it is very important to infer the degree of overlap between classes of individuals.

In this paper, we describe a language, P-CLASSIC, which is a probabilistic extension of the description logic CLASSIC. P-CLASSIC allows the specification of a probability distribution over the properties of individuals. As the basic representational tool, we use *Bayesian networks* (Pearl 1988). Bayesian networks allow a compact and natural representation of complex probability distributions by using independence assumptions. In this case, we use a Bayesian network whose random variables are the basic properties of individuals (the primitive concepts), the numbers of their fillers, and the properties of their fillers.

In general, of course, the domain consists of many different types of individuals. It is rarely the case that the same distribution will be appropriate for each of them. For example, the distribution over the properties of an individual is usually quite different from the distribution over the properties of its fillers. Therefore, the probabilistic component of a P-CLASSIC knowledge base includes a number of different *p-classes (probabilistic classes)*, each of which is a Bayesian network over basic properties, the number of R-fillers (for the different roles R), and the p-classes from which the role fillers are chosen. In addition to the probabilistic component, a P-CLASSIC knowledge base also contains a standard terminological component, describing complex concepts in terms of primitive ones. In this paper we do not consider knowledge bases with ground facts (i.e., Aboxes).

The semantics for P-CLASSIC is a simple extension to the standard semantics of CLASSIC. Following (Halpern 1990), we interpret a p-class as a probability distribution over the elements in the domain. Intuitively, this corresponds to the probability of "choosing" (encountering) this element in this p-class. By assuming that the p-class distribution is as described in the corresponding Bayesian network, and that fillers are chosen independently of each other (from the appropriate p-class), we can show that our p-classes uniquely determine the probability of any complex concept.

A P-CLASSIC knowledge base allows us to answer any *probabilistic subsumption query*: for two complex concepts C, D, what is the probability that C holds within the set of individuals in D. By contrast, standard subsumption can only tell us whether this number is 1 (D is subsumed by C), 0 (D is disjoint from C), or somewhere in between.

Of course, the fact that our representation uniquely determines this number does not necessarily imply that we can effectively compute it in practice. We show that the particular description logic and independence assumptions that we have made also enable us to develop an effective inference algorithm for P-CLASSIC. The algorithm follows the same general lines as the inference algorithm for standard CLASSIC, by representing concepts as a graph. However, in P-CLASSIC we replace the logical inference for comparing pieces of the graph by inference with Bayesian networks for computing the probability of parts of the graph. Furthermore, in contrast to other algorithms for reasoning in other first-order probabilistic formalisms, our algorithm implements a form of *lifted inference*—reasoning at the level of variables rather than at the level of ground terms. This is possible because our independence assumptions enable the algorithm to reuse computation for individuals that are essentially identical (e.g., different R-fillers of the same individual). We show that, in some sense, the complexity of this algorithm is the best that can be hoped for in a language that combines the expressive power of CLASSIC and Bayesian networks. In particular, if we restrict to polynomial time Bayesian networks (e.g., *polytrees* (Pearl 1988)) the complexity of our inference algorithm remains polynomial.

Several works (Shastri 1989; Jaeger 1994; Heinsohn 1991) have considered probabilistic extensions of description logics. There, the focus was on completing partial statistical information using default probabilistic procedures such as entropy maximization or cross-entropy minimization, or on deriving the minimal conclusions available from a small, incomplete set of probabilistic statements. By contrast, our approach follows the more recent tradition, established by Bayesian networks, of having the probabilistic knowledge base completely specify a probability distribution. The full specification approach has been shown to be both conceptually and computationally simpler. As we have discussed, similar computational benefits are obtained in our framework, which supports an inference algorithm which is significantly more efficient than those for the previous works on probabilistic terminological languages.

2 The P-CLASSIC Language

We first briefly review the variant of the CLASSIC description logic that underlies P-CLASSIC, and then describe the probabilistic component of P-CLASSIC. Finally, we describe how these components come together to form a P-CLASSIC knowledge base.

2.1 The Description Logic

The basic vocabulary of a description logic consists of *primitive* concepts (unary predicates) $\mathcal{A}$ and roles (binary relations) $\mathcal{R}$. The language uses a set of *constructors* to build *descriptions*, defining new classes of individuals called *complex concepts*.

The non-probabilistic component of the P-CLASSIC language is a variant of the CLASSIC description logic. Like CLASSIC, we also allow a set of *attributes* $\mathcal{Q}$ in addition to standard roles. Attributes are binary relations which are functional: each individual has exactly one filler for that attribute. In P-CLASSIC we add the restriction that the filler for an attribute be one of a finite prespecified set of individuals.

