Comparing and Evaluating Approaches to Probabilistic Reasoning: Theory, Implementation, and Applications*

Gabriele Kern-Isberner[1], Christoph Beierle[2],
Marc Finthammer[2], Matthias Thimm[3]

[1]Dept. of Computer Science, TU Dortmund, 44221 Dortmund, Germany
[2]Dept. of Computer Science, FernUniversität in Hagen, 58084 Hagen, Germany
[3]Dept. of Computer Science, Universität Koblenz, 56070 Koblenz, Germany

Abstract. The handling of uncertain information is of crucial importance for the success of expert systems. This paper gives an overview on logic-based approaches to probabilistic reasoning and goes into more details about recent developments for relational, respectively first-order, probabilistic methods like Markov logic networks, and Bayesian logic programs. In particular, we feature the maximum entropy approach as a powerful and elegant method that combines convenience with respect to knowledge representation with excellent inference properties. While comparing the different approaches is a difficult task due to the variety of the available concepts and to the absence of a common interface, we address this problem from both a conceptual and practical point of view. On a conceptual layer we propose and discuss several criteria by which first-order probabilistic methods can be distinguished, and apply these criteria to a series of approaches. On the practical layer, we briefly describe some systems for probabilistic reasoning, and go into more details on the KReator system as a versatile toolbox for various approaches to first-order probabilistic relational learning, modelling, and reasoning. Moreover, we illustrate applications of probabilistic logics in various scenarios.

1 Introduction

Real world applications of expert systems (and other computational systems, too) usually have to struggle with the problem that both background knowledge and information on the situation at hand are neither complete nor certain. For instance, in a medical domain, the physician may know that most patients suffering from appendicitis also complain about abdominal pain, but in some cases, the patients show other atypical symptoms; however, these relationships cannot be further specified in a satisfactory way. In the special case of the patient the physician is just facing, she is not even sure whether he feels abdominal pain as he is a boy of three years of age.

* The research reported here was partially supported by the Deutsche Forschungsgemeinschaft (grants KE 1413/2-2 and BE 1700/7-2).

Probabilistic logics offer a rich framework to represent and process uncertain information (for foundational work in this area see [56, 17]), and are linked to statistics and machine learning in a natural way. Knowledge can be extracted from data, expressed in a suitable probabilistic formalism like Bayesian networks [56], and used for uncertain reasoning by applying inference mechanisms. Completeness of knowledge can be achieved by presupposing additional assumptions like conditional independence of variables, like in most probabilistic networks [56], or by making use of the information-theoretical principles of optimum entropy [36]. In both ways, a full probability distribution is generated from partial knowledge, on the base of which probabilities for arbitrary queries can be computed.

Most of the standard probabilistic approaches applied today make use of some type of probabilistic networks and propositional logic. While network techniques are of major importance to allow for local computations, the restriction to propositional logic makes probabilistic knowledge representation inadequate for domains in which relationships between objects are in the focus of investigation. However, generalising established propositional probabilistic methods to first-order knowledge representation turns out to not be an easy task, as the complexity of knowledge representation raises substantially, so that new inference techniques have to be devised. Moreover, the probabilistic semantics of open formulas is not at all clear. For example, the following conditional probabilistic formulas express commonsense knowledge about the relationships between elephants and their keepers which are usually good (elephants like their keepers), but also take exceptions into regard—elephants tend not to like keeper *fred*, except for the good natured elephant *clyde*:

$$(\mathit{likes}(X, Y) \mid \mathit{elephant}(X),\ \mathit{keeper}(Y))\ [0.8]$$
$$(\mathit{likes}(X, \mathsf{fred}) \mid \mathit{elephant}(X))\ [0.3]$$
$$\mathit{likes}(\mathsf{clyde}, \mathsf{fred})[0.9]$$

A schematic grounding of all rules of this knowledge base would cause conflicts with respect to elephant *clyde* and keeper *fred*. Moreover, both statistical (or population-based, respectively) information and subjective views are addressed, as the first two formulas involve all elephants (and keepers), while the third one only considers situations involving *clyde* and *fred*.

Recently, the fields of *probabilistic inductive logic programming* and *statistical relational learning* have put forth a lot of proposals that deal with combining traditional probabilistic models of knowledge like Bayes nets or Markov nets [56] with first-order logic, see [13, 29] for some excellent overviews. This area, located at the intersection of logic, probability theory, and machine learning, investigates methods for representing probabilistic information in a relational context for both reasoning and learning. Many researchers developed liftings of propositional probabilistic models to the first-order case in order to take advantage of methods and algorithms already developed. Among these are the well-known Bayesian logic programs [43] and Markov logic networks [59] which extend Bayes nets and Markov networks [56], respectively, and are based on knowledge-

based model construction [68, 7]. Other approaches also employ Bayes nets for their theoretical foundation, like logical Bayesian networks [18] and relational Bayesian networks [31, 10], or they are influenced by other fields of research like probabilistic relational models [27] by database theory, P-log [1] by answer set programming, and ProbLog [58] by Prolog. The aforementioned approaches are representatives of a vast variety of different approaches having been developed in the past ten to twenty years, and we refer to [13, 11] for a more elaborate discussion of existing approaches and their history.

As a first focal point of this paper, we will concentrate on extensions of the maximum entropy principle to a relational setting. There are few approaches applying maximum entropy methods beyond propositional logic, cf. [47, 41, 24]. We will give an introduction to different relational maximum entropy proposals, presenting in particular the *grounding*, the *aggregating*, and the *averaging* semantics described in [47, 41, 24].

Although hard computational problems, challenging theoretical questions, and many interesting new applications like social networks are associated with this area of research, providing great motivation in developing new approaches for statistical relational learning that deal with specific scenarios, thorough comparisons of approaches are rare. This is no surprise as many approaches build on different logics and employ different methods of propositional probabilistic reasoning methods. There are some papers that evaluate approaches with respect to specific formal criteria. For example, in [32] Jaeger introduces *model-theoretic expressivity analysis* in order to compare the expressive power of different approaches to statistical relational learning. In that paper, it is only shown that relational Bayesian networks [31, 10] are at least as expressive as Markov logic networks [59] within that particular framework. Furthermore, it is conjectured [32] that Bayesian logic programs are equally expressive as relational Bayesian networks. A similar approach is pursued in [49] where it has been shown that Bayesian logic programs and an extension of stochastic logic programs [50] are of equal expressive power regarding some alternative definition of expressivity than the one used in [32]. Furthermore, there are some other attempts to compare approaches to statistical relational learning that focus more on comparisons of implementations like [44], but besides [32] no formal knowledge representation criteria exist to date for such comparisons.

As another main contribution of this paper, we develop a series of evaluation and comparison criteria that aim at characterizing the characteristics of individual approaches to probabilistic relational knowledge representation and provide better means to understand their relationships. The criteria are established from a knowledge representation and reasoning point of view and address various themes, covering language aspects, the dimensions of strict and propositional knowledge, and the role of individuals and universes. In line with the papers's focus on maximum entropy based methods, we will apply these criteria to the three relational maximum entropy semantics mentioned above. Furthermore, we will use the criteria also with respect to Bayesian logic programs and Markov logic networks, two of the most popular approaches from statistical relational

learning. A common principle of these five probabilistic relational techniques treated in this paper is that they all provide a form of inductive completion of the explicitly given knowledge. In future work, we will extend our comparison also to other logic-based probabilistic formalisms.

Besides criteria for comparing relational probabilistic approaches, applying methods to benchmark examples is important for the purpose of comparing and evaluating. However, even seemingly small examples need to be computed by a machine, due to the size explosion caused by grounding, and each of these approaches comes with its own computational peculiarities. What is urgently needed to advance and combine research work in this area is a system that is capable of handling different representation frameworks in parallel. Therefore, we will present the KReator toolbox, a versatile integrated development environment for knowledge engineering in the field of statistical relational learning which aims at filling this gap. As statistical relational learning is a (relatively) young research area there are many different proposals for integrating probability theory in first-order logic, some of them mentioned above. Although many researchers have implementations of their approaches available, most of these implementations are prototypical, and in order to compare different approaches one has to learn the usage of different tools. The KReator system provides a common interface for different approaches to statistical relational learning and supports the researcher and knowledge engineer in developing knowledge bases and using them in a common and easy-to-use fashion.

In the last part of the body of the paper, the use of probabilistic knowledge representation will be illustrated in applications in medical and biochemical domains, with an emphasis on maximum entropy methods.

To summarize, this paper addresses the motivations for using probabilistic methods in logic-based knowledge representation, starting with standard propositional approaches and moving on to relational probabilistic knowledge representation by sketching some major approaches. As a special focus of the paper, we feature approaches that are based on the principle of maximum entropy as an elegant and powerful methodology that provides an excellent framework for commonsense and uncertain reasoning. The paper comprises four main aspects. First, we give an introduction to frameworks for relational maximum entropy which are novel approaches to relational probabilistic knowledge representation and reasoning. Second, we investigate the problem of comparing and evaluating relational probabilistic models by proposing a series of abstract evaluation criteria and applying these to the different formalisms. Third, we present the KReator development environment which is a versatile tool for working with relational probabilistic models, and finally, various application scenarios are described.

The rest of this paper revises and extends work presented in [38, 66, 4], and is organized as follows. In Section 2 we give some background on probabilistic propositional logic, Bayesian logic programs, and Markov logic networks. In Section 3 we introduce novel frameworks of relational maximum entropy. Section 4 proposes and discusses a series of criteria for comparing and judging different

formalisms for relational probabilistic knowledge representation. Section 5 gives an overview of various systems and in particular of the KReator system. Applications of the presented approaches and systems are illustrated in Section 6, and Section 7 concludes the paper and points out further work.

2 Probabilistic Knowledge Representation

In this section we first have a brief look on propositional models for probabilistic reasoning and continue with recalling the basics of the relational approaches of Bayesian logic programs [43] and Markov logic networks [59]. All approaches considered in this paper are based on semantics, i. e., probabilities are computed intensionally.

