INSTITUT FÜR INFORMATIONSSYSTEME
ABTEILUNG WISSENSBASIERTEN SYSTEMEN

NONMONOTONIC PROBABILISTIC LOGICS
UNDER VARIABLE-STRENGTH INHERITANCE
WITH OVERRIDING: ALGORITHMS AND IMPLEMENTATION IN NMPROBLOG

THOMAS LUKASIEWICZ

INFSYS RESEARCH REPORT 1843-05-03
MAY 2005
Abstract. In previous work, I have introduced nonmonotonic probabilistic logics under variable-strength inheritance with overriding. They are formalisms for probabilistic reasoning from sets of strict logical, default logical, and default probabilistic sentences, which are parameterized through a value $\lambda \in [0, 1]$ that describes the strength of the inheritance of default probabilistic knowledge. In this paper, I continue this line of research. I present algorithms for deciding consistency of strength $\lambda$ and for computing tight consequences of strength $\lambda$, which are based on reductions to the standard problems of deciding satisfiability and of computing tight logical consequences in model-theoretic probabilistic logic. Furthermore, I describe an implementation in the system NMPROBLOG.
## Contents

1 Introduction

2 Preliminaries
   2.1 Probabilistic Knowledge Bases
   2.2 Logical Entailment of Strength \( \lambda \)

3 Nonmonotonic Probabilistic Logics
   3.1 Consistency of Strength \( \lambda \)
   3.2 System Z of Strength \( \lambda \)
   3.3 Lexicographic Entailment of Strength \( \lambda \)

4 Semantic Properties
   4.1 General Nonmonotonic Properties
   4.2 Relationships between Probabilistic Formalisms
   4.3 Probabilistic and Classical Special Cases
   4.4 Relationship to G-Coherent Entailment

5 Further Examples

6 Algorithms

7 The System **NMPROBLOG**

8 Summary and Outlook
1 Introduction

During the recent decades, reasoning about probabilities has started to play an important role in artificial intelligence (AI). In particular, reasoning about interval restrictions for conditional probabilities, also called conditional constraints [44], has been a subject of extensive research efforts. Roughly, a conditional constraint is of the form \((\psi|\phi)[l, u]\), where \(\psi\) and \(\phi\) are events, and \([l, u]\) is a subinterval of the unit interval \([0, 1]\). It encodes that the conditional probability of \(\psi\) given \(\phi\) lies in \([l, u]\).

An important approach for handling conditional constraints is model-theoretic probabilistic logic, which has its origin in philosophy and logic, and whose roots can be traced back to already Boole in 1854 [13]. There is a wide spectrum of formal languages that have been explored in model-theoretic probabilistic logic, ranging from constraints for unconditional and conditional events to rich languages that specify linear inequalities over events (see especially the work by Nilsson [51], Fagin et al. [20], Dubois and Prade et al. [14, 18, 2, 17], Frisch and Haddawy [22], and the author [43, 44, 46]; see also the survey on sentential probability logic by Hailperin [34]). The main decision and optimization problems in probabilistic logic are deciding satisfiability, deciding logical consequence, and computing tight logically entailed intervals.

For example, a simple collection of conditional constraints \(KB\) may encode the strict logical knowledge “all eagles are birds” and “all birds have feathers” as well as the purely probabilistic knowledge “birds fly with a probability of at least 0.95”. This \(KB\) is satisfiable, and some logical consequences in model-theoretic probabilistic logic from \(KB\) are “all birds have feathers”, “birds fly with a probability of at least 0.95”, “all eagles have feathers”, and “eagles fly with a probability between 0 and 1”; in fact, these are the tightest intervals that follow from \(KB\). That is, we especially cannot conclude anything from \(KB\) about the ability to fly of eagles.

A closely related research area is default reasoning from conditional knowledge bases, which consist of a collection of strict statements in classical logic and a collection of defeasible rules, also called defaults. The former must always hold, while the latter are rules of the kind \(\psi \leftarrow \phi\), which read as “generally, if \(\phi\) then \(\psi\).” Such rules may have exceptions, which can be handled in different ways.

The literature contains several different proposals for default reasoning from conditional knowledge bases and extensive work on its desired properties. The core of these properties are the rationality postulates of System \(P\) by Kraus, Lehmann, and Magidor [35], which constitute a sound and complete axiom system for several classical model-theoretic entailment relations under uncertainty measures on worlds. They characterize classical model-theoretic entailment under preferential structures [57, 35], infinitesimal probabilities [1, 52], possibility measures [15], and world rankings [58, 32]. As shown by Friedman and Halpern [21], many of these uncertainty measures on worlds are expressible as plausibility measures. The postulates of System \(P\) also characterize an entailment relation based on conditional objects [16]. A survey of the above relationships is given in [6, 23].

Mainly to solve problems with irrelevant information, the notion of rational closure as a more adventurous notion of entailment was introduced by Lehmann [40, 41]. It is equivalent to entailment in System \(Z\) by Pearl [53], to the least specific possibility entailment by Benferhat et al. [5], and to a conditional (modal) logic-based entailment by Lamarre [39]. Finally, mainly to solve problems with property inheritance from classes to exceptional subclasses, the maximum entropy approach to default entailment was proposed by Goldszmidt et al. [30]; lexicographic entailment was introduced by Lehmann [42] and Benferhat et al. [4]; conditional entailment was proposed by Geffner [24, 25]; and an infinitesimal belief function approach was suggested by Benferhat et al. [7].

For example, a conditional knowledge base \(KB\) may encode the strict logical knowledge “all ostriches are birds” and the default logical knowledge “generally, birds fly”, “generally, ostriches do not fly”, and
“generally, birds have wings”. Some desirable conclusions from \( KB \) [33] are “generally, birds fly” and “generally, birds have wings” (which both belong to \( KB \)), “generally, ostriches have wings” (since the set of all ostriches is a subclass of the set of all birds, and thus ostriches should inherit all properties of birds), “generally, ostriches do not fly” (since properties of more specific classes should override inherited properties of less specific classes), and “generally, red birds fly” (since “red” is not mentioned at all in \( KB \) and thus should be irrelevant to the ability to fly of birds).

There are several works in the literature on probabilistic foundations for default reasoning from conditional knowledge bases [1, 52, 30, 12], on combinations of Reiter’s default logic [56] with statistical inference [38, 59], and on a rich first-order formalism for deriving degrees of belief from statistical knowledge including default statements [3]. A series of recent papers has proposed combinations of model-theoretic probabilistic reasoning from conditional constraints with default reasoning from conditional knowledge bases, which are summarized as follows:

- The paper [50] presents weak nonmonotonic probabilistic logics, which are extensions of probabilistic logic by defaults as in conditional knowledge bases under Kraus et al.’s System \( P \) [35], Pearl’s System \( Z \) [53], and Lehmann’s lexicographic entailment [42]. The new formalisms allow for expressing in a uniform framework strict logical knowledge and purely probabilistic knowledge from probabilistic logic, as well as default logical knowledge from default reasoning from conditional knowledge bases. For example, consider the strict logical knowledge “all penguins are birds”, the default logical knowledge “generally, birds have legs” and “generally, birds fly”, and the purely probabilistic knowledge “penguins fly with a probability of at most 0.05”. Obviously, some desired conclusions are “generally, birds have legs”, “generally, birds fly”, and “penguins fly with a probability of at most 0.05”, since these sentences are explicitly stated above. Two other desired conclusions are “generally, penguins have legs” (since the property of having legs of birds should be inherited down to the subclass of all penguins) and “generally, red birds fly” (since the property of being red is not mentioned at all above, and thus should be irrelevant to the ability to fly). In weak nonmonotonic probabilistic logics, we can deal with all the above sentences. In particular, the notion of probabilistic lexicographic entailment also produces all the above desired conclusions.

- A companion paper [47] presents strong nonmonotonic probabilistic logics, which are similar probabilistic generalizations of default reasoning from conditional knowledge bases. They are, however, quite different from the ones in [50] in that they allow for handling default purely probabilistic knowledge, rather than (strict) purely probabilistic knowledge, in addition to strict logical knowledge and default logical knowledge. For example, they allow for expressing sentences “generally, birds fly with a probability of at least 0.95” rather than “birds fly with a probability of at least 0.95”. Intuitively, the former means that being able to fly with a probability of at least 0.95 should apply to all birds and all subclasses of birds, as long as this is consistent, while the latter says that being able to fly with a probability of at least 0.95 should only apply to all birds. This is why the formalisms in [47] are generally much stronger than the ones in [50].