Complex descriptions in our language are built using the following grammar, where $A \in \mathcal{A}$ denotes a primitive concept, $R \in \mathcal{R}$ a role, $Q \in \mathcal{Q}$ denotes an attribute, and C and D represent concept descriptions:

$C, D \rightarrow A \mid$	(primitive concept)
$C \sqcap D \mid$	(conjunction)
$\neg A \mid$	(negation on primitive concepts)
$\forall R.C \mid$	(universal quantification)
$(\geq\ n\, R) \mid (\leq\ n\, R) \mid$	(number restrictions)
$(\mathsf{fills}\ Q\, a)$	(filler specification)

Readers familiar with CLASSIC will see that our language does not contain CLASSIC's same-as constructor, but does support negation on primitive concepts that is not allowed in CLASSIC. CLASSIC also allows the fills constructor to be applied to nonfunctional roles. It should be noted that allowing negation on primitive concepts does not change the expressive power of CLASSIC; it therefore follows from (Borgida and Patel-Schneider 1994) that subsumption in the language described above can be done in polynomial time.

A description logic knowledge base Δ includes a *terminology* Δ_T (the Tbox) and a set of ground atomic facts Δ_A (the Abox). In this paper, we do not consider Aboxes. In CLASSIC, a terminology contains two kinds of statements: *concept introductions*, describing the primitive concepts in the terminology, and *concept definitions*, specifying the defined concepts. In P-CLASSIC a terminology includes only the concept definitions (as we describe shortly), while concept introductions are given as part of the probabilistic component of a knowledge base. A concept definition is a sentence of the form $C := D$, where C is a name of a defined concept and D is a description. Each defined concept is defined by a single concept definition, and we assume that names of defined concepts do not appear in the descriptions.[1]

In our analysis, we use the *canonical form* of a description. A canonical form of a description is $\alpha \sqcap \beta_{R_1} \sqcap \ldots \sqcap \beta_{R_m}$, where α is a conjunction of primitive concepts and their negations and of filler specifications (with no concept or attribute appearing more than once), and β_R is of the form $(\geq\ m_R\, R) \sqcap (\leq\ n_R\, R) \sqcap (\forall\, R.C)$, where C is also in canonical form. Any description in our language can be converted to canonical form in linear time. The *depth* of a description is defined as follows. The depth of α is 0. The depth of a concept $\alpha \sqcap \beta_{R_1} \sqcap \ldots \sqcap \beta_{R_m}$ is 1+Max(depth(α), depth(β_{R_1}), $\ldots$, depth(β_{R_m})).

2.2 The Probabilistic Component of P-CLASSIC

The main motivation for P-CLASSIC is to be able to express the degree of overlap between concepts. A *probabilistic class* (p-class) P specifies a probability distribution over the properties of individuals, allowing us to define the extent of the overlap between them. From these numbers, and appropriate probabilistic independence assumptions, we are able to deduce the answers to arbitrary *probabilistic subsumption queries*: queries of the form "What is the probability that an object belongs to concept D given that it belongs to concept C?". We write such queries as $\Pr(D \mid C)$.

The probabilistic component of P-CLASSIC consists of a set $\mathcal{P}$ of p-classes. Intuitively, a p-class represents our probabilistic information relating to a certain class of individuals. Each p-class $P \in \mathcal{P}$ is represented using a *Bayesian network* (Pearl 1988) N_P. A Bayesian network is a DAG in which the nodes represent random variables. Each variable takes on a value in some predefined range. Each node in the network is associated with a *conditional probability table* (CPT), which defines the probability of each possible value of the node, given each combination of values for the node's parents in the DAG. The network structure encodes the independence assumption that only the parent values are relevant when making this choice. A Bayesian network defines a joint probability distribution over all combinations of values of the variables in the network. The probability of a particular assignment of values is the product over all the nodes in the network of the conditional probability of the value of that node given the values of its parents.

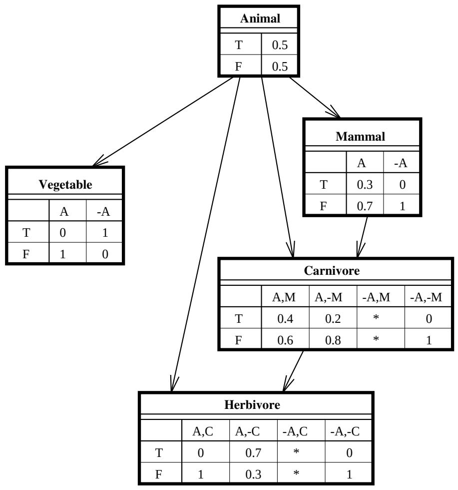

Figure 1: Part of the Bayesian network for the NATURAL THING p-class.