2.1 Propositional Approaches

From a computational point of view, even in the propositional case, probabilities are problematic due to their high complexity. In a seminal paper, Pearl [55] elaborated on graphical structures that allow local propagation of probabilities resulting in a substantial reduction of complexity. These so-called *belief networks* make crucial use both of conditional dependencies, expressed via conditonal probabilities, and conditional independencies between sets of nodes, and are often assumed to model causal relationships. Generally, conditional probabilities express a kind of probabilistic rule and allow the description of prognostic or diagnostic dependencies [56]. A well-known framework which emerged from Pearl's ideas are Bayesian networks. A Bayesian network BN for a set of propositions A is a tuple $BN = (A, E, P)$ such that (A, E) is a directed acyclic graph and P is a probability function that obeys the *local Markov property*

$$\{a\} \perp\!\!\!\perp_{\mathrm{P}} \mathsf{nd}(a) \mid \mathsf{pa}(a) \quad \text{(for every } a \in A\text{)}, \tag{1}$$

which expresses that each vertex a is conditionally independent of its non-descendants $\mathsf{nd}(a)$, given the values of its parents $\mathsf{pa}(a)$. Due to this property, the probability function P can be decomposed into conditional probability functions for each node $a \in A$.

Example 1. We adapt an example on medical diagnosis, cf. [56]. Consider the propositions $A = \{a, b, c, d, e\}$ with the informal interpretations

a	cancer	b	increased serum calcium level	c	brain tumor
d	coma	e	headache		

and a Bayesian network $BN_{\mathrm{med}} = (A, E, P)$ with (A, E) given as depicted in Fig. 1. It follows that P has to adhere to the conditional independence $\{b\} \perp\!\!\!\perp_{\mathrm{P}} \{c\} \mid \{a\}$ (among others). Moreover, the probability of a possible world such as $ab\overline{c}\overline{d}e$ can be written as

$$P(ab\overline{c}\overline{d}e) = P(e \mid \overline{c}) \cdot P(\overline{d} \mid b\overline{c}) \cdot P(\overline{c} \mid a) \cdot P(b \mid a) \cdot P(a).$$

Therefore, P can be completely described by e. g. the following assignments[1]:

$$\begin{aligned} P(a) &= 0.20 & & \\ P(b \mid a) &= 0.80 & P(b \mid \overline{a}) &= 0.20 \\ P(c \mid a) &= 0.20 & P(c \mid \overline{a}) &= 0.05 \\ P(e \mid c) &= 0.80 & P(e \mid \overline{c}) &= 0.60 \\ P(d \mid b \wedge c) &= 0.80 & P(d \mid b \wedge \overline{c}) &= 0.90 \\ P(d \mid \overline{b} \wedge c) &= 0.70 & P(d \mid \overline{b} \wedge \overline{c}) &= 0.05 \end{aligned}$$

Note that the probabilities of negated variables derive from the above equations via e. g. $P(\overline{e} \mid c) = 1 - P(e \mid c)$. By only defining the above conditional probabilities the function P can be compactly stored.

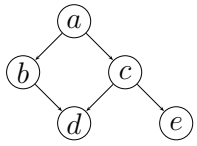

Fig. 1. The graph (A, E) from Ex. 1

Another approach that facilitates conditional probabilities is conditional logic [52] which is a knowledge representation formalism that concentrates on the role of *conditionals* or *if-then-rules*. A conditional of the form $(\psi \mid \phi)$ connects some detached pieces of information ϕ, ψ and represents a rule "*If ϕ then (usually, probably) ψ*". A *probabilistic conditional* is an expression of the form $(\psi \mid \phi)[d]$ with propositional formulas ϕ and ψ and $d \in [0, 1]$.

Example 2. The well-known penguin example that illustrates the problem of exceptions in subclasses can be represented as a knowledge base $\mathcal{R}$ with $\mathcal{R} = \{r_1, r_2, r_3\}$ with

$$r_1 = (bird \mid peng)[1.0] \qquad r_2 = (fly \mid bird)[0.9] \qquad r_3 = (fly \mid peng)[0.01].$$

A probability function P satisfies a probabilistic conditional

$$P \models (\psi \mid \phi)[d] \quad \text{if and only if} \quad P(\psi \mid \phi) = d \text{ and } P(\phi) > 0. \tag{2}$$

Reasoning from conditional probabilistic knowledge bases usually considers the conditional probabilities as constraints and either takes all possible probabilistic models (according to the satisfaction relation defined by Equation (2)) into account, implementing a kind of classical probabilistic consequence operation, or

[1] The numbers have been arbitrarily chosen and may not describe the real world.

selects one best model. One of the most prominent approaches following the first idea was presented by Nilsson [51]. However, inference via classical probabilistic consequence is often too cautious, even in simple cases, the probabilities compatible with the given constraints in the knowledge base span the whole unit interval. On the other hand, selecting a single model appears to be quite bold, so this selection has to be done in a careful way. The principle of maximum entropy has been established as a very useful guideline to solve this problem. The entropy $H(P)$ of a probability function P is defined via $H(P) = -\sum_\omega P(\omega) \log P(\omega)$ with $0 \cdot \log 0 = 0$. By selecting $P^* = \arg\max_{P \models \mathcal{R}} H(P)$ as the (unique) model of the knowledge base $\mathcal{R}$ with maximal entropy, one obtains a probability function that both satisfies all conditionals in $\mathcal{R}$ and adds as few additional information (in the information-theoretic sense) as possible. For a consistent knowledge base $\mathcal{R}$ the maximum entropy model P^* is uniquely determined, cf. [36]. Model-based probabilistic inference via P^* shows excellent logical properties [36], and has been proved to be most adequate for commonsense reasoning [53].

While the approaches sketched above are limited in the sense that they are based on propositional logic, probabilistic relational formalisms, i. e. formalisms that incorporate aspects of first-order logic for knowledge representation, have been proposed. For instance, the proposals in [40, 69, 8] employ concepts of logic programming, and as classical logic programming, they rely on the syntactic representation for computing probabilities. In this paper, we will focus on approaches that are semantically based in the sense that probabilities are computed intensionally. We continue with presenting two examples, Bayesian Logic Programs and Markov Logic Networks. For both formalisms, we use the naming convention that variables are written with a beginning uppercase letter and constants are written with a beginning lowercase letter. [2]

2.2 Bayesian Logic Programs

In contrast to first-order logic, Bayesian logic programs (BLPs) [43] employ an extended form of predicates and atoms. In BLPs, *Bayesian predicates* are predicates that feature an arbitrary set as possible states, i. e. not necessarily the Boolean values $\{\mathsf{true}, \mathsf{false}\}$. For example, the Bayesian predicate *bloodtype*/*1* may represent the blood type of a person using the possible states $S(\mathit{bloodtype}) = \{a, b, ab, 0\}$. Analogously to first-order logic, Bayesian predicates can be instantiated to *Bayesian atoms* using constants and variables and then each ground Bayesian atom represents a single random variable. If A is a Bayesian atom of the Bayesian predicate p we set $S(A) = S(p)$.

Definition 1 (Bayesian clause, conditional probability distribution). *A* Bayesian clause c *is an expression* $c = (h \mid b_1, \ldots, b_n)$ *with Bayesian atoms* $h, b_1, \ldots, b_n$. *With a Bayesian clause* $c = (h \mid b_1, \ldots, b_n)$ *we associate a function*

[2] Note that this convention differs from the standard convention for Markov logic networks.

$\mathsf{cpd}_c : S(h) \times S(b_1) \times \ldots \times S(b_n) \to [0,1]$ *that fulfills*

$$\forall vb_1 \in S(b_1), \ldots, vb_n \in S(b_n) : \sum_{vh \in S(h)} \mathsf{cpd}_c(vh, vb_1, \ldots, vb_n) = 1. \qquad (3)$$

We call cpd_c *a* conditional probability distribution. *Let* CPD_p *denote the set of all conditional probability distributions for atoms of predicate* p*, i. e., it is* $\mathsf{CPD}_p = \{\mathsf{cpd}_{h \mid b_1, \ldots b_n} \mid h \text{ is an atom of } p\}$.

As usual, if the body of a Bayesian clause c is empty ($n = 0$) we write c as (h) instead of $(h \mid)$ and call c a *Bayesian fact*. Condition (3) ensures that cpd_c indeed describes a conditional probability distribution.

Example 3. We represent the well-known alarm example from [56] , which describes the following scenario: Some person X has a house equipped with an alarm system. A burglary will most probably trigger the alarm. If a tornado occurs in the respective town person X lives in, the alarm will most probably be triggered as well. The probability of a burglary at person X's house depends on the neighborhood she lives in. We use the predicates *alarm/1*, *burglary/1*, *tornado/1*, *lives_in/2*, and *neighbourhood/1* with $S(alarm/1) = \{\mathsf{true}, \mathsf{false}\}$, $S(burglary/1) = \{\mathsf{true}, \mathsf{false}\}$, $S(tornado/1) = \{\mathsf{true}, \mathsf{false}\}$, $S(lives\_in/2) = \{\mathsf{true}, \mathsf{false}\}$, and $S(neighborhood/1) = \{\mathsf{bad}, \mathsf{average}, \mathsf{good}\}$. Define the set $\{c_1, c_2, c_3\}$ of Bayesian clauses via

$$\begin{aligned} c_1 = &\quad (alarm(\mathsf{X}) \mid burglary(\mathsf{X})) \\ c_2 = &\quad (alarm(\mathsf{X}) \mid lives\_in(\mathsf{X}, \mathsf{Y}), tornado(\mathsf{Y})) \\ c_3 = &\quad (burglary(\mathsf{X}) \mid neighborhood(\mathsf{X})) \end{aligned}$$

For each Bayesian clause c_i, we define a function cpd_{c_i} which expresses our subjective beliefs (note that function values with first argument false derive directly)

$$\begin{aligned} \mathsf{cpd}_{c_1}(\mathsf{true}, \mathsf{true}) &= 0.9 & \mathsf{cpd}_{c_1}(\mathsf{true}, \mathsf{false}) &= 0 \\ \mathsf{cpd}_{c_2}(\mathsf{true}, \mathsf{true}, \mathsf{true}) &= 0.9 & \mathsf{cpd}_{c_2}(\mathsf{true}, \mathsf{false}, \mathsf{true}) &= 0 \\ \mathsf{cpd}_{c_2}(\mathsf{true}, \mathsf{true}, \mathsf{false}) &= 0.01 & \mathsf{cpd}_{c_2}(\mathsf{true}, \mathsf{false}, \mathsf{false}) &= 0 \\ \mathsf{cpd}_{c_3}(\mathsf{true}, \mathsf{bad}) &= 0.6 & \mathsf{cpd}_{c_3}(\mathsf{true}, \mathsf{average}) &= 0.4 \\ \mathsf{cpd}_{c_3}(\mathsf{true}, \mathsf{good}) &= 0.3 \end{aligned}$$

For example, cpd_{c_2} expresses our subjective belief on the probability that the alarm of a person X will go off given that we know that X lives in town Y and there is currently a tornado in Y to be 0.9. Furthermore, we believe that the probability that the alarm of X will go on if we know that X lives in Y and that there is no tornado in Y is 0.01.