- Finally, the papers [48, 49] define nonmonotonic probabilistic logics under variable-strength inheritance with overriding, which are a general approach to nonmonotonic probabilistic reasoning, which subsumes the approaches in [50] and [47] as special cases. Roughly, these formalisms also allow for handling strict logical knowledge, default logical knowledge, and default purely probabilistic knowledge, but the inheritance of purely probabilistic knowledge is controlled by a strength \( \lambda \in [0, 1] \). For \( \lambda = 0 \) (resp., \( \lambda = 1 \)), these formalisms coincide with the weak (resp., strong) nonmonotonic ones in
For example, suppose that we have the default probabilistic knowledge "generally, yellow objects are easy to see with a probability between 0.8 and 0.9". In nonmonotonic probabilistic reasoning of strength 0 (resp., 0.5 and 1), we then conclude "generally, yellow birds are easy to see with a probability in [0, 1] (resp., [0.6, 1] and [0.8, 0.9])".

To date, however, there have been no algorithms for nonmonotonic probabilistic logics under variable-strength inheritance with overriding. Furthermore, there have been no implementations, neither of these unifying formalisms, nor of the special cases of weak and strong nonmonotonic probabilistic logics. In this paper, I try to fill this gap. The main contributions can be summarized as follows:

- I recall the nonmonotonic probabilistic logics under variable-strength inheritance with overriding presented in [48, 49], namely, probabilistic entailment in System Z of strength $\lambda$ (or $z_\lambda$-entailment) and probabilistic lexicographic entailment of strength $\lambda$ (or $lex_\lambda$-entailment). I also recall their semantic properties, and I provide several new examples.

- I present an algorithm for deciding consistency of strength $\lambda$, which is based on a reduction to deciding satisfiability in model-theoretic probabilistic logic. I also present algorithms for computing tight entailed intervals under $z_\lambda$-entailment and $lex_\lambda$-entailment, which are based on reductions to deciding satisfiability and computing tight logically entailed intervals in model-theoretic probabilistic logic.

- I describe the system NMPROBLOG, which includes implementations of the above algorithms. Here, deciding satisfiability and computing tight logically entailed intervals in model-theoretic probabilistic logic are reduced to deciding the solvability of a system of linear constraints and to solving linear programs, respectively, which is done using "lp_solve" [9].

The rest of this paper is organized as follows. Section 2 gives some technical preliminaries. In Sections 3 and 4, we recall the notions of $z_\lambda$- and $lex_\lambda$-entailment, and their semantic properties. In Section 5, we give some further examples to illustrate the notions of $z_\lambda$- and $lex_\lambda$-entailment. Sections 6 describe the algorithms for deciding consistency of strength $\lambda$ and computing tight entailed intervals under $z_\lambda$- and $lex_\lambda$-entailment. In Section 7, we present the system NMPROBLOG. Section 8 summarizes the main results and gives an outlook on future research.

2 Preliminaries

In this section, we recall probabilistic knowledge bases and the main concepts from model-theoretic probabilistic logic. Furthermore, we define the monotonic notion of logical entailment of strength $\lambda \in [0, 1]$.

2.1 Probabilistic Knowledge Bases

We now recall the concept of a probabilistic knowledge base. We start by defining logical constraints and probabilistic formulas, which are interpreted by probability distributions over a set of possible worlds.

We assume a set of basic events $\Phi = \{p_1, \ldots, p_n\}$ with $n \geq 1$. We use $\bot$ and $\top$ to denote false and true, respectively. We define events by induction as follows. Every element of $\Phi \cup \{\bot, \top\}$ is an event. If $\phi$ and $\psi$ are events, then also $\neg \phi$ and $(\phi \land \psi)$. A conditional event is an expression of the form $\psi|\phi$, where $\psi$ and $\phi$ are events. A conditional constraint has the form $(\psi|\phi)[l, u]$, where $\psi$ and $\phi$ are events, and $l, u \in [0, 1]$ are reals. We define probabilistic formulas by induction as follows. Every conditional constraint is a probabilistic formula. If $F$ and $G$ are probabilistic formulas, then also $\neg F$ and $(F \land G)$. We use $(F \lor G)$
and \((F \Leftrightarrow G)\) to abbreviate \(-(-F \land -G)\) and \(-(-F \land G)\), respectively, where \(F\) and \(G\) are either two events or two probabilistic formulas, and adopt the usual conventions to eliminate parentheses. A *logical constraint* is an event of the form \(\psi \Leftrightarrow \phi\).

A *world* \(I\) is a truth assignment to the basic events in \(\Phi\) (that is, a mapping \(I: \Phi \rightarrow \{\text{true}, \text{false}\}\)), which is inductively extended to all events as usual (that is, \(I(\bot) = \text{false}, I(\top) = \text{true}, I(\lnot \phi) = \text{true} \iff I(\phi) = \text{false}\), and \(I(\phi \land \psi) = \text{true} \iff I(\phi) = I(\psi) = \text{true}\)). We denote by \(\mathcal{I}_\Phi\) the set of all worlds for \(\Phi\). A world \(I\) *satisfies* an event \(\phi\), or \(I\) is a *model* of \(\phi\), denoted \(I \models \phi\), iff \(I(\phi) = \text{true}\). A *probabilistic interpretation* \(Pr\) is a probability function on \(\mathcal{I}_\Phi\) (that is, a mapping \(Pr: \mathcal{I}_\Phi \rightarrow [0,1]\) such that all \(Pr(I)\) with \(I \in \mathcal{I}_\Phi\) sum up to 1). The *probability* of an event \(\phi\) in \(Pr\), denoted \(Pr(\phi)\), is the sum of all \(Pr(I)\) such that \(I \in \mathcal{I}_\Phi\) and \(I \models \phi\). For events \(\phi\) and \(\psi\) with \(Pr(\phi) > 0\), we write \(Pr(\psi|\phi)\) to abbreviate \(Pr(\psi \land \phi) / Pr(\phi)\). The *truth* of logical constraints and probabilistic formulas \(F\) in \(Pr\), denoted \(Pr \models F\), is defined by:

- \(Pr \models \psi \Leftrightarrow \phi \iff Pr(\psi \land \phi) = Pr(\phi)\);
- \(Pr \models (\psi|\phi)[l, u] \iff Pr(\phi) = 0 \text{ or } Pr(\psi|\phi) \in [l, u]\);
- \(Pr \models \lnot F \iff \text{not } Pr \models F\);
- \(Pr \models (F \land G) \iff Pr \models F \text{ and } Pr \models G\).

We say \(Pr\) *satisfies* \(F\), or \(Pr\) is a *model* of \(F\), iff \(Pr \models F\). We say \(Pr\) *satisfies* a set of logical constraints and probabilistic formulas \(F\), or \(Pr\) is a *model* of \(F\), denoted \(Pr \models F\), iff \(Pr\) is a model of all \(F \in \mathcal{F}\). A set of logical constraints and probabilistic formulas \(\mathcal{F}\) is *satisfiable* iff a model of \(\mathcal{F}\) exists. A conditional constraint \((\psi|\phi)[l, u]\) is a *logical consequence* of \(\mathcal{F}\), denoted \(\mathcal{F} \models (\psi|\phi)[l, u]\), iff each model of \(\mathcal{F}\) is also a model of \((\psi|\phi)[l, u]\). It is a *tight logical consequence* of \(\mathcal{F}\), denoted \(\mathcal{F} \models_{\text{tight}} (\psi|\phi)[l, u]\), iff \(l = \inf Pr(\psi|\phi)\) (resp., \(u = \sup Pr(\psi|\phi)\)) subject to all models \(Pr\) of \(\mathcal{F}\) with \(Pr(\phi) > 0\). Here, we define \(l = 1\) and \(u = 0\), when no such model \(Pr\) exists.

A *probabilistic knowledge base* \(KB = (L, P)\) consists of a finite set of logical constraints \(L\) and a finite set of conditional constraints \(P\). We say \(KB\) is *satisfiable* iff \(L \cup P\) is satisfiable. A conditional constraint \(C\) is a *logical consequence of* \(KB\), denoted \(KB \models C\), iff \(L \cup P \models C\). It is a *tight logical consequence* of \(KB\), denoted \(KB \models_{\text{tight}} C\), iff \(L \cup P \models_{\text{tight}} C\). The following example illustrates the syntactic notion of a probabilistic knowledge base.