Figure 1 shows part of the Bayesian network for a p-class. This network contains a node for each of the primitive concepts ANIMAL, VEGETABLE, MAMMAL, CARNIVORE and HERBIVORE. The value of each node is either *true* or *false*, depending on whether an object belongs to the concept or not. The network defines a joint probability distribution over the set of truth assignments to these concepts. For example, consider the truth assignment (ANIMAL, $\neg$VEGETABLE, $\neg$MAMMAL, CARNIVORE, $\neg$HERBIVORE). Its probability is computed by the product

$$\begin{aligned}&\Pr(\text{ANIMAL}) \cdot \Pr(\neg\text{VEGETABLE} \mid \text{ANIMAL}) \cdot \\ &\Pr(\neg\text{MAMMAL} \mid \text{ANIMAL}) \cdot \Pr(\text{CARNIVORE} \mid \text{ANIMAL}, \neg\text{MAMMAL}) \cdot \\ &\Pr(\neg\text{HERBIVORE} \mid \text{ANIMAL}, \text{CARNIVORE}) \\ &\quad = 0.5 \cdot 1 \cdot 0.7 \cdot 0.2 \cdot 1 = 0.07.\end{aligned}$$

[1] This is not a restriction when the terminology has no cycles (as in CLASSIC) because the definitions in the terminology can be unfolded. However, as usual, unfolding a terminology may cause its size to grow exponentially.

In general, N_P will contain a node for each primitive concept $A \in \mathcal{A}$. For any description C which is a conjunction of primitive concepts and their negations, the probability $N_P(C)$ is defined by the network. This allows simple subsumption queries of the form $\Pr(D \mid C)$ to be answered when both C and D are conjunctions of primitive concepts and their negations.

The network for a p-class also determines the probability distribution over attribute values. For each attribute $Q \in \mathcal{Q}$, N_P contains a node FILLS(Q). The CPT for a node lists some finite set of objects $v_1, \ldots, v_k$, and for each v_i, the probability that (fills Q v_i) holds given each possible combination of values for the node's parents. Note that the node for an attribute also enumerates the set of values that the attribute may take, and therefore can be used instead of CLASSIC's one-of constructor for attributes.

To fully describe an individual, we also need to describe the number of its various fillers and their properties. For each role $R \in \mathcal{R}$, the network specifies the number of R-fillers by including a NUMBER(R) node, which takes on values between 0 and some upper bound b_R. The node's value denotes the number of R-fillers that the object has. To describe the properties of the fillers, we simply assign a p-class for each role, which specifies the distribution from which the fillers are chosen. Thus, for each role R, the network contains a PC(R) node, whose value ranges over the set $\mathcal{P}$ of p-classes. This node is always deterministic, i.e., for any combination of values of its parents, exactly one p-class is assigned probability 1, and all other p-classes are assigned probability 0. For simplicity of presentation, we place some restrictions on the topology of the network. We assume that a NUMBER(R) node may only be a parent of the corresponding PC(R) node. The PC(R) node may not be a parent of any other node.

The probabilistic component of a P-CLASSIC knowledge base recursively describes a distribution over the properties of an object. One of the p-classes, denoted P^*, is the *root* p-class, denoting the distribution over all objects. The properties of an object are chosen according to the Bayeisan network N_{P^*}. As we traverse the Bayesian network from its roots down, each node tells us the probability with which its value should be chosen. The root nodes are chosen with the appropriate unconditional probability, while the distribution used for other nodes is determined by prior choices for the values of their parents. In particular, the network dictates how the number of fillers for each role is chosen given the basic properties and attribute values. By specifying the p-class for the fillers, the network specifies how the properties of the fillers are chosen recursively using a similar process.

As stated earlier, CLASSIC allows us to state concept introductions in its terminology. In P-CLASSIC, concept introductions can be represented directly in the probabilistic component of the knowledge base (in fact, concept introductions are a special case of probabilistic assertions). A concept introduction is a sentence of the form $A \subseteq D$, where A is a name of a primitive concept and D is a concept description. For simplicity of exposition, we assume that D does not mention any of the roles. (In Section 5.2 we describe how to remove this restriction.) As in CLASSIC, we consider concept introductions that are acyclic, i.e., if $\varphi_1, \ldots, \varphi_n$ is a list of concept introductions, then the description in φ_i can only mention the concepts introduced in $\varphi_1, \ldots, \varphi_{i-1}$, or the concept THING (which denotes the set of all individuals). A concept introduction $A \subseteq D$ is encoded in the knowledge base by specifying in *each* p-class that the probability of $\neg D \sqcap A$ is 0. Since the concept introductions are acyclic, we can encode this information by making the concepts appearing in D parents of A, and setting the appropriate entries in the CPT for A to 0.

Figure 2 shows the probabilistic component of the knowledge-base for a domain of natural objects. There are three p-classes, each being a network containing the nodes ANIMAL, VEGETABLE, MAMMAL, CARNIVORE, HERBIVORE, FILLS(SIZE), NUMBER(EATS) and PC(EATS).