Example 4. We represent the example from the introduction about elephants and keepers as a BLP. Let *likes*/2, *keeper*/1, and *elephants*/1 be some predicates with $S(likes) = \{\mathsf{true}, \mathsf{false}\}$, $S(keeper) = \{\mathsf{true}, \mathsf{false}\}$, and $S(elephant) = \{\mathsf{true}, \mathsf{false}\}$. Define the set $\{c_1, c_2, c_3\}$ of Bayesian clauses via

$$\begin{aligned} c_1 = &\quad (likes(\mathsf{X}, \mathsf{Y}) \mid elephant(\mathsf{X}), keeper(\mathsf{Y})) \\ c_2 = &\quad (likes(\mathsf{X}, \mathsf{fred}) \mid elephant(\mathsf{X})) \\ c_3 = &\quad (likes(\mathsf{clyde}, \mathsf{fred})) \end{aligned}$$

For each Bayesian clause c_i, we define a function cpd_{c_i} which expresses our subjective beliefs (note that function values with first argument false are omitted again)

$$\begin{aligned} \mathsf{cpd}_{c_1}(\mathsf{true}, \mathsf{true}, \mathsf{true}) &= 0.8 & \mathsf{cpd}_{c_1}(\mathsf{true}, \mathsf{false}, \mathsf{true}) &= 0.5 \\ \mathsf{cpd}_{c_1}(\mathsf{true}, \mathsf{true}, \mathsf{false}) &= 0.5 & \mathsf{cpd}_{c_1}(\mathsf{true}, \mathsf{false}, \mathsf{false}) &= 0.5 \\ \mathsf{cpd}_{c_2}(\mathsf{true}, \mathsf{true}) &= 0.3 & \mathsf{cpd}_{c_2}(\mathsf{true}, \mathsf{false}) &= 0.5 \\ \mathsf{cpd}_{c_3}(\mathsf{true}) &= 0.9 \end{aligned}$$

Note that some of the probabilities defined for each conditional probability distribution are somewhat arbitrary. The problem is that defining a probability for a rule, given that its premise is not fulfilled, is a hard task. For instance, consider clause c_2 stating that usually elephants do not like Fred. But what is the probability of a non-elephant liking Fred? It is a serious drawback of Bayesian logic programs (and Bayes nets in general) that they demand for a full specification of a conditional probability distribution even if complete information is not available. In this example, we filled in this missing information by committing to as little information as possible, i. e. by assigning equal probabilities to the remaining cases.

The clauses c_1 and c_2 in Example 3 illustrate that it is possible to have multiple clauses with the same head. This means that there may be multiple causes for some effect or multiple explanations for some observation. In order to represent these kinds of scenarios the probabilities of causes or explanations have to be aggregated. Appropriate choices for such so-called *combining rules* are *average* or *noisy-or*, cf. [56] and [43].

Now we are able to define Bayesian logic programs as follows.

Definition 2 (Bayesian logic program). *A* Bayesian logic program B *is a tuple* $B = (C, D, R)$ *with a (finite) set of Bayesian clauses* $C = \{c_1, \ldots, c_n\}$*, a set of conditional probability distributions* $D = \{\mathsf{cpd}_{c_1}, \ldots, \mathsf{cpd}_{c_n}\}$ *(one for each clause in* C*), and a set of combining rules* $R = \{\mathsf{cr}_{p_1}, \ldots, \mathsf{cr}_{p_m}\}$ *(one for each Bayesian predicate appearing in* C*).*

Semantics are given to Bayesian logic programs via transformation into the propositional case, i. e. into Bayesian networks. Given a specific (finite) universe U, a Bayesian network BN can be constructed by introducing a node for every

grounded Bayesian atom in B. Using the conditional probability distributions of the grounded clauses and the combining rules of B, a (joint) conditional probability distribution can be specified for any node in BN. If BN is acyclic, this transformation uniquely determines a probability distribution P on the grounded Bayesian atoms of B which permits inference, i. e. P can be used to answer queries.

2.3 Markov Logic Networks

Markov logic [59] establishes a framework which combines Markov networks [56] with first-order logic to handle a broad area of statistical relational learning tasks. The Markov logic syntax complies with first-order logic where each formula is quantified by an additional weight value. Semantics are given to sets of Markov logic formulas by a probability distribution over propositional possible worlds that is calculated as a log-linear model over weighted ground formulas.

Definition 3 (Markov logic network). *A* Markov logic network (MLN) $L = \{(F_1, w_1), \ldots, (F_n, w_n)\}$ *is a set of first-order logic formulas F_i, where each formula F_i is quantified by a real value w_i. Together with a finite set of constants C it defines a* (ground) Markov network $M_{L,C}$ *as follows:*

- *$M_{L,C}$ contains a node for each possible grounding of each predicate appearing in L.*
- *$M_{L,C}$ contains an edge between two nodes iff their ground atoms appear together in at least one grounding of one formula in L.*
- *$M_{L,C}$ contains one feature (function) for each possible grounding of each formula F_i in L. The value of the feature for a possible world x is 1, if the ground formula is true for x (and 0 otherwise). Each feature is weighted by the weight w_i of its respecting formula F_i.*

Let $n_i(\omega)$ denote the number of true groundings of the formula F_i for a possible world ω in the ground Markov network $M_{L,C}$. For a ground Markov network $M_{L,C}$, a probability distribution $P_{M_{L,C}}$ over possible worlds $\omega \in \Omega$ can be specified by the following log-linear model [16]

$$P_{M_{L,C}}(\omega) = \frac{1}{Z} \exp\left(\sum_{(F_i, w_i) \in L} w_i n_i(\omega)\right) \tag{4}$$

with the normalization factor

$$Z = \sum_{\omega \in \Omega} \exp\left(\sum_{(F_i, w_i) \in L} w_i n_i(\omega)\right).$$

In a straightforward way, the probability of an arbitrary formula A can be computed over those possible worlds satisfying A

$$P_{M_{L,C}}(A) = \sum_{\omega \in \Omega:\, \omega \models A} P_{M_{L,C}}(\omega)$$

where $\omega \models A$ denotes the classical first-order satisfaction relation.

In order to represent the uncertain beliefs in a Markov logic network the weights of formulas have to be determined. In [59] it is suggested that weights of formulas have to be learned from data. Nonetheless, in [23, 59] a heuristic is discussed that determines weights of formulas from probabilities. In [59] an interpretation of the weight w_i of a formula F_i is provided as the log-odd between a world where F_i is true and a world where F_i is false, other things being equal. Considering this interpretation one might choose $w_i = \log \frac{p}{1-p}$ as the weight of a formula F_i when p is the intended probability of F. In [23] an extension of Markov logic is introduced that allows the direct representation of probabilities and conditional probabilities. However, it has to be noted that both the heuristic of [59] and the approach of [23] do not guarantee that the resulting log-linear model $P_{M_{L,C}}$ actually realizes the intended probabilities. That is, even if the probability p of some formula F has been specified in an MLN via the respective approach, the probability $P_{M_{L,C}}(F)$ might be very different from p, see also [23] for a discussion.

In the following, we represent the informal scenarios underlying Examples 3 and 4 using Markov logic. In particular, we do not aim at translating the probabilistic models of Examples 3 and 4 to Markov logic but we intend to give an adequate commonsense representation of the information. Consequently, we have chosen to model rule-like knowledge using material implications. Even though this might be a quite intuitive modeling approach at first glance, it should also be mentioned that implications are (in general) a sub-optimal choice to model conditional knowledge. However, we deliberately use this approach as it is the most obvious choice.

Example 5. In the following example, we model the relations described in Example 3 as an MLN (using the Alchemy syntax [45] for MLN files). The "!" operator used in the predicate declarations of *lives_in* and *nhood* enforces that the respective variables will have mutually exclusive and exhaustive values, i. e. that every person lives in exactly one town and exactly one neighborhood (in terms of ground atoms). The weights of the formulas express the subjective strength of each rule. The actual weights given below are estimated in a way such that the resulting probabilities match the intended probabilities given in Example 3. We declare the typed predicates $alarm(Person)$, $nhood(Person, hood\_state!)$, $lives\_in(Person, town!)$, $burglary(Person)$ with the types and constants $Person = \{james, carl\}$, $town = \{yorkshire, austin\}$, $hood\_state = \{bad, average, good\}$. Now consider the following weighted formulas:

$$
\begin{aligned}
2.2 \quad & burglary(\mathsf{X}) \Rightarrow alarm(\mathsf{X}) \\
2.2 \quad & lives\_in(\mathsf{X}, \mathsf{Y}) \wedge tornado(\mathsf{Y}) \Rightarrow alarm(\mathsf{X}) \\
-0.8 \quad & nhood(\mathsf{X}, good) \Rightarrow burglary(\mathsf{X}) \\
-0.4 \quad & nhood(\mathsf{X}, average) \Rightarrow burglary(\mathsf{X}) \\
0.4 \quad & nhood(\mathsf{X}, bad) \Rightarrow burglary(\mathsf{X})
\end{aligned}
$$

Note that, the rule-like knowledge from Example 3 has been modeled as material implications.