**Example 2.1** The strict logical knowledge “all penguins are birds”, the default logical knowledge “generally, birds have legs”, and the default purely probabilistic knowledge “generally, yellow objects are easy to see with a probability between 0.8 and 0.9”, “generally, birds fly with a probability of at least 0.9”, and “generally, penguins fly with a probability of at most 0.1” can be expressed by the following probabilistic knowledge base \(KB = (L, P):\)

\[
L = \{\text{bird} \Leftrightarrow \text{penguin}\},
P = \{(\text{legs} | \text{bird})[1, 1], (\text{see} | \text{yellow})[.8, .9], (\text{fly} | \text{bird})[.9, 1], (\text{fly} | \text{penguin})[0, .1]\}. \quad \square
\]

### 2.2 Logical Entailment of Strength \(\lambda\)

As a first step towards the definition of \(z_{\lambda}\) and \(lex_{\lambda}\)-entailment in Section 3, we now define the monotonic notion of *logical entailment of strength* \(\lambda \in [0, 1]\). It already realizes an inheritance of default purely probabilistic knowledge controlled by \(\lambda\). But, in contrast to \(z_{\lambda}\)- and \(lex_{\lambda}\)-entailment, it does not have an overriding mechanism, and this often produces *local inconsistencies*.
In the sequel, we use $\phi \succsim \lambda$ to abbreviate the probabilistic formula $-(\phi|\top)[0,0] \land (\phi|\top)[\lambda,1]$. Informally, for any probabilistic interpretation $Pr$ that satisfies $\phi \succsim \lambda$, it holds that $Pr(\phi) > 0$, if $\lambda = 0$, and $Pr(\phi) \succsim \lambda$, otherwise. We define the notion of logical entailment of strength $\lambda \in [0,1]$ (or simply $\lambda$-logical entailment) as follows. A conditional constraint $C = (\psi|\phi)[l,u]$ is a $\lambda$-logical consequence of $KB = (L,P)$, denoted $KB \models C$, iff $L \cup P \cup \{\phi \succsim \lambda\} \models C$. It is a tight $\lambda$-logical consequence of $KB$, denoted $KB \models_{\lambda_{\text{tight}}} C$, iff $L \cup P \cup \{\phi \succsim \lambda\} \models_{\text{tight}} C$.

**Example 2.2** Let $KB$ be as in Example 2.1. Some tight logical consequences of strength $\lambda$ among 0, 0.2, 0.4, 0.6, 0.8, and 1 are shown in Table 1 (less desired intervals are red). We observe an inheritance of default logical knowledge along subclass relationships, which is independent from $\lambda$. For example, the default logical property of having legs is inherited from birds down to yellow birds. Furthermore, we observe an inheritance of default purely probabilistic knowledge along subclass relationships, which depends on the strength $\lambda$. For example, being easy to see with a probability in $[.8, .9]$ is inherited from yellow objects down to yellow birds, but the new intervals are $[0,1], [0,1], [.5, 1], [.67, 1], [.75, 1]$, and $[.8, .9]$, respectively. Finally, for $\lambda > 1/9$, there are local inconsistencies related to penguins (since being able to fly with a probability of at least 0.9 is inherited from birds down to penguins, and there it is inconsistent with being able to fly with a probability of at most 0.1).

<table>
<thead>
<tr>
<th>Strength $\lambda$</th>
<th>0</th>
<th>0.2</th>
<th>0.4</th>
<th>0.6</th>
<th>0.8</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>legs</strong></td>
<td>bird</td>
<td>[.1, 1]</td>
<td>[.1, 1]</td>
<td>[.1, 1]</td>
<td>[.1, 1]</td>
<td>[.1, 1]</td>
</tr>
<tr>
<td><strong>legs</strong></td>
<td>yellow&amp; bird</td>
<td>[.1, 1]</td>
<td>[.1, 1]</td>
<td>[.1, 1]</td>
<td>[.1, 1]</td>
<td>[.1, 1]</td>
</tr>
<tr>
<td><strong>legs</strong></td>
<td>penguin</td>
<td>[.1, 1]</td>
<td>[.1, 1]</td>
<td>[.1, 1]</td>
<td>[.1, 1]</td>
<td>[.1, 1]</td>
</tr>
<tr>
<td><strong>fly</strong></td>
<td>bird</td>
<td>[.9, 1]</td>
<td>[.9, 1]</td>
<td>[.9, 1]</td>
<td>[.9, 1]</td>
<td>[.9, 1]</td>
</tr>
<tr>
<td><strong>fly</strong></td>
<td>yellow&amp; bird</td>
<td>[.0, 1]</td>
<td>[.5, 1]</td>
<td>[.75, 1]</td>
<td>[.83, 1]</td>
<td>[.88, 1]</td>
</tr>
<tr>
<td><strong>fly</strong></td>
<td>penguin</td>
<td>[.0, 1]</td>
<td>[.1, 1]</td>
<td>[.1, 1]</td>
<td>[.1, 1]</td>
<td>[.1, 1]</td>
</tr>
<tr>
<td><strong>fly</strong></td>
<td>yellow&amp; penguin</td>
<td>[.0, 1]</td>
<td>[.1, 1]</td>
<td>[.1, 1]</td>
<td>[.1, 1]</td>
<td>[.1, 1]</td>
</tr>
<tr>
<td><strong>see</strong></td>
<td>yellow</td>
<td>[.8, 9]</td>
<td>[.8, 9]</td>
<td>[.8, 9]</td>
<td>[.8, 9]</td>
<td>[.8, 9]</td>
</tr>
<tr>
<td><strong>see</strong></td>
<td>yellow&amp; bird</td>
<td>[.0, 1]</td>
<td>[.5, 1]</td>
<td>[.67, 1]</td>
<td>[.75, 1]</td>
<td>[.8, 9]</td>
</tr>
<tr>
<td><strong>see</strong></td>
<td>yellow&amp; penguin</td>
<td>[.0, 1]</td>
<td>[.1, 1]</td>
<td>[.1, 1]</td>
<td>[.1, 1]</td>
<td>[.1, 1]</td>
</tr>
</tbody>
</table>

### 3 Nonmonotonic Probabilistic Logics

In this section, we recall the notions of $z_\lambda$- and $lex_\lambda$-entailment from [48, 49]. They are parameterized through a value $\lambda \in [0,1]$ that describes the strength of the inheritance of default purely probabilistic knowledge.

#### 3.1 Consistency of Strength $\lambda$

We now recall the notion of consistency of strength $\lambda$ (or $\lambda$-consistency) for probabilistic knowledge bases $KB = (L,P)$. We first give some preparative definitions.
A probabilistic interpretation $Pr \lambda$-verifies (resp., $\lambda$-falsifies) a conditional constraint $(\psi|\phi)[l, u]$ iff $Pr$ verifies (resp., falsifies) $(\psi|\phi)[l, u]$ and satisfies $\phi \equiv \lambda$. A set of conditional constraints $P \lambda$-tolerates a conditional constraint $C$ under a set of logical constraints $L$ iff $L \cap P$ has a model that $\lambda$-verifies $C$. We say $P$ is under $L$ in $\lambda$-conflict with $C$ iff no model of $L \cup P$ $\lambda$-verifies $C$. A conditional constraint ranking $\sigma$ on $KB = (L, P)$ is $\lambda$-admissible with $KB$ iff every $P' \subseteq P$ that is under $L$ in $\lambda$-conflict with some $C \in P$ contains some $C'$ such that $\sigma(C') < \sigma(C)$.

We are now ready to define the notion of $\lambda$-consistency. We say $KB$ is $\lambda$-consistent iff there exists a conditional constraint ranking $\sigma$ on $KB$ that is $\lambda$-admissible with $KB$. The following theorem characterizes the $\lambda$-consistency of $KB$ through the existence of an ordered partition of $P$.

**Theorem 3.1** A probabilistic knowledge base $KB = (L, P)$ is $\lambda$-consistent iff an ordered partition $(P_0, \ldots, P_k)$ of $P$ exists such that each $P_i$, $0 \leq i \leq k$, is the set of all $C \in \bigcup_{j=i}^k P_j$ that are $\lambda$-tolerated under $L$ by $\bigcup_{j=i}^k P_j$.