In the NATURAL THING p-class, $\Pr(\text{ANIMAL}) = 0.5$, while $\Pr(\text{VEGETABLE} \mid \text{ANIMAL}) = 0$, and $\Pr(\text{VEGETABLE} \mid \neg\text{ANIMAL}) = 1$. These assertions encode the terminological knowledge that everything is either an animal or a vegetable, and the two concepts are disjoint.[2] In the CPT for MAMMAL, we see that only animals can be mammals, and $\Pr(\text{MAMMAL} \mid \text{ANIMAL}) = 0.3$. Only animals can be carnivores, and mammals are more likely than other animals to be carnivorous; the entries in the column for $\neg$ANIMAL,MAMMAL are irrelevant since that combination is impossible. The FILLS(SIZE) node indicates that the value of the *size* attribute must be *big*, *medium*, or *small*, and the conditional probability of each value. Since vegetables don't eat, NUMBER(EATS) is always 0 if VEGETABLE is true, while for non-vegetables (i.e., animals) it is a number between 1 and 6 with the given distribution. Finally, PC(EATS) depends on CARNIVORE and HERBIVORE: for carnivores it is CARNIVORE FOOD, for herbivores it is HERBIVORE FOOD, while for things which are neither it is NATURAL THING. Since nothing is both a carnivore and a herbivore, that column is irrelevant.

The CARNIVORE FOOD p-class is the same as NATURAL THING conditioned on ANIMAL being true. Thus VEGETABLE is false, and the other nodes only contain columns that are consistent with these facts. HERBIVORE FOOD is the same as NATURAL THING conditioned on VEGETABLE being true. In this case the p-class is deterministic except for the value of FILLS(SIZE), since ANIMAL, MAMMAL, CARNIVORE and HERBIVORE are all false, and NUMBER(EATS) is 0. PC(EATS) is irrelevant since there are no *eats*-fillers.

3 Semantics of P-CLASSIC

The semantics of P-CLASSIC is an extension of the semantics of CLASSIC. The basic element is an *interpretation*. An interpretation I contains a non-empty domain $\mathcal{O}^I$. It assigns

[2] Strictly speaking, the probabilistic version of this statement is slightly weaker than the terminological version, because it is possible that the set of things that are both animal and vegetable is non-empty but of measure zero.

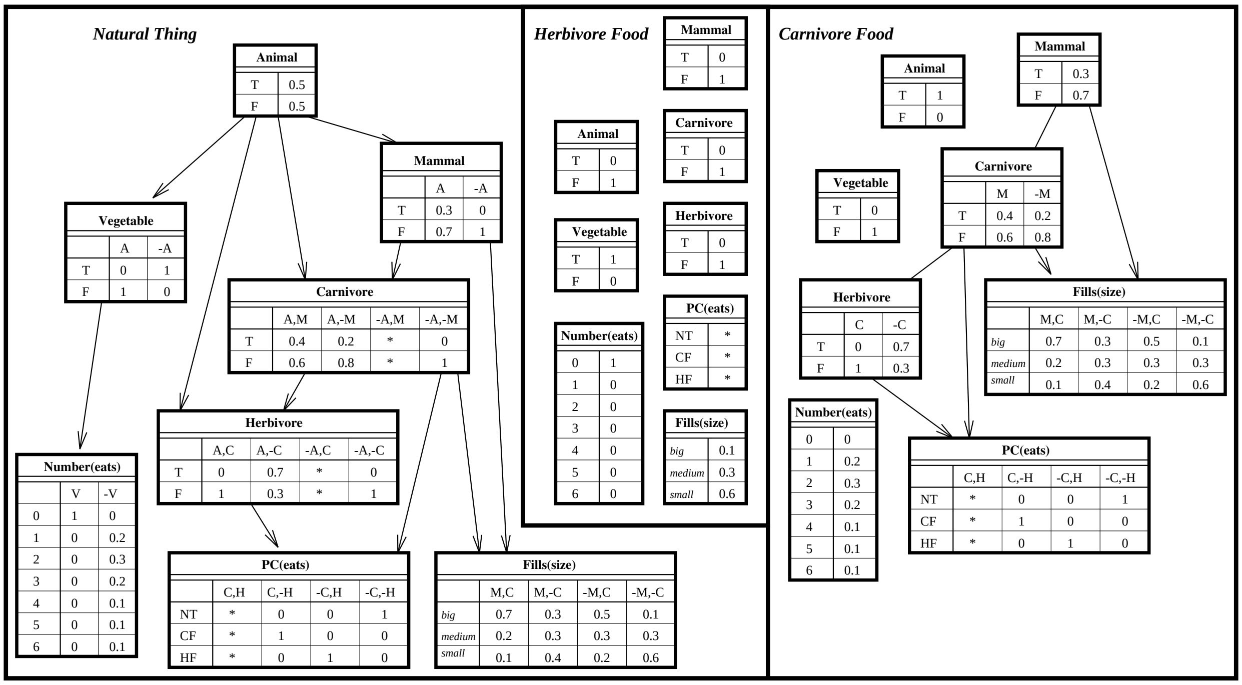

Figure 2: P-classes for the nature domain.