Example 6. We continue with representing Example 4 as an MLN:

$$\begin{aligned} 2 \quad & \mathit{elephant}(\mathsf{X}) \wedge \mathit{keeper}(\mathsf{Y}) \Rightarrow \mathit{likes}(\mathsf{X}, \mathsf{Y}) \\ -0.8 \quad & \mathit{elephant}(\mathsf{X}) \Rightarrow \mathit{likes}(\mathsf{X}, \mathsf{fred}) \\ 2.2 \quad & \mathit{likes}(\mathsf{clyde}, \mathsf{fred}) \end{aligned}$$

As it was the case in Example 5 the rule-like knowledge of this example has been modeled using material implications.

3 Relational Probabilistic Conditional Logic

In the following, we present a relational probabilistic framework that extends probabilistic conditional logic to the relational case and bases its inference mechanism on the principle of maximum entropy, cf. Section 2.1. Parts of this framework have been introduced previously in [47] and [41].

As a base of our probabilistic language, we use a fragment $\mathcal{L}$ of a first-order language over a signature Σ containing only predicates and constants, the formulas of which are well-formed according to the usual standards, but without any quantifiers. As before, we write variables with a beginning uppercase letter and constants with a beginning lowercase letter. The language $\mathcal{L}$ may be typed, i. e., the constants $U = U_{\mathcal{L}}$ and the variables $V = V_{\mathcal{L}}$ of $\mathcal{L}$ are partitioned into different types, and the arguments of the predicates may be typed as well. A grounding substitution $\theta : V_{\mathcal{L}} \to U_{\mathcal{L}}$ instantiates variables with constants. It is extended to formulas in the usual way, e. g., we define $\theta(p(X, Y) \wedge q(X)) = p(\theta(X), \theta(Y)) \wedge q(\theta(X))$. A grounding substitution θ is *legal* if any variable of type S in r is mapped to a constant of type S. We extend this relational language $\mathcal{L}$ to a probabilistic conditional language by introducing conditionals and probabilities.

Definition 4 (Relational probabilistic conditional). *A* relational probabilistic conditional *r is an expression of the form $r = (\phi \,|\, \psi)[\alpha]$ with formulas $\phi, \psi \in \mathcal{L}$ and $\alpha \in [0, 1]$. The conditional and the probabilistic parts of r are denoted by $\mathit{Cond}(r) = (\phi \,|\, \psi)$ and $\mathit{Prob}(r) = \alpha$, respectively.*

We also allow *substitution constraints* to appear inside the conditional part of a constraint. For variables X, Y a substitution constraint is an expression $\mathsf{X} \neq \mathsf{Y}$ with the intuitive meaning that variables X and Y should not be instantiated by the same constant. For example, the conditional

$$(\mathit{knowEachOther}(\mathsf{X}, \mathsf{Y}) \,|\, \mathit{has\_father}(\mathsf{X}, \mathsf{Z}) \wedge \mathit{has\_father}(\mathsf{Y}, \mathsf{Z}) \wedge \mathsf{X} \neq \mathsf{Y})[0.9]$$

states that the probability that two different persons having the same father know each other is 0.9. Substitution constraints are extended to refer to constants

as well, i. e., we allow expressions of the form $\mathsf{X} \neq c$ and even $c_1 \neq c_2$ (for constants c, c_1, c_2), as, on a syntactical layer, the relation $=$ is treated the same way as any binary predicate.

A conditional r is called *ground* iff r contains no variables. Non-ground conditionals can be grounded by legal grounding substitutions. The language of all relational probabilistic conditionals is denoted by $(\mathcal{L} \mid \mathcal{L})^{rel}$, and the restricted language on all ground conditionals using constants from U is denoted by $(\mathcal{L} \mid \mathcal{L})^{rel}_U$. A set $\mathcal{R}$ of relational probabilistic conditionals is called an *RPCL knowledge base*. In the following, we represent Example 3 as an RPCL knowledge base.

Example 7. Let $\mathcal{L}$ contain the types $S = \{Person, Town, Status\}$ with constants

$$\begin{aligned} U_{Person} &= \{james, carl\} \quad \text{of type } Person, \\ U_{Town} &= \{yorkshire, austin\} \quad \text{of type } Town, \text{ and} \\ U_{Status} &= \{bad, average, good\} \quad \text{of type } Status \end{aligned}$$

and predicates

$$\begin{aligned} Pred = \{\ & alarm(Person), \\ & burglary(Person), \\ & lives\_in(Person, Town), \\ & nhood(Person, Status)\ \}. \end{aligned}$$

Let $\mathcal{R} = \{c_1, \ldots, c_7\}$ be an RPCL knowledge base given via

$$\begin{aligned} c_1 &= (alarm(\mathsf{X}) \mid burglary(\mathsf{X}))\ [0.9] \\ c_2 &= (alarm(\mathsf{X}) \mid lives\_in(\mathsf{X}, \mathsf{Y}) \wedge tornado(\mathsf{Y}))\ [0.9]\ \} \\ c_3 &= (burglary(\mathsf{X}) \mid nhood(\mathsf{X}, bad))\ [0.6] \\ c_4 &= (burglary(\mathsf{X}) \mid nhood(\mathsf{X}, average))\ [0.4] \\ c_5 &= (burglary(\mathsf{X}) \mid nhood(\mathsf{X}, good))\ [0.3] \\ c_6 &= (nhood(\mathsf{X}, \mathsf{Z}) \wedge nhood(\mathsf{X}, \mathsf{Y}) \wedge \mathsf{Y} \neq \mathsf{Z})\ [0.0] \\ c_7 &= (lives\_in(\mathsf{X}, \mathsf{Z}) \wedge lives\_in(\mathsf{X}, \mathsf{Y}) \wedge \mathsf{Y} \neq \mathsf{Z})\ [0.0] \end{aligned}$$

Notice that conditionals c_6 and c_7 ensure that the predicates "*nhood*" and "*lives_in*" cannot assign two different states to the same individual. Observe also that—in contrast to BLPs—we do not need to specify the probability of *alarm* being true if no burglary takes place which is usually hard to estimate.

Note that the example on elephants and keepers as given in the introduction makes already uses the syntax of relational probabilistic conditional logic.

Regarding semantics, ground conditionals $r \in (\mathcal{L} \mid \mathcal{L})^{rel}_U$ can be interpreted as in the propositional case [61]. That is, if $\Omega_{\mathcal{L}}$ is the set of interpretations of $\mathcal{L}$

and $P : \Omega_{\mathcal{L}} \to [0,1]$ is a probability distribution on $\Omega_{\mathcal{L}}$ then $P \models (\phi|\psi)[\alpha]$ iff

$$P(\phi \mid \psi) = \frac{P(\phi \wedge \psi)}{P(\psi)} = \alpha \quad \text{and} \quad P(\psi) > 0 \tag{5}$$

with

$$P(\phi) = \sum_{\omega \in \Omega_{\mathcal{L}}, \omega \models \phi} P(\omega)$$

for a ground formula $\phi \in \mathcal{L}$. Non-conditional formulas $(\phi)[\alpha]$ can be considered consistently as conditionals with tautological premise $(\phi \mid \top)[\alpha]$, so that no explicit distinction between conditionals and flat formulas is necessary in the following. For a substitution expression $c \neq c'$ with *different* constant symbols c, c' we define $\omega \models c \neq c'$ to be true for every $\omega \in \Omega_{\mathcal{L}}$. Correspondingly, for a substitution expression $c \neq c$ we define $\omega \models c \neq c$ to be false for every $\omega \in \Omega_{\mathcal{L}}$.

In the general case, if r contains variables, different groundings may yield different conditional probability values. This could be handled by assigning probabilistic intervals to open conditionals in order to cover the probabilities of all instantiations, as is done e. g. in [40]. However, this approach only allows us to draw very vague inferences on the probabilities of individual instantiations. So, following our major guideline of using expressive point-wise probabilities for knowledge representation, we consider *grounding strategies* here.

Definition 5 (Grounding operator). *A* grounding operator *(GOP) $\mathcal{G}$ is a function $\mathcal{G} : \mathfrak{P}((\mathcal{L} \mid \mathcal{L})^{rel}) \to \mathfrak{P}((\mathcal{L} \mid \mathcal{L})^{rel}_U)$.*

A GOP $\mathcal{G}$ takes a general relational knowledge base $\mathcal{R}$ and maps it to a ground one $\mathcal{G}(\mathcal{R})$ by instantiating variables according to the language of the knowledge base and some strategy. By doing so we may use the propositional probabilistic semantics for the propositional case.

The actual definition of a GOP relies on grounding substitutions for variables. For a conditional r let $\Gamma(r)$ denote the set of all legal grounding substitutions for r. If r contains substitution constraints we assume them to be respected, e. g. if r contains the constraint $\mathsf{X} \neq \mathsf{Y}$ then every $\theta \in \Gamma(r)$ obeys $\theta(\mathsf{X}) \neq \theta(\mathsf{Y})$. The most simple approach to ground a knowledge base is *universal instantiation* which naively instantiates every variable with every constant of the same type (for more sophisticated grounding operators, please see [48]).

Definition 6 (Naive grounding operator). *The* naive grounding operator *$\mathcal{G}_U$ is defined as $\mathcal{G}_U(\mathcal{R}) := \{\theta(r) \mid r \in \mathcal{R}, \theta \in \Gamma(r)\}$.*

Using the naive grounding operator we can define probabilistic satisfaction for relational probabilistic conditional logic via *grounding semantics*. Let P be a probability distribution as above and let r be a relational probabilistic conditional. Then P $\mathcal{G}_U$-satisfies r, denoted by $P \models_{\mathcal{G}_U} r$, iff

$$P \models r' \quad \text{for all} \quad r' \in \mathcal{G}_U(\{r\}).$$

Consequently, a probability distribution P $\mathcal{G}_U$-satisfies a knowledge base $\mathcal{R}$, denoted by $P \models_{\mathcal{G}_U} \mathcal{R}$, iff $P \models r'$ for all $r' \in \mathcal{G}_U(\mathcal{R})$.

Both *averaging* and *aggregating semantics* [41, 67] take a more sophisticated approach in defining probabilistic satisfaction by interpreting the intended probability x of a conditional with free variables only as a guideline for the probabilities of its instances and the actual probabilities may differ from x.