The unique ordered partition $(P_0, \ldots, P_k)$ of $P$ in Theorem 3.1 is called the $z_\lambda$-partition of $KB = (L, P)$. Hence, $KB$ is $\lambda$-consistent iff its $z_\lambda$-partition exists. The following example shows some $z_\lambda$-partitions.

**Example 3.2** Let $KB = (L, P)$ be as in Example 2.1. For every $\lambda \in [0, 1/9]$, the $z_\lambda$-partition is given by $(P_0) = (P)$. For every $\lambda \in (1/9, 1]$, the $z_\lambda$-partition is given by $(P_0, P_1) = (P \setminus \{(fly\land penguin)[0, .1]\}, \{(fly\land penguin)[0, .1]\})$. Thus, $KB$ is $\lambda$-consistent, for all $\lambda \in [0, 1]$.

It can also be shown that $KB = (L, P)$ is $\lambda$-consistent iff a probability ranking $\kappa$ exists that is $\lambda$-admissible with $KB$. Formally, a probability ranking $\kappa$ maps each probabilistic interpretation on $\mathcal{I}_\Phi$ to a member of $\{0, 1, \ldots\} \cup \{\infty\}$ such that $\kappa(Pr) = 0$ for at least one interpretation $Pr$. It is extended to all logical constraints and probabilistic formulas $F$ as follows. If $F$ is satisfiable, then $\kappa(F) = \min \{\kappa(Pr) \mid Pr \models F\}$; otherwise, $\kappa(F) = \infty$. A probability ranking $\kappa$ is $\lambda$-admissible with a probabilistic knowledge base $KB = (L, P)$ iff $\kappa(\neg F) = \infty$ for all $F \in L$ and $\kappa(\phi \equiv \lambda) < \infty$ and $\kappa(\phi \equiv \lambda \land (\psi|\phi)[l, u]) < \kappa(\phi \equiv \lambda \land \neg(\psi|\phi)[l, u])$ for all $(\psi|\phi)[l, u] \in P$.

### 3.2 System Z of Strength $\lambda$

We next recall the notion of $z_\lambda$-entailment, $\lambda \in [0, 1]$, for $\lambda$-consistent probabilistic knowledge bases $KB = (L, P)$. It is linked to a conditional constraint ranking $z_\lambda$ on $KB$ and a probability ranking $\kappa^{z_\lambda}$. Let $(P_0, \ldots, P_k)$ be the $z_\lambda$-partition of $KB$. For every $j \in \{0, \ldots, k\}$, each $C \in P_j$ is assigned the value $j$ under $z_\lambda$. Then, $\kappa^{z_\lambda}$ on all probabilistic interpretations $Pr$ is defined as follows:

$$
\kappa^{z_\lambda}(Pr) = \begin{cases} 
\infty & \text{if } Pr \not\models L \\
0 & \text{if } Pr \models L \cup P \\
1 + \max_{C \in P \setminus C : Pr \not\models C} z_\lambda(C) & \text{otherwise.}
\end{cases}
$$

The probability ranking $\kappa^{z_\lambda}$ defines a preference relation on probabilistic interpretations: For probabilistic interpretations $Pr$ and $Pr'$, we say $Pr$ is $z_\lambda$-preferable to $Pr'$ if $\kappa^{z_\lambda}(Pr') < \kappa^{z_\lambda}(Pr)$. A model $Pr$ of a set of logical constraints and probabilistic formulas $\mathcal{F}$ is a $z_\lambda$-minimal model of $\mathcal{F}$ iff no model of $\mathcal{F}$ is $z_\lambda$-preferable to $Pr$.

We are now ready to define the notion of $z_\lambda$-entailment as follows. A conditional constraint $C = (\psi|\phi)[l, u]$ is a $z_\lambda$-consequence of $KB$, denoted $KB \models^{z_\lambda} C$, iff every $z_\lambda$-minimal model of $L \cup \{\phi \equiv \lambda\}$
satisfies \( C \). It is a tight \( z_\lambda \)-consequence of \( KB \), denoted \( KB \models^\sim z_\lambda C \), iff 
\( l = \inf Pr(\psi | \phi) \) (resp., \( u = \sup Pr(\psi | \phi) \)) subject to all \( z_\lambda \)-minimal models \( Pr \) of \( L \cup \{ \phi \models \lambda \} \).

The following example shows that the notion of \( z_\lambda \)-entailment realizes an inheritance of default logical and default purely probabilistic properties from classes to non-exceptional subclasses, where the inheritance of default purely probabilistic properties depends on the strength \( \lambda \). However, \( z_\lambda \)-entailment does not inherit properties from classes to subclasses that are exceptional relative to some other property (and thus, like its classical counterpart, shows the problem of inheritance blocking).

**Example 3.3** Let \( KB = (L, P) \) be as in Example 2.1. Some tight \( z_\lambda \)-consequences, where \( \lambda \in \{0, 0.2, 0.4, 0.6, 0.8, 1\} \), are shown in Table 2. Observe that, in contrast to Table 1, there are no empty intervals “[1,0]”, that is, no local inconsistencies. Then, observe that the default logical property of having legs is inherited from the class of birds down to yellow birds, independently from \( \lambda \), while the default purely probabilistic property of being easy to see with a probability between 0.8 and 0.9 is also inherited from the class of yellow objects to yellow birds, but this is controlled by \( \lambda \). Furthermore, for every strength \( \lambda > 1/9 \), these properties are not inherited down to the exceptional classes of penguins and yellow penguins, respectively. Note that for every strength \( \lambda \leq 1/9 \), the default logical property of having legs is inherited down to penguins, since there is only some weak inheritance of default purely probabilistic knowledge, and thus no conflict between the abilities of fly of birds and penguins.

| \( \text{legs} \) | \( \text{bird} \) | \( \text{legs} \) | \( \text{yellow} \land \text{bird} \) | \( \text{legs} \) | \( \text{penguin} \) | \( \text{legs} \) | \( \text{yellow} \land \text{penguin} \) | \( \text{fly} \) | \( \text{bird} \) | \( \text{fly} \) | \( \text{yellow} \land \text{bird} \) | \( \text{fly} \) | \( \text{penguin} \) | \( \text{fly} \) | \( \text{yellow} \land \text{penguin} \) | \( \text{see} \) | \( \text{yellow} \) | \( \text{see} \) | \( \text{yellow} \land \text{bird} \) | \( \text{see} \) | \( \text{yellow} \land \text{penguin} \) |
| \( \{1,1\} \) | \( \{1,1\} \) | \( \{1,1\} \) | \( \{1,1\} \) | \( \{1,1\} \) | \( \{0,1\} \) | \( \{0,1\} \) | \( \{0,1\} \) | \( \{9,1\} \) | \( \{9,1\} \) | \( \{9,1\} \) | \( \{9,1\} \) | \( \{9,1\} \) | \( \{9,1\} \) | \( \{9,1\} \) | \( \{8.9\} \) | \( \{8.9\} \) | \( \{8.9\} \) | \( \{8.9\} \) | \( \{8.9\} \) | \( \{8.9\} \) | \( \{8.9\} \) | \( \{8.9\} \) |

Table 2: Some tight \( z_\lambda \)-consequences.

<table>
<thead>
<tr>
<th>( \text{Strength} \lambda )</th>
<th>( 0 )</th>
<th>( 0.2 )</th>
<th>( 0.4 )</th>
<th>( 0.6 )</th>
<th>( 0.8 )</th>
<th>( 1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{see} )</td>
<td>( \text{yellow} )</td>
<td>( {0,1} )</td>
<td>( {0,1} )</td>
<td>( {0,1} )</td>
<td>( {0,1} )</td>
<td>( {0,1} )</td>
</tr>
</tbody>
</table>

### 3.3 Lexicographic Entailment of Strength \( \lambda \)

We finally recall the notion of \( lex_\lambda \)-entailment for \( \lambda \)-consistent probabilistic knowledge bases \( KB = (L, P) \). Note that, like \( z_\lambda \)-entailment, it can also be expressed in terms of a unique probability ranking for \( KB \).