an element $a^I \in \mathcal{O}^I$ to every individual a, a unary relation A^I to every concept name $A \in \mathcal{A}$, a binary relation R^I over $\mathcal{O}^I \times \mathcal{O}^I$ to every role $R \in \mathcal{R}$, and a total function $Q^I : \mathcal{O} \to \mathcal{O}$ to every attribute $Q \in \mathcal{Q}$. The interpretations of the descriptions are defined recursively on their structure as follows ($\sharp\{S\}$ denotes the cardinality of a set S):

$(C \sqcap D)^I = C^I \cap D^I$, $(\neg A)^I = \mathcal{O}^I \setminus A^I$,
$(\forall R.C)^I = \{d \in \mathcal{O}^I \mid \forall e : (d,e) \in R^I \to e \in C^I\}$,
$(\geq n\ R)^I = \{d \in \mathcal{O}^I \mid \sharp\{e \mid (d,e) \in R^I\} \geq n\}$,
$(\leq n\ R)^I = \{d \in \mathcal{O}^I \mid \sharp\{e \mid (d,e) \in R^I\} \leq n\}$
$(\mathsf{fills}\ Q\ a)^I = \{d \in \mathcal{O}^I \mid Q^I(d) = a\}$.

An interpretation I is a model of a terminology Δ_T if $C^I = D^I$ for every concept definition $C := D$ in Δ_T. A concept C is said to be *subsumed* by a concept D w.r.t. a terminology Δ_T if $C^I \subseteq D^I$ for every model I of Δ_T.

In order to extend this semantics to P-CLASSIC, we have to provide an interpretation for the p-classes. Following (Halpern 1990), we interpret a p-class as an objective (statistical) probability.[3] That is, each p-class will be associated with a distribution *over the domain* $\mathcal{O}^I$. Intuitively, a p-class P describes a random event: the selection of an individual from the domain. The probability with which an individual is chosen depends on its properties (as determined by the other components of the interpretation). First, a truth assignment to the primitive concepts and an assignment of values to attributes is chosen, according to the probability distribution defined in the network N_P. Given this assignment, the number of fillers for each role is chosen, according to the conditional probability tables of the NUMBER nodes. Finally, the properties of the fillers are chosen, using another random event, as described by the p-class determined by the PC node for the appropriate role. Our semantics require that the probability of choosing an element be consistent with this type of random experiment for choosing its properties.

Definition 3.1: Let I be some interpretation over our vocabulary. A probability distribution μ over $\mathcal{O}^I$ is *consistent* with a p-class P if the following condition holds. For every conjunctive description C such that:
(a) for every primitive class $A \in \mathcal{A}$, C contains either A or $\neg A$ as a conjunct,
(b) for every attribute $Q \in \mathcal{Q}$, C contains a conjunct (fills $Q\ v$) for some v, and
(c) for every role $R \in \mathcal{R}$, C contains a conjunct $(=\ h\ R)$ for some integer h,[4]
we have that:

1. $\mu(C^I) = N_P(C)$, i.e., the probability of the set of individuals in the interpretation of C is the same as the probability assigned to C by the Bayesian network for P.
2. For every role R, consider the experiment consisting of selecting an object x according to μ, conditioning on the event $x \in C^I$, and picking one of x's R-fillers at random. This experiment generates a new distribution μ'. This new distribution μ' must be consistent with P', where P' is the deterministic value of $N_P(\text{PC(R)} \mid C)$.

[3]It is also possible to ascribe semantics to this language using subjective probabilities, i.e., distribution over possible interpretations. For our purposes, the statistical interpretation is quite natural, and significantly simpler to explain.

[4]We use $(=\ h\ R)$ as a shorthand for $(\leq\ h\ R) \sqcap (\geq\ h\ R)$.

3. Given C, the different fillers are all independent of each other. In other words, the experiment just described generates the same distribution when we also condition on properties of other fillers, and recursively on the properties of fillers of other fillers, and so on. This independence assumption applies both to other fillers of the same role, and to fillers of other roles.

∎

We can now define an interpretation I^p for a P-CLASSIC knowledge base to consist of an interpretation I for the description logic component and a set of probability distributions μ_P^I over $\mathcal{O}^I$, one for every p-class P, such that μ_P^I is consistent with P. For any description D, we say $I^p \models \Pr(D) = \rho$ if $\mu_{P^*}^I(D) = \rho$, where P^* is the root p-class.

One somewhat counterintuitive consequence of the independence assumption is that if any role has non-zero probability of having more than one filler, then any model for the knowledge base must be infinite. In a finite domain, the distribution for R-fillers must assign a non-zero probability to some element. If that element is chosen as the first R-filler, then by the requirement that different role-fillers be distinct, it must have zero probability of being the second R-filler, thereby violating the independence condition.

In an infinite domain, on the other hand, μ_p^I is a probability measure over the domain. While each individual element has probability zero of being selected, the measure assigns a non-zero probability to *properties* of elements. In fact, we can construct a domain of this type, by defining domain elements to correspond to the set of description-logic properties that they satisfy. This construction is the basis for the following theorem. Proofs of theorems are omitted due to space limitations.