As for averaging semantics, the entailment relation $\models_{\varnothing}$ is defined by

$$P \models_{\varnothing} (\psi \mid \phi)[\alpha] \text{ iff } \frac{\sum_{(\psi' \mid \phi')[\alpha] \in \mathcal{G}_U((\psi \mid \phi)[\alpha])} P(\psi' \mid \phi')}{|\mathcal{G}_U((\psi \mid \phi)[\alpha])|} = \alpha\,. \tag{6}$$

Intuitively spoken, a probability distribution P $\varnothing$-satisfies a conditional $(\psi \mid \phi)[\alpha]$ if the average of the individual instantiations of $(\psi \mid \phi)[\alpha]$ is α.

Aggregating semantics is inspired by statistical approaches. However, instead of counting objects, or tuples of objects, respectively, that make a formula true, we sum up the probabilities of the correspondingly instantiated formulas. The entailment relation $\models_{\odot}$ is defined by

$$P \models_{\odot} (\psi \mid \phi)[\alpha] \text{ iff } \frac{\sum_{(\psi' \mid \phi')[\alpha] \in \mathcal{G}_U((\psi \mid \phi)[\alpha])} P(\psi'\phi')}{\sum_{(\psi' \mid \phi')[\alpha] \in \mathcal{G}_U((\psi \mid \phi)[\alpha])} P(\psi')} = \alpha\,. \tag{7}$$

If P is a uniform distribution, we end up with a statistical interpretation of the conditional. However, the probabilities in this paper will be subjective, so $\models_{\odot}$ mimicks the statistical view from a subjective perspective.

Analogously as for $\mathcal{G}_U$ before, if $\circ$ is one of $\varnothing$ or $\odot$ then we say that P $\circ$-satisfies a knowledge base $\mathcal{R}$, denoted by $P \models_{\circ} \mathcal{R}$ iff $P \models_{\circ} r$ for all $r \in \mathcal{R}$. Note that all three semantics are extensions of classical probabilistic semantics for propositional probabilistic conditional logic [35].

Having properly defined models of knowledge bases, we now adopt the approach of reasoning with maximum entropy—see Section 2.1—for the relational case and define

$$P^{ME}_{\mathcal{R},\circ} = \arg\max_{P \models_{\circ} \mathcal{R}} \mathcal{H}(P) \tag{8}$$

with $\circ$ being one of $\mathcal{G}_U$, $\varnothing$, or $\odot$. For our general framework of relational probabilistic conditional logic (RPCL), we abbreviate the approaches of reasoning based on the principle of maximum entropy with grounding (using the naive grounding operator), averaging, and aggregating semantics with $\mathrm{ME}_{\mathcal{G}_U}$, $\mathrm{ME}_{\varnothing}$, and $\mathrm{ME}_{\odot}$, respectively. We say that a formula $(\psi \mid \phi)[x]$ is $ME_{\circ}$-inferred from $\mathcal{R}$ iff $P^{ME}_{\mathcal{R},\circ} \models_{\circ} (\psi \mid \phi)[x]$ with $\circ$ being one of $\mathcal{G}_U$, $\varnothing$, or $\odot$.

For a more detailed discussion of the above semantics and the properties of inference on the principle of maximum entropy we refer the reader to [47, 41].

4 Evaluation and Comparison Criteria

When looking at probabilistic relational modeling, there are different motivations and objectives for choosing a particular representation or a specific approach.

In [9], a comparison between several statistical relational learning systems is done, with an emphasis on the learning aspects. Here, we will concentrate on the point of view of knowledge representation and reasoning; since primarily, BLPs and MLNs are statistical models, it is interesting to investigate them from this perspective. We formulate a series of criteria yielding some useful guidance for judging and comparing approaches to probabilistic relational knowledge representation. These criteria are organized along several themes, ranging from language aspects to aspects concerning individuals and universes; Fig. 2 gives an overview. For every criterion, we evaluate each of the five approaches discussed above—BLPs, MLNs, and the three RPCL approaches—with respect to that criterion. Most of the criteria do not lead to to simple yes/no answers, but require a more detailed elaboration. A preliminary discussion of these properties can be found in [4].

Fig. 2. Themes and topics of the evaluation and comparison criteria

4.1 Language Aspects

The semantics of the components of a knowledge representation language should be as declarative as possible. In particular, it should be possible to express basic concepts directly and to have an intuitive meaning for all language constructs, inference and learning results:

(L-1) Direct expression of probabilities in the form of "A holds with a probability of x".

(L-2) Direct expression of conditional probabilities as in "Provided A holds, then the probability of B is x".

Obviously, (L-1) can be viewed as a special case of (L-2) if the precondition A in (L-2) is set to be true. However, since generally, conditionals cannot be reduced to unconditional sentences, it is useful to distinguish the two cases (L-1) and

(L-2) because there are approaches that do not support the direct expression of conditionals.

Since an RPCL knowledge base supports representing formulas of the form $(B \mid A)[x]$ which constrain the conditional probability of B given A to x in any model, $\mathrm{ME}_{\mathcal{G}_U}$, $\mathrm{ME}_{\varnothing}$, $\mathrm{ME}_{\odot}$ obviously fulfill (L-1) and (L-2).

A similar observation holds for BLPs when taking into account the conditional probability distribution functions cpd_c which must be defined for any Bayesian clause c. If B is a Bayesian atom with predicate p, the Bayesian clause $c = (B \mid A)$ together with cpd_c satisfying $\mathsf{cpd}_c(B \mid A) = x$ ensures that in any ground model of a BLP, the conditional probability of B' given A' of all instances A' and B' of A and B is x. Note that this property requires that p does not occur in the head of any other clause in the BLP; if p occurs in multiple heads, the combining rule for p may yield a probability value different from x.

Since in an MLN there is no obvious correspondence between the weight of a clause and its corresponding probability and because conditionals are not supported, (L-1) and (L-2) are not satisfied by MLNs.

(L-3) Qualitative statements like "A is more probable than B" or "A is very probable".

While in propositional Bayes nets qualitative expressions of this kind have been encoded by the means of second-order probability distributions [34], such qualitative statements can be expressed in none of the five relational approaches investigated in this paper. Whether and how second order distributions could also be used in a relational setting has still to be investigated.

(L-4) Commonsense meaning: Probabilities are used for expressing uncertainty, and for each basic construct of the knowledge representation language there should be a clear intuitive or commonsense meaning. Examples of such meanings are a statistical interpretation of expressions (with respect to a population), or a subjective degree of belief (with respect to the set of possible worlds). In particular, for formulas containing variables, it should be possible to express an intuitive commonsense meaning of the formula in general, or for its ground instances, respectively.

First, the difference between statistical and subjective interpretations can be illustrated in the context of the elephant-keeper-example from the introduction by contrasting "Most elephants like their keepers" (statistical, as it refers to a whole population) vs. "Mostly, an elephant likes its keeper" (subjective, as it refers to situations, i.e., possible worlds). Basically, all three approaches considered look at probabilities from a subjective point of view. However, it is not possible to translate straightforwardly formal probabilistic (or weighted, respectively) expressions into commonsense probabilistic statements in each framework.

A Bayesian clause c in a BLP expresses qualitative information about the conditional probability of the clause's head given the body of the clause; the conditional probabilities given by cpd_c that is applied for each instance provide

inputs for computing subjective conditional probabilities by applying the combining functions and making use of the network structure. Unfortunately, the information specified in the knowledge base of a BLP may not correspond to the probabilities established in the appertaining probabilistic model. This is due to the use of the heuristic combination functions which only approximate (more or less well) probabilistic inferences. To illustrate this, in Example 3, clause c_1 together with cpd_{c_1} suggest that if X's house is burglarized, then the probability that the alarm will ring is 0.9. However, when instatiating X e.g. with *carl*, completely different values may result for $P(\mathit{alarm}(\mathsf{carl}) \mid \mathit{burglary}(\mathsf{carl}))$, depending on the probabilites of the other atoms. So, while the Bayesian clauses look like conditional probabilities, their meaning for ground instances is not clear.

Similarly, the probabilities resulting from an MLN can be classified as subjective, as the MLN semantics is based on possible worlds. Although it can be observed that the greater the weight w of an MLN clause F the more impact F will have on the probability distribution in the resulting ground Markov network $M_{L,C}$, a more precise probabilistic meaning of (F, w) is not evident since the weight w expresses merely the relative strength of the formula F in the context of the other formulas specified in the MLN knowledge base.

For each ground conditional $(B \mid A)[x]$ in an RPCL knowledge base, its commonsense meaning is given by the conditional probability of B given A being x for grounding, averaging, and aggregating semantics. However, the commonsense interpretation of conditionals with free variables is substantially different in these three semantics. For grounding semantics, a relational conditional is understood as a universal quantification of the subjective conditional probability of each ground instance within the range of the grounding operator. For instance, in the RPCL framework with the naive grounding operator, it is possible to assign to a conditional $(B \mid A)[x]$ in the knowledge base the default commonsense meaning "For all instances $(B' \mid A')[x]$ of $(B \mid A)[x]$". For averaging and aggregating semantics, the commonsense meaning is a mixture of statistical interpretation and degrees of belief. The averaging semantics yields a statistics of subjective conditional beliefs. For instance, the first conditional from the example in the introduction with an interpretation via $\mathrm{ME}_{\varnothing}$ reads as "Considering *a random* elephant-keeper-pair, the average subjective probability that the elephant likes its keeper is 0.9." The aggregating semantics exchanges the role of statistical (or population based) and subjective view by providing a kind of subjectively weighted statistics. Here, c_1 is understood as "Considering *all* elephant-keeper-pairs, the expected subjective probability that elephants like their keepers is 0.9." In contrast to the averaging semantics, the aggregating semantics gives more weight to individuals (or tuples of individuals) that are more likely to fulfill the premise of a conditional.

By taking both statistical and subjective aspects into account, both averaging and aggregating semantics allow a more realistic approach to commonsense reasoning in a relational probabilistic context. When entering a zoo (or considering the vague population of all elephants and keepers in the world) and uttering a conditional $(\mathit{likes}(X, Y) \mid \mathit{elephant}(X),\ \mathit{keeper}(Y))\ [0.8]$, human beings are very

likely to express something like "In (about) 80 % of all situations that involve an elephant and its keeper, I will notice that the elephant likes the keeper." This statement takes both beliefs about possible worlds and about the population into account, and it is exactly this perspective that averaging and aggregating semantics aim to represent. For a further discussion and evaluation of these semantics, see [41].