We use the \( z_\lambda \)-parition \( (P_0, \ldots, P_k) \) of \( KB \) to define a lexicographic preference relation on probabilistic interpretations as follows. For probabilistic interpretations \( Pr \) and \( Pr' \), we say \( Pr \) is \( lex_\lambda \)-preferable to \( Pr' \) iff some \( i \in \{0, \ldots, k\} \) exists such that \( \{|C \in P_j | Pr \models C| > \{|C \in P_j | Pr' \models C| \} \) and \( \{|C \in P_j | Pr \models C| = \{|C \in P_j | Pr' \models C| \} \) for all \( i < j \leq k \). A model \( Pr \) of a set of logical constraints and probabilistic formulas \( \mathcal{F} \) is a \( lex_\lambda \)-minimal model of \( \mathcal{F} \) iff no model of \( \mathcal{F} \) is \( lex_\lambda \)-preferable to \( Pr \).

We now define the notion of \( lex_\lambda \)-entailment as follows. A conditional constraint \( C = (\psi | \phi)[l, u] \) is a \( lex_\lambda \)-consequence of \( KB \), denoted \( KB \models^lex_\lambda C \), iff every \( lex_\lambda \)-minimal model of \( L \cup \{ \phi \models \lambda \} \) satisfies \( C \).
It is a tight $\lambda$-consequence of $KB$, denoted $KB \models^T_{\lambda} C$, iff $l = \inf Pr(\psi|\phi)$ (resp., $u = \sup Pr(\psi|\phi)$) subject to all $\lambda$-minimal models $Pr$ of $L \cup \{\phi \models \lambda\}$.

The following example shows that the notion of $\lambda$-entailment realizes an inheritance of default properties, without showing the problem of inheritance blocking.

**Example 3.4** Let $KB = (L, P)$ be as in Example 2.1. Some tight $\lambda$-consequences, where $\lambda \in \{0, 0.2, 0.4, 0.6, 0.8, 1\}$, are shown in Table 3. In particular, for every strength $\lambda \in [0, 1]$, the default logical property of having legs is inherited from the class of birds to the exceptional subclass of penguins, while the default purely probabilistic property of being easy to see with a probability between 0.8 and 0.9 is also inherited from the class of yellow objects to the exceptional subclass of yellow penguins.

**Table 3: Some tight $\lambda$-consequences.**

<table>
<thead>
<tr>
<th></th>
<th>Strength $\lambda$</th>
<th>0</th>
<th>0.2</th>
<th>0.4</th>
<th>0.6</th>
<th>0.8</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>legs</td>
<td>bird</td>
<td>[1,1]</td>
<td>[1,1]</td>
<td>[1,1]</td>
<td>[1,1]</td>
<td>[1,1]</td>
<td>[1,1]</td>
</tr>
<tr>
<td>legs</td>
<td>yellow $\land$ bird</td>
<td>[1,1]</td>
<td>[1,1]</td>
<td>[1,1]</td>
<td>[1,1]</td>
<td>[1,1]</td>
<td>[1,1]</td>
</tr>
<tr>
<td>legs</td>
<td>penguin</td>
<td>[1,1]</td>
<td>[1,1]</td>
<td>[1,1]</td>
<td>[1,1]</td>
<td>[1,1]</td>
<td>[1,1]</td>
</tr>
<tr>
<td>legs</td>
<td>yellow $\land$ penguin</td>
<td>[1,1]</td>
<td>[1,1]</td>
<td>[1,1]</td>
<td>[1,1]</td>
<td>[1,1]</td>
<td>[1,1]</td>
</tr>
<tr>
<td>fly</td>
<td>bird</td>
<td>[.9,1]</td>
<td>[.9,1]</td>
<td>[.9,1]</td>
<td>[.9,1]</td>
<td>[.9,1]</td>
<td>[.9,1]</td>
</tr>
<tr>
<td>fly</td>
<td>yellow $\land$ bird</td>
<td>[.0,1]</td>
<td>[.5,1]</td>
<td>[.75,1]</td>
<td>[.83,1]</td>
<td>[.88,1]</td>
<td>[.9,1]</td>
</tr>
<tr>
<td>fly</td>
<td>penguin</td>
<td>[.0,1]</td>
<td>[.0,1]</td>
<td>[.0,1]</td>
<td>[.0,1]</td>
<td>[.0,1]</td>
<td>[.0,1]</td>
</tr>
<tr>
<td>fly</td>
<td>yellow $\land$ penguin</td>
<td>[.0,1]</td>
<td>[.0,5]</td>
<td>[.0,25]</td>
<td>[.0,17]</td>
<td>[.0,13]</td>
<td>[.0,1]</td>
</tr>
<tr>
<td>see</td>
<td>yellow $\land$ bird</td>
<td>[.0,1]</td>
<td>[.5,1]</td>
<td>[.67,1]</td>
<td>[.75,1]</td>
<td>[.8,.9]</td>
<td>[.8,.9]</td>
</tr>
<tr>
<td>see</td>
<td>yellow $\land$ penguin</td>
<td>[.0,1]</td>
<td>[.0,1]</td>
<td>[.5,1]</td>
<td>[.67,1]</td>
<td>[.75,1]</td>
<td>[.8,.9]</td>
</tr>
</tbody>
</table>

**4 Semantic Properties**

In this section, we briefly summarize some semantic properties of $\lambda$-logical, $z_\lambda$-, and $\lambda$-entailment.

**4.1 General Nonmonotonic Properties**

The notions of $\lambda$-logical, $z_\lambda$-, and $\lambda$-entailment all satisfy probabilistic versions of the postulates Right Weakening, Reflexivity, Left Logical Equivalence, Cut, Cautious Monotonicity, and Or proposed by Kraus.
et al. [35], which are commonly regarded as being particularly desirable for any reasonable notion of non-monotonic entailment [48, 49]. Furthermore, all three notions satisfy the desirable property of *Rational Monotonicity* [35], which describes a restricted form of monotony and allows to ignore certain kinds of irrelevant knowledge.

### 4.2 Relationships between Probabilistic Formalisms

As for the relationships between the three formalisms, it holds that $\lambda$-logical entailment is stronger than both $\text{lex}_\lambda$- and $z_\lambda$-entailment. Moreover, $\text{lex}_\lambda$-entailment is stronger than $z_\lambda$-entailment. These relationships between $\lambda$-logical, $z_\lambda$-, and $\text{lex}_\lambda$-entailment are illustrated in Fig. 1.

In general, $\lambda$-logical entailment is strictly stronger than $\text{lex}_\lambda$-entailment, which in turn is strictly stronger than $z_\lambda$-entailment. However, in the special case when $\phi = \top$, the three notions of $\lambda$-logical, $z_\lambda$-, and $\text{lex}_\lambda$-entailment of $(\psi \mid \phi)[l, u]$ from $\lambda$-consistent $KB = (L, P)$ all coincide. Furthermore, also when $L \cup P \cup \{\phi \succcurlyeq \lambda\}$ is satisfiable, the three notions of $\lambda$-logical, $z_\lambda$-, and $\text{lex}_\lambda$-entailment of $(\psi \mid \phi)[l, u]$ from $\lambda$-consistent $KB = (L, P)$ all coincide.

### 4.3 Probabilistic and Classical Special Cases

For $\lambda = 0$, the notion of $\lambda$-logical entailment from $KB$ coincides with standard logical entailment from $KB$. For $\lambda = 0$ (resp., $\lambda = 1$), the notions of $z_\lambda$- and $\text{lex}_\lambda$-entailment coincide with weak (resp., strong) probabilistic $z$- and $\text{lex}$-entailment introduced in [50] (resp., [47]). Furthermore, for $\lambda = 0$, the notion of $\lambda$-consistency coincides with the notion of g-coherence (see e.g. [12]).

As for classical special cases, $z_\lambda$- and $\text{lex}_\lambda$-entailment of $(\beta \mid \alpha)[1, 1]$ from $\lambda$-consistent probabilistic knowledge bases of the form $KB = (L, P)$, where $P = \{(\psi_i \mid \phi_i)[1, 1] \mid i \in \{1, \ldots, n\}\}$, coincide with the classical notions of Pearl’s entailment in System Z and Lehmann’s lexicographic entailment of the default $\beta \leftarrow \alpha$ from the default counterpart of $KB$. Furthermore, $\lambda$-logical entailment of $(\beta \mid \alpha)[1, 1]$ from such $KB$ coincides with propositional logical entailment of $\beta \leftarrow \alpha$ from the propositional counterpart of $KB$ (see Fig. 1). Finally, the notion of $\lambda$-consistency for such $KB$ coincides with the notion of $\varepsilon$- (or also $p$-) consistency for the default counterpart of $KB$.