Theorem 3.2: *A* P-CLASSIC *knowledge base is always satisfiable in some interpretation.*

This result follows because the acyclicity and locality of the CPTs prevent us from specifying an inconsistent set of probabilistic constraints (including with respect to terminological information). Note that a satisfiable interpretation may assign the empty set to some concepts.

The conditions in Definition 3.1 are sufficient to guarantee that the probability of any description is uniquely determined by a P-CLASSIC knowledge base. As we will see, this result is a consequence of the fact that a Bayesian network uniquely specifies a probability distribution, and of our independence assumptions. The following theorem is proved by induction on the depth of the canonical form of a description.

Theorem 3.3: *For any* P-CLASSIC *knowledge base* Δ*, and any description* C*, there is a unique* $\rho \in [0,1]$ *such that* $\Delta \models (\Pr(C) = \rho)$.

4 Inference Algorithm

It is possible to devise a simple algorithm for computing the probability of a concept of the form $\alpha \sqcap \beta_{R_1} \sqcap \ldots \sqcap \beta_{R_m}$, by first computing the probability of α, and then, recursively, computing the probabilities of $\beta_{R_1} \ldots, \beta_{R_m}$. However, the cost of such a procedure would be exponential in the depth of the concept and in the number of primitive concepts and attributes. To obtain a tractable algorithm, we make two observations. First, probabilities can be computed bottom up, beginning with subexpressions of depth 0. Probabilities of deeper expressions are computed using the stored probabilities of the subexpressions. The second observation is that the probability that β_R holds is completely determined by the number and p-class of the R-fillers. In Bayesian network terms, this probability is *d-separated* (Pearl 1988) from the rest of the network by NUMBER(R) and PC(R). Thus, we can add a node to the network representing the event β_R, with the parents NUMBER(R) and PC(R). For each pair of values (h, P') for the parents, we can compute the conditional probability of β_R given h and P', and add it to the conditional probability table for the new node. We then assert as evidence that all the β_R hold, as well as α, and compute the probability of C by computing the probability of the evidence in the Bayesian network.

We rely on the Bayesian network inference algorithm to compute this probability in the most efficient manner possible. In particular, if the original network is a polytree,[5] thus supporting linear time inference, the new network will continue to support linear time inference after the transformation. The complete algorithm is shown in Figure 3.

Theorem 4.1: *Algorithm* **ComputeProbability** *is sound and complete. In other words, for any description* C *and* P-CLASSIC *knowledge base* Δ*, it returns the number* ρ *such that* $\Delta \models (\Pr(C) = \rho)$.

The following theorem shows that the complexity of the algorithm is linear in the length of C and polynomial in the number of p-classes. It is important to note that the complexity of reasoning in P-CLASSIC is no worse than that of its components, i.e., CLASSIC and Bayes nets alone.

Theorem 4.2 : *The running time of algorithm* **ComputeProbability** *is linear in the length of* C *and quadratic in the number of p-classes in* KB*. If all the Bayesian networks for the p-classes are polytrees, then the running time of* **ComputeProbability** *is also polynomial in the size of the knowledge base.*

As an example, consider computing the probability of $\mathsf{mammal} \sqcap (\geq 1\ eats) \sqcap \forall eats.\mathsf{mammal}$ using the networks in Figure 2. The depth of this expression is 1, and it contains the depth 0 subexpression mammal. The first stage of the computation calculates *Prob*(mammal) for each of the p-classes NATURAL THING, CARNIVORE FOOD and HERBIVORE FOOD; these probabilities are found to be 0.15, 0.3 and 0 respectively. Next we calculate $Prob_{P^*}(\mathsf{mammal} \sqcap (\geq 1\ eats) \sqcap \forall eats.\mathsf{mammal})$. In this

[5] A *polytree* is a Bayesian network whose underlying undirected graph is acyclic, i.e., there is only one path of influence between any pair of nodes. Polytrees support linear time probabilistic inference (Pearl 1988).