(L-5) Closure properties: The results obtained by inference should be expressible in the knowledge representation language, thus enabling, e. g., the integration of inferred knowledge into a knowledge base. Another closure aspect refers to the query language: Can any formula being allowed in a knowledge base be used in a query?

A language property closely related to the above is also the following one.

(L-6) Inference: What kind of queries can be answered, and which can be answered efficiently?

As (L-5) and (L-6) address similar issues we discuss these two properties together. In Bayesian logic programming queries have the form $Q = (H \mid B_1, \ldots, B_n)$ with ground atoms $H, B_1, \ldots, B_n$ and the result of a query of this form to a BLP B is the probability that H is true given that $B_1, \ldots, B_n$ are true. However, BLPs themselves consist of Bayesian clauses together with conditional probability distributions. While a query Q has the same form as a Bayesian clause the conditional probability distribution needed to parametrize this clause is given only partially by the result of querying. In particular, by a single inference step one does not obtain a piece of information that can be directly integrated into the knowledge base. However, BLP inference can be used for computing a cpd_Q for Q by generating all possible combinations of evidence for Q, allowing one to add this information as a BLP clause.

In Markov logic the result of inference is a (restricted) first-order formula together with the probability of this formula with respect to the MLN. Since knowledge representation with MLNs bases on assigning weights to formulas, MLN inference results cannot be used directly in an MLN without altering their intended meaning. However, from a categorical point of view adding a formula with its probability to an MLN is possible. But, as the weights of an MLN do not have to be normalized, interpreting probabilities as weights may yield unintuitive results.

As RPCL is defined using an explicit model theory, inference results which are formulas of the knowledge representation formalisms themselves, can be directly integrated into an RPCL knowledge base (independently of the actual semantics).

In all five approaches it is not possible to pose any representable formula as a query. In particular, queries must be ground and taking a logic formula F from a corresponding knowledge base, every ground instance of F can be used in a query. For example, given the body of the BLP clause

$$(\mathit{alarm}(\mathit{james}) \mid \mathit{burglary}(\mathit{james}), \mathit{lives\_in}(\mathit{james}, \mathit{yorkshire}), \mathit{nhood}(\mathit{average}))$$

as evidence, the BLP inference mechanism will determine the conditional probability of $alarm(james)$ given the evidence. Consequently, open queries such as "What is the probability of $alarm(X)$ given $burglary(X)$?" are allowed in none of the approaches. However, this is quite clear as the interpretation of such a query is not straightforward: should this mean "What is the *average* probability of $alarm(X)$ given $burglary(X)$ for all X?" or "Given some *normal* X, what is the probability of $alarm(X)$ given $burglary(X)$?", see [41] for some discussion. If a support system offers posing queries with free variables (as it is allowed e.g. in Alchemy), then such a query is being treated as an abbreviation for posing a sequence of all possible ground instantiations of that query.

We now turn to efficiency issues with respect to query answering. For MLNs, there exist several algorithms which allow to perform approximate inference efficiently. The straightforward calculation of an MLN query requires to determine and to sum up the (conditional) probabilities of all satisfying worlds. Since this direct approach is exponential in the number of ground atoms, it is unsuitable for ground Markov networks of realistic size. The approximate algorithms use efficient techniques as e. g. Markov Chain Monte Carlo methods and knowledge-based model construction to reduce the number of necessary calculations and to construct only the required part of the ground Markov network (depending on the respective query and the supplied evidence). We refer to [16] for a detailed discussion of these techniques and algorithms.

RPCL inference requires solving the numerical optimization problem (8) whose complexity grows in the number of possible groundings that have to be considered, cf. Equations (6) and (7). The expert system SPIRIT [62] provides an algorithm and data-structures to calculate the solution of optimization problem (8) efficiently in many cases. Since the algorithm works on a decomposition of the probability distribution, it can overcome the exponential size of the distribution. But how efficiently this decomposition can be performed depends on the structure of the (ground) conditionals and the respective query. Therefore, further work is needed to investigate how well this algorithm performs with the grounded conditional structures arising from $\mathrm{ME}_{\mathcal{G}_U}$ inference. Some criteria under which the complexity of $\mathrm{ME}_{\mathcal{G}_U}$ inference reduces to the complexity of reasoning under propositional maximum entropy are given in [24, 46]. Inference for aggregating semantics also requires solving the corresponding convex optimization problem (8). This can be accomplished by employing the well-known Generalized Iterative Scaling (GIS) algorithm technique [12]. An early implementation of an adjusted GIS algorithm already shows promising results [19].

With respect to the complexity of inference, further experimental and theoretical work is needed.

(L-7) Independence of syntax: A general requirement for logic-based representations applies also here: The semantics should be the same if the explicit knowledge is expressed by syntactic variants.

Each of the formalisms make use of a set of relational probabilistic formulas (i.e., a knowledge base) that allow probabilistic inferences for queries, together

with independence assumptions, combining functions, grounding operators etc. We might take this set of possible inferred sentences as the *semantics* of the knowledge base within the formalism. We will consider only syntactic variations that should not change inferences from a basic logical point of view, without taking complex probabilistic inferences into account.

Let KB be a knowledge base in one of the five relational approaches. Since for any variable renaming σ, the respective semantics of KB and $\sigma(KB)$ coincide, semantical equivalence with respect to variable renaming holds for BLPs, MLNs, and RME. Another form of syntactic variants arises from logically equivalent formulas, e. g. A and $A \wedge A$. In places where such formulas are allowed, they do not give rise to a different semantics in any of the five approaches BLPs, MLNs, or the different RPCL variants.

However, it should be noted that logical equivalence of single formulas has to be distinguished carefully from the case of adding a syntactic variant of a knowledge base element to that knowledge base: If $F \in KB$ and σ is a variable renaming replacing some variable in F with a new variable not occurring in KB, then in general $KB \cup \{\sigma(F)\}$ has a different semantics both for BLPs and for MLNs. For instance, when using *noisy-or* as the combining function, the probability expressed by F—and thus also by $\sigma(F)$—will typically increase when adding $\sigma(F)$ to KB.

Example 8. Consider a BLP consisting of the single clause $c = (A(\mathsf{X}) \mid B(\mathsf{X}))$ with $\mathsf{cpd}_c(\mathsf{true}, \mathsf{true}) = 0.9$, $\mathsf{cpd}_c(\mathsf{true}, \mathsf{false}) = 0.5$ and with *noisy-or* being the combining rule for predicate A. Then querying this BLP with $(A(d) \mid B(d))$ results (obviously) in the probability 0.9 for $A(d)$ being true given $B(d)$ is true. However, adding the clause c' with $c' = (A(\mathsf{Y}) \mid B(\mathsf{Y}))$ (with $\mathsf{cpd}_c = \mathsf{cpd}_{c'}$) which is a syntactical variant of c results in a probability of $1-(1-0.9)\cdot(1-0.9) = 0.99$ as both c and c' determine a probability of 0.9 and these probabilities have to be combined by the corresponding combining function (*noisy-or* in this case) to obtain the final answer to the given query.

Example 9. Similarly, consider an MLN consisting of the single formula $(B(\mathsf{X}) \Rightarrow A(\mathsf{X}),\ 1)$. Querying this MLN with $(A(d) \mid B(d))$ results in the (approximated) probability 0.764974 for $A(d)$ being true given $B(d)$ is true. However, adding the syntactic variant $(B(\mathsf{Y}) \Rightarrow A(\mathsf{Y}),\ 1)$ results in an (approximated) probability of 0.861964 (these probabilities have been computed with the Alchemy system).

For RPCL, the addition of syntactic variants in the sense described above does not influence the outcome of inference (for grounding, averaging, and aggregating semantics).

For BLPs, only atoms are allowed in the clauses of the knowledge base, but negation can be simulated for Boolean predicates by using the negated form as an atom and switching "true" to "false", and conversely. Doing so, we obtain a dually negated form of the BLP with the same semantics, as the conditional probability tables remain substantially the same in this case. For MLNs, the dual negation of a weighted formula (F, w) cannot even be defined in a reasonable way.

(L-8) Explanation capabilities for inference: It is desirable to have explanation capabilities of inference results. Which elements of the knowledge base are to what degree responsible for an inferred result? Is it possible to identify elements which did not affect a result in any way? Can some results be derived directly from certain elements of the knowledge base or does any result essentially require the calculation of an appropriate model?

The explanation of a BLP inference result is given by the obtained local Bayes net which also encodes a (logical) derivation of the query. Therefore, it is obvious which clauses of the BLP knowledge base were involved in the calculation of the result. So the BLP approach offers some distinct level of explanation capability.

MLN inference is based on a log-linear model that has to be normalized in order to represent a probability distribution, cf. [29, Ch. 12]. The value of this normalization constant depends on the relationships among the formulas of an MLN knowledge base. Therefore, an inferred probability depends on all formulas of the knowledge base, because the weights of the formulas are relative values, where the larger the weight the greater the influence of the formula. Since MLN inference involves the construction of an appropriate ground Markov network, independencies among certain ground atoms are indicated by this network. So some independency aspects of inferred results can be explained by the net structure.

Inference in RPCL relies on solving the optimization problem (8). In some special cases (regarding the query and the conditionals in the knowledge base), the result of a query might be estimated directly considering how reasoning under the maximal entropy distribution "behaves". So in such rare cases, the inferred result can be explained by certain aspects of the knowledge base (having the principle of maximum entropy in mind). But in general, no intuitive explanation of inference results is evident for both the MLN and RPCL approaches.

(L-9) Expressivity: How is an explicit model theory given? Which models can be expressed using a particular approach, and how can they be defined?