### 4.4 Relationship to G-Coherent Entailment

Similarly to $z_\lambda$- and $\text{lex}_\lambda$-entailment, one can also define a probabilistic generalization of entailment in System $P$ of strength $\lambda \in [0, 1]$, called $p_\lambda$-entailment, which is strictly weaker than $z_\lambda$-entailment (see Fig. 1). However, since entailment in System $P$ does not realize a general property inheritance along subclass relationships, also $p_\lambda$-entailment does not have such an inheritance, and in particular generally does not depend on $\lambda$ (see Table 4, which shows some tight $p_\lambda$-consequences from $KB$ of Example 2.1). For $\lambda = 0$, this notion of $p_\lambda$-entailment coincides with the notion of g-coherent entailment (e.g. [12]).

### 5 Further Examples

In reasoning from statistical knowledge and degrees of belief, $z_1$- and $\text{lex}_1$-entailment show a similar behavior as reference-class reasoning [55, 36, 37, 54] in a number of uncontroversial examples. But, they also avoid many drawbacks of reference-class reasoning. In detail, they can handle complex scenarios and even purely probabilistic subjective knowledge as input. Moreover, conclusions are drawn in a global way from
all the available knowledge as a whole. See [47] for further details. The following example illustrates the use of \(\text{lex}_1\)-entailment for reasoning from statistical knowledge and degrees of belief.

**Example 5.1** Suppose that we have the statistical knowledge “all penguins are birds”, “between 90% and 95% of all birds fly”, “at most 5% of all penguins fly”, and “at least 95% of all yellow objects are easy to see”. Moreover, assume that we believe “Sam is a yellow penguin”. What do we then conclude about Sam’s property of being easy to see? Under reference-class reasoning, which is a machinery for dealing with statistical knowledge and degrees of belief, we conclude “Sam is easy to see with a probability of at least 0.95”. This is exactly what we obtain using the notion of \(\text{lex}_1\)-entailment. More precisely, the above statistical knowledge can be represented by the following probabilistic knowledge base \(KB = (L, P)\):

\[
L = \{\text{bird} \iff \text{penguin}\},
\]

\[
P = \{(\text{fly}\mid\text{bird})\mid[0.9,0.95], (\text{fly}\mid\text{penguin})\mid[0.05], (\text{see}\mid\text{yellow})\mid[0.95,1]\}\).
\]

It is then not difficult to verify that \(KB\) is 1-consistent, and that \((\text{see}\mid\text{yellow}\land\text{penguin})\mid[0.95,1]\) is a tight conclusion from \(KB\) under \(\text{lex}_1\)-entailment. Some other tight intervals for \(\text{see}\mid\text{yellow}\land\text{penguin}\) from \(KB\) under \(\lambda\)-logical, \(\text{lex}_\lambda\)-, \(\text{z}_{\lambda}\)-, \(\text{p}_{\lambda}\)-entailment are shown in Table 5.

**Table 5:** Tight intervals for \(\text{see}\mid\text{yellow}\land\text{penguin}\).

<table>
<thead>
<tr>
<th>Strength (\lambda)</th>
<th>0</th>
<th>0.2</th>
<th>0.4</th>
<th>0.6</th>
<th>0.8</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\models^{\lambda}_{\text{tight}})</td>
<td>[0,1]</td>
<td>[0,1]</td>
<td>[0,1]</td>
<td>[0,1]</td>
<td>[0,1]</td>
<td></td>
</tr>
<tr>
<td>(\models_{\text{lex}_\lambda})</td>
<td>[0,1]</td>
<td>[0.75,1]</td>
<td>[0.88,1]</td>
<td>[0.92,1]</td>
<td>[0.94,1]</td>
<td>[0.95,1]</td>
</tr>
<tr>
<td>(\models_{\text{z}_{\lambda}})</td>
<td>[0,1]</td>
<td>[0.1]</td>
<td>[0.1]</td>
<td>[0.1]</td>
<td>[0.1]</td>
<td></td>
</tr>
<tr>
<td>(\models_{\text{p}_{\lambda}})</td>
<td>[0,1]</td>
<td>[0,1]</td>
<td>[0,1]</td>
<td>[0,1]</td>
<td>[0,1]</td>
<td></td>
</tr>
<tr>
<td>(\models^{\lambda}_{\text{tight}})</td>
<td>[0,1]</td>
<td>[0,1]</td>
<td>[0,1]</td>
<td>[0,1]</td>
<td>[0,1]</td>
<td></td>
</tr>
</tbody>
</table>

The next example is from the area of medical diagnosis.
Example 5.2 In a hospital, physicians have to diagnose whether patients with acute abdominal pain are suffering from appendicitis or not. Diagnosing appendicitis is a difficult task, since a lot of different symptoms (as e.g. high temperature, a high rate of leucocytes, vomiting, and various types of pains) can indicate appendicitis, but often only the joint occurrence of several of these symptoms reliably supports the diagnosis. Here, we only consider four possible symptoms of appendicitis (app), namely a high rate of leucocytes (leuco_high) and the following three different types of pain: rectal pain (rec_pain), pain when released (pain_rel), and rebound tenderness (reb_tender). Thus, our view on this area is a very simplified one. Let our knowledge about the relationships between app, leuco_high, and the three types of pain be expressed by the following probabilistic knowledge base $KB = (L, P)$, where $L = \emptyset$ and $P$ is given as follows:

$$P = \{(\text{reb}_\text{tender} \mid \text{pain}_\text{rel})[.70, .75],
\quad (\text{reb}_\text{tender} \mid \text{leuco}_\text{high})[.70, .75],
\quad (\text{app} \mid \text{rec}_\text{pain} \land \text{pain}_\text{rel})[.70, .75],
\quad (\text{app} \mid \text{rec}_\text{pain} \land \text{reb}_\text{tender})[.65, .70],
\quad (\text{app} \mid \text{pain}_\text{rel} \land \text{reb}_\text{tender} \land \text{leuco}_\text{high})[.80, .85]\}.$$  

Suppose now that Judy is a patient showing the symptoms leuco_high and pain_rel. Which is the probability that Judy has appendicitis? Which is the probability that she has appendicitis given that she also feels rectal pain? Some tight intervals for app|leuco_high \land pain_rel and app|leuco_high \land pain_rel \land rec_pain, respectively, from $KB$ under $\lambda$-logical, $z_\lambda$, $lex_\lambda$, and $p_\lambda$-entailment are shown in Tables 6 and 7, respectively.

<table>
<thead>
<tr>
<th>Strength $\lambda$</th>
<th>0</th>
<th>0.2</th>
<th>0.4</th>
<th>0.6</th>
<th>0.8</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>$||^\lambda_{\text{tight}}$</td>
<td>[0, 1]</td>
<td>[.08, .99]</td>
<td>[.38, .93]</td>
<td>[.48, .91]</td>
<td>[.53, .9]</td>
<td>[.56, .9]</td>
</tr>
<tr>
<td>$||^\text{lex}<em>\lambda</em>{\text{tight}}$</td>
<td>[0, 1]</td>
<td>[.08, .99]</td>
<td>[.38, .93]</td>
<td>[.48, .91]</td>
<td>[.53, .9]</td>
<td>[.56, .9]</td>
</tr>
<tr>
<td>$||^z\lambda_{\text{tight}}$</td>
<td>[0, 1]</td>
<td>[.08, .99]</td>
<td>[.38, .93]</td>
<td>[.48, .91]</td>
<td>[.53, .9]</td>
<td>[.56, .9]</td>
</tr>
<tr>
<td>$||^p\lambda_{\text{tight}}$</td>
<td>[0, 1]</td>
<td>[0, 1]</td>
<td>[0, 1]</td>
<td>[0, 1]</td>
<td>[0, 1]</td>
<td>[0, 1]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Strength $\lambda$</th>
<th>0</th>
<th>0.2</th>
<th>0.4</th>
<th>0.6</th>
<th>0.8</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>$||^\lambda_{\text{tight}}$</td>
<td>[0, 1]</td>
<td>[0, 1]</td>
<td>[.41, 1]</td>
<td>[.57, 1]</td>
<td>[.66, .92]</td>
<td>[1, 0]</td>
</tr>
<tr>
<td>$||^\text{lex}<em>\lambda</em>{\text{tight}}$</td>
<td>[0, 1]</td>
<td>[0, 1]</td>
<td>[.41, 1]</td>
<td>[.57, 1]</td>
<td>[.66, .92]</td>
<td>[.7, .75]</td>
</tr>
<tr>
<td>$||^z\lambda_{\text{tight}}$</td>
<td>[0, 1]</td>
<td>[0, 1]</td>
<td>[.41, 1]</td>
<td>[.57, 1]</td>
<td>[.66, .92]</td>
<td>[0, 1]</td>
</tr>
<tr>
<td>$||^p\lambda_{\text{tight}}$</td>
<td>[0, 1]</td>
<td>[0, 1]</td>
<td>[0, 1]</td>
<td>[0, 1]</td>
<td>[0, 1]</td>
<td>[0, 1]</td>
</tr>
</tbody>
</table>