ALGORITHM **ComputeProbability**(C, KB)
// C is a description in canonical form.
// KB is a P-CLASSIC knowledge base.
// Returns ρ such that $KB \models (\Pr(C) = \rho)$.
// **AddNode**(BN, N, Π, CPT) adds node N with parents Π and
// conditional probability table CPT to Bayesian network BN.
// **AddEvidence**(BN, N = v) asserts that N has value V in BN.
// **Evaluate**(BN) computes the probability of the evidence in BN.
for x = 0 to $depth(C)$
  for each p-class P and subexpression C'
    of depth x in C **do:**
  // when $x = depth(C)$, only do this for the root p-class P^*
  // $C' = \alpha \sqcap \beta_{R_1} \sqcap \ldots \sqcap \beta_{R_m}$,
  // $\beta_{R_j} = (\geq\ l_j\ R_j) \sqcap (\leq\ u_j\ R_j) \sqcap (\forall\ R_j.D_j)$
    BN := N_P
    for j := 1 to m // skip if m = 0
      for h := 0 to b_j // b_j is the bound on the
        for each p-class P' // number of R_j-fillers
          if $h \geq l_j$ **and** $h \leq u_j$
            CPT[h, P', *true*] := *Prob*$_{P'}(D_j)^h$
          else CPT[h, P', *true*] := 0
          CPT[h, P', *false*] = 1 - CPT[h, P', *true*]
      Π := {NUMBER(R_j), PC(R_j)}
      AddNode(BN, β_{R_j}, Π, CPT)
      AddEvidence(BN, β_{R_j} = *true*)
    AddEvidence(BN, α)
    // i.e., **AddEvidence** for each conjunct in α
    Prob$_P(C')$ = **Evaluate**(BN)
return *Prob*$_{P^*}(C)$

Figure 3: Algorithm **ComputeProbability**

description, α is mammal, and there is one β component corresponding to the *eats* role. P^* is the p-class NATURAL THING. A node is added to N_{P^*} for β_{eats}, with the parents NUMBER(EATS) and PC(EATS). Figure 4 shows the conditional probability that β_{eats} is true for each value of its parents. Because the description requires that ($\geq$ 1 *eats*), the entries in the first column are all 0. When there is exactly one filler of the p-class NATURAL THING, the entry is 0.15, which is the previously computed value of *Prob*$_{NT}$(mammal). The remaining entries in the first row are 0.15^2, 0.15^3 and so on. Similarly, the entries for CARNIVORE FOOD are 0.3, 0.3^2, etc. The entries for HERBIVORE FOOD are all 0 because *Prob*$_{HF}$(mammal) = 0. The probability of our query is then computed in the resulting network, by asserting mammal and β_{eats} to be true, resulting in an answer of approximately 0.007.

5 Discussion and Extensions

The motivation in designing P-CLASSIC was to develop a tractable first-order probabilistic logic. To that end, we have shown that P-CLASSIC has a sound, complete and efficient inference algorithm. In this section, we discuss several features and limitations of the expressive power of P-CLASSIC.

	0	1	2	3	4	5	6
NT	0	0.15	0.023	0.004	0.001	0.000	0.000
CF	0	0.3	0.09	0.027	0.008	0.002	0.001
HF	0	0	0	0	0	0	0

Figure 4: The probability of ($\geq$ 1 *eats*) $\sqcap$ ($\forall$*eats*.mammal) given values of NUMBER(EATS) and PC(EATS).

We mention several possible extensions both to the underlying description logic and to the probabilistic component. We examine the extent to which these extensions can be accommodated in our framework and how they would affect the complexity of reasoning.

5.1 The Underlying Description Logic

P-CLASSIC can be easily extended to handle disjunctive concepts, existential quantification, negation on arbitrary concepts (not only primitive ones) and qualified number restrictions. Our semantics provide a well-defined probability for descriptions using these constructors. However, the inference algorithm for computing probabilities for such descriptions is significantly more complicated, and is no longer polynomial in the length of the description. This is not surprising, since the lower bounds (NP-hardness or worse) of subsumption in the presence of these constructs will apply to the probabilistic extension as well.

Another restriction, as mentioned above, is that the number of fillers for each role be bounded. This restriction exists for two reasons. First, it allows us to completely specify the distribution over the number of fillers for a role. Second, the inference algorithm considers all possible values for the number of fillers, so this restriction is required to guarantee that the algorithm terminates. We can address the first issue by expressing the probability that the number of fillers is n using a closed form function $f(n)$. The inference problem is somewhat harder. In certain cases, the algorithm may be able to compute a closed form expression for an infinite series involving $f(n)$. Alternatively, arbitrarily close approximations to the true probability can be obtained by summing a sufficiently long finite prefix of this series.

The only constructor from CLASSIC that is not included in P-CLASSIC is same-as, which enables to describe equality of attribute paths (e.g., to state that the path wife.father reaches the same individual as the path mother). Such equalities are harder to incorporate into our language because our semantics depends heavily on the assumption that different fillers are completely independent. Equality of individuals reached via different filler chains obviously contradicts this independence assumption.

5.2 The probabilistic component

The tractability of our language rests heavily on the independence assumptions made in the probabilistic component: (1) the properties of different fillers are independent, and (2) the properties of an object and of one of its fillers are independent given the filler's p-class. For example, our assumptions prevent us from stating that a non-carnivore always eats at least one vegetable. Such an assertion would imply that the *eats*-fillers are not independent, since if all but one of the fillers are not vegetables, the last one must be. Thus, although we can (albeit at a high computational cost) compute the probability of any description with disjunctions or existentials, we cannot assert terminological properties involving these constructs without modifying the semantics.