The semantic models of BLPs and MLNs are ground Bayes and Markov nets, respectively, representing a probability distribution. Likewise, the semantics of an RPCL knowledge base also yields a probability distribution. While any of these approaches can be used to define an arbitrary probability distribution (over a finite domain), the more interesting question is *how* this can be done. Jaeger [32] proposes a schema of comparing two formalisms by using two components: A generic component that is independent of a particular universe, and a component that takes into account a universe of constants. Using this framework, [32] shows that relational Bayesian networks (RBNs) [31, 10] can encode MLN models, i. e., that RBNs are at least as expressive as MLNs; in both cases, basic versions of the respective languages are considered. No other expressivity comparison results using the framework of [32] seem to have been published. It can be expected that due to the special requirements of the respective frameworks, many pairs of probabilistic relational languages will be just incomparable; for

instance, it is easy to see that MLNs cannot be used to encode RPCL as the strict realization of probabilities resp. conditional probabilities is not possible with MLNs, see also the discussion in Section 2.3. Furthermore, consider the MLN $\{(A(X), w_1), (A(c), w_2)\}$ with some constant c and $w_1 \neq w_2$. In the log-linear model of this MLN the actual probability of $A(c)$ strongly depends on the size of the universe. In RPCL, given a ground conditional $(\phi \mid \psi)[\alpha]$ the probability $P(\phi \mid \psi)$ of every model P of $(\phi \mid \psi)[\alpha]$ does not depend on the size of the universe and is always α, cf. Equation (5). Assume there is a mapping σ that maps a weighted formula to a conditional and is independent of the (size of) the universe. Then $\sigma((A(c), w_2))$ is the same with respect to changes to the universe and so is $P(\sigma((A(c), w_2)))$ for every model P of the corresponding RPCL knowledge base. Consequently, RPCL can also not be used to model MLNs.

The sharp separation of generic and specific knowledge as required in the expressivity analysis proposed in [32] is problematic since it prohibits a modeling taking into account both types of knowledge in the form as it is done for instance in the example on elephants and their keepers. Further investigations on how to state and how to compare the expressive power of probabilistic relational languages is needed.

4.2 Strict and Propositional Knowledge

In any probabilistic relational modeling language two essential dimensions are present that distinguish the respective approach from propositional logic: The *probabilistic* and the *relational* dimension. Therefore, from a knowledge representation point of view, the following questions arise naturally. What happens if one cuts down any of these two dimensions? Which kind of logic does one obtain?

(SP-1) Strict Knowledge: Suppose one restricts the sentences occurring in a knowledge base such that only strict probabilistic knowledge can be expressed. What is the representation level of this degenerated case, and what are its semantics and inference properties? In particular, what is its relationship to classical non-probabilistic logic?

Among the formalisms BLP, MLN, and RPCL, only MLNs allow for existential quantifiers (which in the Alchemy system are replaced by corresponding finite disjunctions over instantiations with the elements of the underlying universe). Looking at the language of logical MLN formulas we thus have the same as first-order logic. In order to express that a particular formula F represents strict knowledge, the weight of F must be set to infinity [16]; in the MLN system Alchemy, the corresponding representation is to write down the formula as in "F.", i. e. without any weight. In this case, all possible worlds violating the strict formula are assigned zero probabilities by the MLN, and the probabilities of the satisfying worlds sum up to 1. Hence, the formulas that can be inferred with probability 1 from such an MLN knowledge base $\mathcal{F}$ containing only strict formulas together with a set of constants C are the same as the formulas that

can be derived from $\mathcal{F}$ in a classical way where the universe of the underlying signature is identified with C. However, note that inference in Markov logic requires that the set of constants C is finite. Therefore, this "classical inference" is restricted to finite domains and does not feature the full expressivity of first-order inference.

Strict knowledge can be expressed by a BLP containing only conditional probabilities with values 0 and 1. In this case, also BLP semantics and BLP inference match semantics and inference in first-order logic, see below. Similarly, a strict knowledge base in RPCL is also obtained by allowing just the two extreme probabilities 0 and 1. In the following, we will look at the relationship of the obtained logics to first-order logic for BLPs and RPCL in some more detail.

Let $FOL_\forall$ be the subset of first-order formulas that are quantifier-free but all variables are assumed to be universally quantified. For a set $\Phi \subseteq FOL_\forall$ and $\phi \in FOL_\forall$ let $\Phi \models \phi$ be the classical inference relation, i. e., $\Phi \models \phi$ is true if for all variable assignments VA the set Φ classically entails $VA(\phi)$. Let $\mathcal{B}$ be the set of BLPs containing only conditional probabilities with values 0 and 1. For a Bayesian clause $c = (H \mid B_1, \ldots, B_n)$ with $\mathsf{cpd}_c(\mathsf{true}, \ldots, \mathsf{true}) = 1$ ($\mathsf{cpd}_c(\mathsf{true}, \ldots, \mathsf{true}) = 0$) consider the transformation

$$\Gamma(c) = B_1 \wedge \ldots \wedge B_n \Rightarrow (\neg)H.$$

Then the set

$$\Gamma(\mathcal{B}) = \bigcup_{B \in \mathcal{B}} \bigcup_{c \in B} \{\Gamma(c)\}$$

is obviously a subset of $FOL_\forall$. It is easy to see that for $B \in \mathcal{B}$, a ground query $c = (H \mid B_1, \ldots, B_n)$ has probability 1 in B if and only if $\Gamma(B) \models \Gamma(c)$. However, note that $\Gamma(\mathcal{B}) \neq FOL_\forall$ as e. g. the formula $A \vee B \in FOL_\forall$—which is not a Horn clause—is not in the image of Γ. Consequently, the restriction of BLPs to strict knowledge neither represents full first-order logic nor $FOL_\forall$.

For strict RPCL we obtain an almost complete correspondence to $FOL_\forall$ (independently of the semantics actually used for RPCL). Let $\mathcal{R}$ be the set of RPCL knowledge bases where every conditional has probability 1 or 0. Let $\Phi \subseteq FOL_\forall$ be a set of formulas and consider the transformation Δ given via

$$\Delta(\Phi) = \{(\phi \mid \top)[1] \mid \phi \in \Phi\}$$

containing only strict formulas. Then obviously $\Delta(\Phi) \in \mathcal{R}$ and also (if Φ is consistent) $\Phi \models \phi$ if and only if $\Delta(\Phi) \models^{ME}_{\mathcal{G}_U} \Delta(\{\phi\})$ for every $\phi \in FOL_\forall$; analogous observations hold for averaging and aggregating semantics. Looking at the other direction, let $KB \in \mathcal{R}$ be a strict RPCL knowledge base and consider the transformation Λ given via

$$\Lambda(KB) = \{\neg A \vee B \mid (B \mid A)[1] \in KB\} \cup \{A \wedge \neg B \mid (B \mid A)[0] \in KB\} \quad .$$

Then clearly $\Lambda(KB) \in FOL_\forall$. If KB is consistent using grounding operator $\mathcal{G}_U$ then also $KB \models^{ME}_{\mathcal{G}_U} r$ if and only if $\Lambda(KB) \models \Lambda(\{r\})$ for all conditionals r with probability 1. Moreover, if KB is consistent with respect to averaging or aggregating semantics, the inference in KB and $\Lambda(KB)$ is the same

also for these semantics. However, for the strict RPCL knowledge base $KB' = \{(B \mid A)[1],\ (A \mid \top)[0]\}$ we observe that KB' has no models since a probability distribution P can satisfy a conditional $\{(B \mid A)[x]$ only if $P(A) > 0$, independently of the actual semantics, cf. (5). On the other hand, $KB'_{FOL_\forall} = \{\neg A \vee B,\ \neg A\}$ does have a model. Thus, reducing a conditional to material implication may not be adequate even in the case of only strict probabilistic conditionals, see also [5]. However, for the subset $\mathcal{R}^c \subseteq \mathcal{R}$ of consistent knowledge bases and the subset $FOL^c_\forall \subseteq FOL_\forall$ of consistent sets of formulas we obtain complete correspondences.

Likewise, we can look at the degenerated knowledge representation formalism obtained by cutting out any relational representation aspects.

(SP-2) Propositional Knowledge: What kind of logic does one obtain if a knowledge base contains only ground knowledge? What are its semantics and inference properties, and in particular, what is its relationship to propositional probabilistic logic?

Let BN be a Bayesian network. Then obviously B can be represented as a Bayesian logic program with predicates of arity zero. For the other direction, let B be a BLP where every Bayesian clause is ground. It may be the case that B contains multiple clauses with the same head. However, a combined conditional probability distribution for the joint conditional probability can be compiled directly by using the combining rules of the predicate of the head. Therefore B can be transformed into a BLP B' such that B' does not contain multiple clauses with the same head and B and B' have the same inference behavior. Then, obviously B' corresponds to a propositional Bayesian network.

If L is an MLN containing only ground atoms, then for any set C of constants the corresponding ground Markov net (see Definition 3) is independent of C. Consequently, L represents a unique probability function P and Equation (4) simplifies to

$$P_{M_{L,C}}(\omega) = \frac{1}{Z} \exp\left(\sum_{(F_i,w_i)\in L, \omega \models F_i} w_i\right)$$

with the normalization factor

$$Z = \sum_{\omega\in\Omega} \exp\left(\sum_{(F_i,w_i)\in L, \omega \models F_i} w_i\right).$$

and is therefore equivalent to a propositional Markov net.

For a ground RPCL knowledge base, grounding, averaging, and aggregating semantics coincide with classical probabilistic semantics in probabilistic conditional logic [61, 36] and inference based on the principle of maximum entropy is the same as in the propositional case, cf. [67].

4.3 Individuals and Universes

The core idea of relational knowledge representation is to talk about a set of elements (a *universe*) and the relations among them. Thus, methods are needed for specifying elements belonging to the universe, to refer to elements in the universe, and to reason about elements and their properties and relationships. In general, relational approaches may differ according to whether and how they support any of the following criteria.

(U-1) Listing of elements: Can (or must) a universe be specified by explicitly listing all its elements?

The given facts in a BLP must all be ground; they determine the specific context of the BLP, thus allowing to list all elements of a universe by mentioning them in the atoms of the BLP. So the constants of a BLP are implicitly introduced by respective ground atoms occurring in the Bayesian clauses of the knowledge base or in a distinct query. When defining an MLN, an explicit listing of all constants C must be given, and the semantics of an MLN requires that different constants denote different elements and that there are no elements other than the ones denoted by constants. Although it is possible [59] to extend the definition of an MLN in order to allow different constants to stand for the same element, the set of all constants must be explicitly listed and each element of the universe has to be denoted by at least one constant. Similarly, all constants in an RPCL knowledge base denote different elements, and there are no other elements.