6 Algorithms

Algorithm consistency in Fig. 2 decides whether a given probabilistic knowledge base $KB = (L, P)$ is $\lambda$-consistent. If $KB$ is $\lambda$-consistent, then consistency also returns the $z_\lambda$-partition of $KB$. It is similar to an algorithm for deciding g-coherence by Biazzo et al. [11], which in turn is a probabilistic generalization of an
Algorithm consistency
Input: probabilistic knowledge base $KB = (L, P)$ and strength $\lambda$.
Output: $S_\lambda$-partition of $KB$, if $KB$ is $\lambda$-consistent; nil otherwise.
1. if $P = \emptyset$ then if $L$ is satisfiable then return $()$ else return nil;
2. $R := P$;
3. $i := -1$;
4. repeat
5. $i := i + 1$;
6. $D[i] := \{(\psi|\phi)[l, u] \in R \mid L \cup R \cup \{\phi \models \lambda\} \text{ is satisfiable}\}$;
7. $R := R \setminus D[i]$;
8. until $R = \emptyset$ or $D[i] = \emptyset$;
9. if $R = \emptyset$ then return $(D[0], \ldots, D[i])$ else return nil.

Figure 2: Algorithm consistency

Algorithm tight-z-consequence
Input: probabilistic knowledge base $KB = (L, P)$, conditional event $\beta|\alpha$, and strength $\lambda$.
Output: interval $[l, u] \subseteq [0, 1]$ such that $KB \models \lambda^{\alpha\beta}(\beta|\alpha)[l, u]$.
1. $\mathcal{P} := \text{consistency}(KB, \lambda)$;
2. if $P = \text{nil}$ then return $[1, 0]$;
3. $(P_0, \ldots, P_k) := \mathcal{P}$; $R := L$;
4. if $R \cup \{\alpha \models \lambda\}$ is unsatisfiable then return $[1, 0]$;
5. $j := k$;
6. while $j \geq 0$ and $R \cup P_j \cup \{\alpha \models \lambda\}$ is satisfiable do begin
7. $R := R \cup P_j$;
8. $j := j - 1$;
9. end;
10. compute $l, u \in [0, 1]$ s. t. $R \cup \{\alpha \models \lambda\} \models \lambda^{\alpha\beta}(\beta|\alpha)[l, u]$;
11. return $[l, u]$.

Figure 3: Algorithm tight-z-consequence

Algorithm tight-lex-consequence
Input: probabilistic knowledge base $KB = (L, P)$, conditional event $\beta|\alpha$, and strength $\lambda$.
Output: interval $[l, u] \subseteq [0, 1]$ such that $KB \models \lambda^{\alpha\beta}(\beta|\alpha)[l, u]$.
1. $\mathcal{P} := \text{consistency}(KB, \lambda)$;
2. if $P = \text{nil}$ then return $[1, 0]$;
3. $(P_0, \ldots, P_k) := \mathcal{P}$; $R := L$;
4. if $R \cup \{\alpha \models \lambda\}$ is unsatisfiable then return $[1, 0]$;
5. $\mathcal{H} := \{\emptyset\}$;
6. for $j := k$ downto 0 do begin
7. $n := 0$;
8. $\mathcal{H}' := \emptyset$;
9. for each $G \subseteq P_j$ and $H \in \mathcal{H}$ do
10. if $R \cup G \cup H \cup \{\alpha \models \lambda\}$ is satisfiable then
11. if $n = |G|$ then $\mathcal{H}' := \mathcal{H}' \cup \{G \cup H\}$
12. else if $n < |G|$ then begin
13. $\mathcal{H}' := \{G \cup H\}$;
14. $n := |G|$;
15. end;
16. $\mathcal{H} := \mathcal{H}'$;
17. end;
18. $(l, u) := (1, 0)$;
19. for each $H \in \mathcal{H}$ do begin
20. compute $c, d \in [0, 1]$ s. t. $R \cup H \cup \{\alpha \models \lambda\} \models \lambda^{\alpha\beta}(\beta|\alpha)[c, d]$;
21. $(l, u) := (\min(l, c), \max(u, d))$;
22. end;
23. return $[l, u]$.

Figure 4: Algorithm tight-lex-consequence
algorithm for deciding \( \varepsilon \)-consistency in default reasoning [31]. More precisely, if \( P = \emptyset \), then Step 1 returns the empty partition \( \emptyset \), if \( L \) is satisfiable; and \texttt{nil}, otherwise. If \( P \neq \emptyset \), then Steps 2–8 try to compute the \( z_\lambda \)-partition \( \mathcal{P} \) of \( KB \), and Step 9 returns \( \mathcal{P} \), if this succeeds; and \texttt{nil}, otherwise.

Algorithms tight-s-consequence, with \( s = z \) and \( s = \text{lex} \), in Figs. 3 and 4 compute tight intervals under \( z_\lambda \)- and \text{lex}_\lambda\-entailment from \( KB = (L, P) \), respectively. They are similar to algorithms from [50] for computing tight entailed intervals under weak probabilistic \( z \)- and \text{lex}\-entailment, respectively. Similarly to algorithms for lexicographic inference in [8], they are based on a compilation step. More precisely, if \( KB \) is not \( \lambda \)-consistent, then \([1, 0]\) is returned in Step 2. If \( KB \) is \( \lambda \)-consistent, and \( L \cup \{ \alpha \succ \lambda \} \) is unsatisfiable, then \([1, 0]\) is returned in Step 4. Otherwise (that is, \( KB \) has a \( z_\lambda \)-partition \( \{P_0, \ldots, P_k\} \), and \( L \cup \{ \alpha \succ \lambda \} \) is satisfiable), we use the following Theorem 6.1 saying that then a set \( \mathcal{D}_\alpha^{s}(KB) \subseteq 2^P \), \( s \in \{ z_\lambda, \text{lex}_\lambda \} \), exists such that \( KB \models^s (\beta|\alpha)[l, u] \) iff \( L \cup H \cup \{ \alpha \succ \lambda \} \models (\beta|\alpha)[l, u] \) for all \( H \in \mathcal{D}_\alpha^{s}(KB) \).

In this case, we compute \( \mathcal{D}_\alpha^{s}(KB) \) along the \( z_\lambda \)-partition of \( KB \) in Steps 5–9 (resp., 5–17), and the requested tight interval in Step 10 (resp., 18–22).

For \( G, H \subseteq P \), we say \( G \) is \( z_\lambda \)-preferable to \( H \) iff some \( i \in \{0, \ldots, k\} \) exists such that \( P_i \subseteq G, P_i \nsubseteq H \), and \( P_j \subseteq G \) and \( P_j \not\subseteq H \) for all \( i < j \leq k \). We say \( G \) is \text{lex}_\lambda\-preferable to \( H \) iff some \( i \in \{0, \ldots, k\} \) exists such that \( |G \cap P_i| > |H \cap P_i| \) and \( |G \cap P_j| = |H \cap P_j| \) for all \( i < j \leq k \). For \( D \subseteq 2^P \) and \( s \in \{ z_\lambda, \text{lex}_\lambda \} \), we say \( G \) is \( s \)-minimal in \( D \) iff \( G \in D \) and no \( H \in D \) is \( s \)-preferable to \( G \).