One type of correlation between fillers can actually be dealt with fairly easily within our framework. Assume that our vocabulary contains the concept healthy-to-eat. We may want to assert that the healthiness of the various foods eaten by a person tends to be correlated. We can accomplish this by introducing a new concept health-conscious and two p-classes representing HEALTHY FOOD and UNHEALTHY FOOD. The value of the node PC(EATS) can now depend on the value of HEALTH-CONSCIOUS, in that a value of HEALTHY FOOD is more likely when health-conscious is true. This new concept is a *hidden concept* (analogous to a hidden variable in Bayesian networks). It plays the same role as a regular primitive concept in the definition of the p-class, but it is not a primitive concept in the language and therefore does not appear in the terminology or in queries.

One promising alternative with greater expressive power arises from our recent work on functional stochastic programs (Koller *et al.* 1997). The basic idea is that we view each p-class as a "stochastic function", and each individual as a call to the function for that appropriate p-class. A call to such a stochastic function chooses (randomly) the properties of the individual for which it is called. The properties of the fillers for an individual are chosen by recursive calls to the functions for the appropriate p-classes. Once we view a filler as a function, we can pass the relevant properties of the individual (e.g., on-a-diet) as a parameter to the function. The parameters can directly influence the distribution of the filler properties. In our example of Section 2, the p-classes for CARNIVORE FOOD and HERBIVORE FOOD are equivalent to those that would have been obtained had we passed the value of animal to the filler as a parameter (with a value of *true* and *false* respectively).

In addition to taking arguments, a stochastic function can also "return values". In our context, this feature would allow properties of an object to depend on those of its fillers. Thus, for example, we can have the probability over the cholesterol-level attribute depend on the actual fat content of the foods eaten by the person. We could represent this type of dependency by introducing a node into the Bayesian network corresponding to the complex concept $\forall$*eats*.low-fat. The node representing cholesterol-level can now be a child of this new node, encoding the desired dependency. In general, we could have fillers recursively "pass back" the truth value of deeply nested complex concepts. This extension supports concept introductions $A \subseteq D$ where D is an arbitrary complex concept. Somewhat surprisingly, we can accomodate the functional view of fillers fairly easily within our framework, and without a significant increase in computational cost. We defer details to the full version of this paper.

Acknowledgements We thank Manfred Jaeger for useful discussions. Some of this work was done while Daphne Koller and Avi Pfeffer were visiting AT&T Research. This work was also supported through the generosity of the Powell foundation, by ONR grant N00014-96-1-0718, and by DARPA contract DACA76-93-C-0025, under subcontract to Information Extraction and Transport, Inc.

References

Y. Arens, C. Knoblock, and W. Shen. Query reformulation for dynamic information integration. *J. on Intelligent and Cooperative Information Systems*, (6) 2/3, 1996.

F. Bacchus. *Representing and Reasoning with Probabilistic Knowledge*. MIT Press, Cambridge, MA, 1990.

Alexander Borgida and Peter Patel-Schneider. A semantics and complete algorithm for subsumption in the CLASSIC description logic. *Journal of Artificial Intelligence Research*, 1, 1994.

R. Brachman, A. Borgida, D. McGuinness, P. Patel-Schneider, and L. Resnick. Living with CLASSIC: When and how to use a KL-ONE-like language. In John Sowa, editor, *Principles of Semantic Networks*. Morgan Kaufmann, 1991.

J. S. Breese. Construction of belief and decision networks. *Computational Intelligence*, 1992.

R. P. Goldman and E. Charniak. Dynamic construction of belief networks. *UAI*, 1990.

J. Y. Halpern. An analysis of first-order logics of probability. *Artificial Intelligence*, 46, 1990.

J. Heinsohn. A hybrid approach for modeling uncertainty in terminological logics. In *Proc. ECSQARU*. Springer Verlag, 1991.

M. Jaeger. A logic for default reasoning about probabilities. In *UAI*, 1994.

D. Koller, D. McAllester, and A. Pfeffer. Effective Bayesian inference for stochastic programs. In *AAAI*, 1997.

A. Levy, A. Rajaraman, and J. Ordille. Query answering algorithms for information agents. In *AAAI*, 1996.

R. M. MacGregor. A deductive pattern matcher. In *AAAI*, 1988.

L. Ngo, P. Haddawy, and J. Helwig. A theoretical framework for context-sensitive temporal probability model construction with application to plan projection. In *UAI*, 1995.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, 1988.

C. Petalson. The BACK system: an overview. In *SIGART bulletin*, volume 2(3), 1991.

D. Poole. Probabilistic horn abduction and bayesian networks. *Artificial Intelligence*, 64(1), November 1993.

L. Shastri. Default reasoning in semantic networks: a formalization of recognition and inheritance. *Artificial Intelligence*, 39(3), 1989.

J. Wright, E. Weixelbaum, K. Brown, G. Vesonder, S. Palmer, J. Berman, and H. Moore. A knowledge-based configurator that supports sales, engineering and manufacturing at AT&T network systems. In *IAAI*, 1993.