**Theorem 6.1** Let \( KB = (L, P) \) be \( \lambda \)-consistent, \( \beta|\alpha \) a conditional event, and \( L \cup \{ \alpha \succ \lambda \} \) be satisfiable. Let \( s \in \{ z_\lambda, \text{lex}_\lambda \} \) and \( \mathcal{D}_\alpha^{s}(KB) \) be the set of all \( s \)-minimal elements in \( \{ H \subseteq P \mid L \cup H \cup \{ \alpha \succ \lambda \} \) is satisfiable\}. Then, \( l \) (resp., \( u \)) such that \( KB \models^s (\beta|\alpha)[l, u] \) is given by \( l = \min c \) (resp., \( u = \max d \)) subject to \( L \cup H \cup \{ \alpha \succ \lambda \} \models^\text{tight} (\beta|\alpha)[c, d] \) and \( H \in \mathcal{D}_\alpha^{s}(KB) \).

The above three algorithms are based on reductions to (i) deciding whether a given \( KB = (L, P) \) has a model \( Pr \) such that \( Pr(\alpha) > 0 \) for a given event \( \alpha \), and to (ii) computing tight logically entailed intervals from a given \( KB \) for a given conditional event \( \beta|\alpha \). The number of tasks (i) and (ii) to be solved in the first two algorithms (resp., the third algorithm) is in \( O(|P|^2) \) (resp., \( O(2^{|P|}) \)). The task (i) can be reduced to deciding whether a system of linear constraints is solvable, while (ii) can be reduced to computing the optimal values of two linear programs. These two well-known results are summarized as follows.

**Theorem 6.2** Let \( KB = (L, P) \) be a probabilistic knowledge base, and \( \alpha, \beta \) be events. Let \( R = \{ I \in I_\Phi \mid I \models L \} \). Let \( LC \) denote the system of linear constraints in Fig. 5 over the variables \( y_r \) \((r \in R)\). Then, (a) \( L \cup P \) has a model \( Pr \) such that \( Pr(\alpha) > 0 \) iff \( LC \) is solvable. (b) If \( L \cup P \) has a model \( Pr \) such that \( Pr(\alpha) > 0 \), then \( l \) (resp., \( u \)) such that \( L \cup P \models^\text{tight} (\beta|\alpha)[l, u] \) is the optimal value of the following linear program over the variables \( y_r \) \((r \in R)\):

\[
\text{minimize (resp., maximize) } \sum_{r \in R} y_r \text{ subject to } LC.
\]

7 The System NMPROBLOG

The system NMPROBLOG is written in the programming language C, and uses \texttt{lp.solver 5.1} for deciding the solvability of systems of linear constraints and for computing the optimal values of linear programs. The graphical user interface (GUI) of NMPROBLOG has been built using \texttt{glade 2.6}”. Its main components are the main window, one window each for checking satisfiability, for checking \( \lambda \)-consistency, and for computing
Consider again the probabilistic knowledge base $KB$. Fig. 11 shows the time used by NMPROBLOG to load from a file with suffix ".tax" a set of statements of the form "##", and "#" to declare the basic events in $KB$, and to express the logical constraints in $KB$. The system NMPROBLOG loads from a file with suffix ".tax" a set of statements of the following forms: (i) $p = 1$, where $p$ is a nonempty string, to declare $p$ as $\top$, (ii) $p = 0$, where $p$ is a nonempty string, to declare $p$ as $\bot$, (iii) $p < 1$, where $p$ is a nonempty string, to declare $p$ as a basic event, and (iv) $\psi > \phi$, where $\psi$ and $\phi$ are events (in which "<", "&", and "#" encode $\neg$, $\land$, and $\lor$, respectively), to express that $\phi$ implies $\psi$. Furthermore, it then loads from a file with suffix ".prb" a set of statements of the form "## $\psi$ $\phi$ $l$ $u$", where $\psi$ and $\phi$ are events as above, and $l$ and $u$ are real numbers, to encode the conditional constraint $(\psi|\phi)[l, u]$. Figure 5: System of linear constraints.

\[
\sum_{r \models \psi \land \phi} u y_r + \sum_{r \models \psi \land \phi} (u-1) y_r \geq 0 \quad \text{for all } (\psi|\phi)[l, u] \in P, u < 1
\]

\[
\sum_{r \models \neg \psi \land \phi} y_r = 1
\]

\[
\sum_{r \models \neg \psi \land \phi} y_r \geq 0 \quad \text{for all } r \in R
\]

\[
\sum_{r \models \psi \land \phi} -l y_r + \sum_{r \models \psi \land \phi} (1-l) y_r \geq 0 \quad \text{for all } (\psi|\phi)[l, u] \in P, l > 0
\]

Note that every basic event in the ".prb"-file and in queries (window for computing tight entailed intervals) must be declared in the ".tax"-file.

**Example 7.1** Consider again the probabilistic knowledge base $KB = (L, P)$ given in Example 2.1. The ".tax"-file contains the statements $1 > bird$, $1 > penguin$, $1 > fly$, $1 > legs$, $1 > see$, $1 > yellow$, and $bird > penguin$ to declare the basic events in $KB$ and to express the logical constraints in $L$. The ".prb"-file contains the statements $legs$ $bird$ $1.0$ $1.0$, $see$ $yellow$ $0.8$ $0.9$, $fly$ $bird$ $0.9$ $1.0$, and $fly$ $penguin$ $0.0$ $0.1$ to express the conditional constraints in $P$. After reading the ".tax"- and the ".prb"-file, one can open the window for computing tight consequences in Fig. 10 and, for example, compute the tight interval $[l, u]$ such that $KB \models^{\text{lex}_\lambda}_{\text{tight}} (\text{see} \land \text{bird})[l, u]$, $\lambda = 0.5$, which is given by $[l, u] = [0.6, 1]$ (see Fig. 10).

**Example 7.2** Fig. 11 shows the time used by NMPROBLOG on a chain of $n$ correlated basic events ($2^n$ variables and $4(n-1)+1$ constraints in the generated linear optimization problems) for checking satisfiability and $\lambda$-consistency, as well as computing the $z_\lambda$-partition and tight entailed intervals under $\lambda$-logical, $z_\lambda$, $\text{lex}_\lambda$, and $p_\lambda$-entailment. Here, all the above reasoning tasks can be solved in few minutes, even when large linear optimization problems are generated (up to 16384 variables and 53 linear constraints).

\section{Summary and Outlook}

I have recalled nonmonotonic probabilistic logics under variable-strength inheritance with overriding, namely, the notions of $z_\lambda$- and $\text{lex}_\lambda$-entailment, along with their semantic properties and some new examples. I have presented algorithms for deciding $\lambda$-consistency and for computing tight entailed intervals under $z_\lambda$ and $\text{lex}_\lambda$-entailment, which are based on reductions to the problems of deciding satisfiability and of computing tight logically entailed intervals in model-theoretic probabilistic logic.

Furthermore, I have presented the system NMPROBLOG (available at http://www.kr.tuwien.ac.at/staff/lukasiew/nmproblog.tar.gz), which comprises an implementation of these algorithms. NMPROBLOG allows for (i) checking the satisfiability of probabilistic knowledge bases $KB$,
Figure 6: Main window of NMPROBLOG.

Figure 7: Window for checking satisfiability.

Figure 8: Window for checking \( \lambda \)-consistency.
Figure 9: Window for computing the $z_{\lambda}$-partition.

Figure 10: Window for computing tight entailed intervals.
(ii) checking the $\lambda$-consistency of $KB$, and (iii) computing the $z_\lambda$-partition of $KB$, as well as (iv) computing tight entailed intervals under any among $\lambda$-logical, $\text{lex}_\lambda$, $z_\lambda$, and $p_\lambda$-entailment, for any strength $\lambda \in \{i/100 \mid i \in \{0, \ldots, 100\}\}$. In particular, it thus also allows for probabilistic and default reasoning in all the special cases of $\lambda$-logical, $\text{lex}_\lambda$, $z_\lambda$, and $p_\lambda$-entailment that are summarized in Section 4.

A topic of future research is to explore whether there are techniques for more efficient or even tractable inference in nonmonotonic probabilistic logics under variable-strength inheritance with overriding (e.g., along the lines of [11] and [19]), and to eventually include them into NMPROBLOG.

References


29. GLADE GTK+ User Interface Builder (version 2.6.8). Available at http://glade.gnome.org/.