(U-2) Open universe: Is it possible to have an *open* universe whose number of elements is not a-priori known?

In BLP, MLN, and RPCL it is not possible to specify such open universes directly. Just to the contrary, the grounding of formulas with constants from the universe is an essential part of each formalism. For instance, the constants occurring in a query Q together with the constants in a BLP P determine the Herbrand universe used to construct the ground Bayesian network for answering Q. However, in all approaches the extensional part—i. e. the ground atoms resp. the given constants—can be exchanged while reusing the given generic knowledge.

(U-3) (Proto)Typical elements: Can (proto)typical elements within a universe be specified or identified?

A universally quantified variable X in a relational statement expresses that this statement applies to all elements of the considered universe. However, as the example on elephants and keepers demonstrates, there is the need to also express knowledge about individuals, referred to by specific constants; in any of the five approaches, generic statements using variables may be combined with statements about individuals. In the elephant-keeper example, asking about a keeper jim will return the same probability as asking the same question about a keeper tom since the respective knowledge bases do not contain any specific

information neither about *jim* nor about *tom*. So, besides named individuals which are mentioned in the knowledge base, the universe will usually contain individuals on which no specific information is explicitly expressed and which hence may serve as *typical* individuals. More precisely, let $C_{\mathcal{R}}$ be the set of constants occurring in a set of rules $\mathcal{R}$ and let C_U be the set of all constants under consideration. (Note that for MLN and RPCL, C_U is given explicitly, and that for a BLP, C_U is determined when a query is posed.) Then the elements in $C_{typical} = C_U \backslash C_{\mathcal{R}}$ cannot be distinguished by any query asked with respect to $\mathcal{R}$: If $d_1, d_2 \in C_{typical}$ and Q is a query containing d_1, then the query Q' obtained from Q by replacing d_1 by d_2 (and possibly also d_2 by d_1) yields the same probability as Q. This observation holds for all of the five approaches. So, if typicality is interpreted in terms of being indistinguishable and least specific, then all five approaches can identify and represent typical elements. On the other hand, no approach supports the immediate identification of prototypical individuals as most appropriate personification of a given concept. For this, criteria to make a qualitative distinction between individuals would be required which are not provided by any of the approaches. In the context of qualitative default reasoning, an approach similar to probabilistic aggregation semantics has been presented recently which provides further properties that allow to identify prototypical individuals (for more details, please see [42]).

(U-4) Inference for individuals: There should be a well-defined inference mechanism to infer probabilities for particular individuals (either prototypical individuals or specific, named individuals). Does such inference depend on the number of elements in a universe, and if so, what is the dependency?

Obviously, all approaches provide for querying about specific individuals. For example, given a BLP, a ground Bayes net can be constructed to infer probabilities for some ground query involving arbitrary constants. Similarly, this holds for MLNs and the approaches based on maximum entropy. Further, the number of elements in the universe might influence the probability of a query in all approaches. Consider the BLP B containing the clauses $(D(X) \mid A(X,Y))$ and $(A(X,Y))$. Given the query $D(c)$ for some constant c the probability of $D(c)$ depends on the number of instances of $D(c,Y)$, i. e., on the number of constants in the universe. If *noisy-or* is the combining rule for D then the probability of $D(c)$ tends towards one when the number of constants in the universe tends towards infinity, independently of the actual conditional probability distributions of $(D(X) \mid A(X,Y))$ and $(A(X,Y))$. A similar observation can be made for MLNs and RPCL.

Another dependency of the number of elements in the universe and probabilities of queries arises for RPCL under averaging and aggregating semantics. Consider now the conditional $(D \mid A)[x]$ and averaging semantics. If $(D' \mid A')$ is an instance of $(D \mid A)$ that does not mention any constants in the knowledge base then it is easy to see that the probability of $(D' \mid A')$ tends towards x if the number of elements in the universe tends towards infinity, cf. [41].

(U-5) Grounding: Is there a mechanism for (consistent) grounding of a knowledge base?

The semantics of a BLP or an MLN knowledge base is defined via complete groundings with respect to a given finite universe L of constants, yielding a (ground) Bayesian network or a (ground) Markov net, respectively. In a BLP, the logic part consists of Horn clauses which do not allow the specification of negated conclusions, so that inconsistencies on the logical level are avoided. Conflicting specifications on the quantitative level may arise e. g. when having syntactical variants of a clause such as $(B(X) \mid A(X))$ and $(B(Y) \mid A(Y))$ with different cpd's. Such conflicts are resolved via the combining rules like *noisy-or* (cf. Sec. 2.2).

An MLN might contain both $(F,\, w)$ and $(\neg F,\, w)$, thus assigning equal weights to the contradicting formulas F and $\neg F$. Due to the specification of the probability distribution depending on the weights given in an MLN (cf. Equation. (4)), the grounded MLN semantics is still consistent and well defined.

Also for RPCL, a universe L of constants must be given. As mentioned before, naive complete grounding with respect to L might generate an inconsistency, yielding a situation where the instantiated knowledge base does not have any model.

Another important aspect connected to the notion of relational knowledge and universes is the question of whether probabilities are interpreted statistically or as subjective degrees of belief, cf. the discussion in the context of (L-4).

After our discussion on formal criteria on relational probabilistic models, we continue with investigating the practical side of contrasting relational probabilistic approaches by presenting various systems and applications.

5 Implementations for Relational Probabilistic Knowledge Representation

Relational probabilistic models such as BLPs, MLNs, and RME are quite complex models and formal comparisons are not always sufficient for an in-depth understanding and evaluation. An important aspect for comparison is an analysis of the computational behaviors of the models on e. g. benchmark examples. Furthermore, while prototypical implementations of specific approaches to relational probabilistic knowledge representation (and approaches for any problem in general) are essential for validating results and evaluation, these software solutions are often very hard to use and differ significantly in their usage. The KReator system provides a common interface to different formalisms for relational probabilistic models and we continue by giving a brief overview on its architecture and usage.

5.1 The KReator System

KReator is an integrated development environment for representing, reasoning, and learning with relational probabilistic knowledge.[3] KReator aims at providing a versatile toolbox for researchers and knowledge engineers in the field of statistical relational learning. KReator's intuitive graphical user interface provides an easy access to its core functionalities, e. g., to specify knowledge bases, ask queries to knowledge bases, and to learn knowledge bases from data (see Fig. 3 for a screenshot of the KReator user interface). KReator supports these tasks for different knowledge representation formalisms by a flexible plugin architecture. A plugin for a specific formalism encapsulates all its data structures and algorithms and connects them by a standardized interface to the KReator framework. At this time, plugins for Bayesian logic programs, Markov logic networks, relational maximum entropy, as well as Relational Bayesian Networks [31, 10] and Probabilistic Prolog [58] are already supplied with KReator. Plugins for other formalism, e. g. P-log [1] and PRL [28], are currently under development.

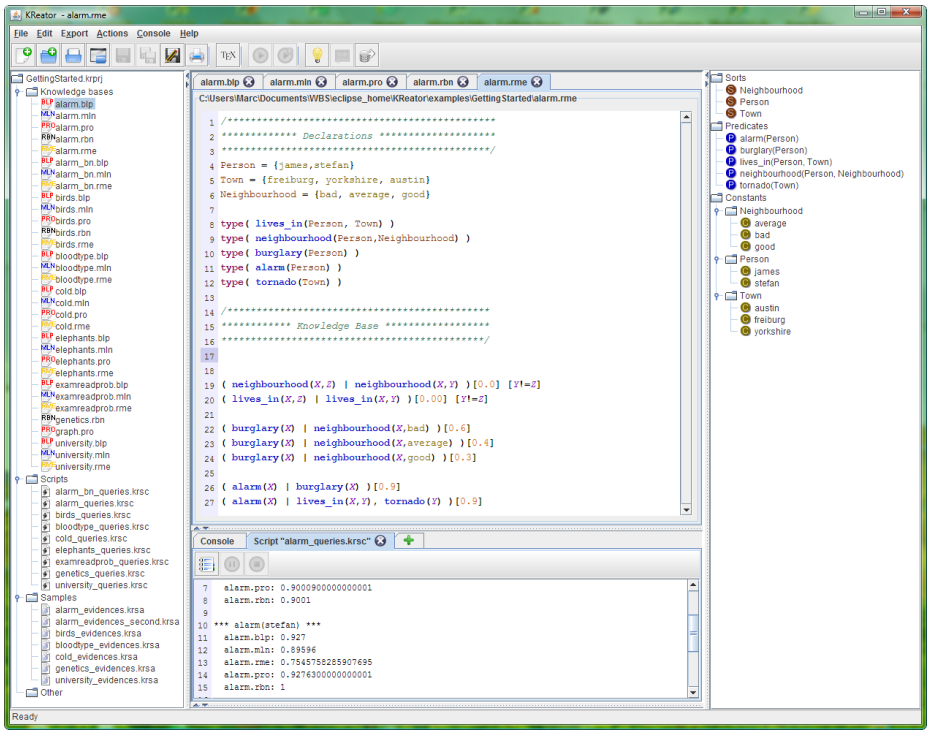

Fig. 3. Graphical user interface of the KReator system

[3] The development of KReator is part of the KReate project, cf. `www.fernuni-hagen.de/wbs/research/kreate/`

Some of these KReator plugins make use of other software systems to accomplish their tasks: Performing inference on MLNs is done using the Alchemy software package 2 [45], a console-based tool for processing Markov logic networks. To process ground RME knowledge bases, KReator uses a so-called ME-adapter to communicate with a MaxEnt-reasoner. Currently, such adapters are supplied for the SPIRIT reasoner [63] and for MEcore [20] which are tools for processing propositional conditional probabilistic knowledge bases using maximum entropy methods (see Sec. 5.3 for a description of these systems).

The main advantage of using KReator (instead of using prototypical implementations of each formalism) is to address knowledge engineering tasks to different formalisms in a common and unified way. Besides this, KReator offers several convenience features, e. g. project management, scripting, syntax highlighting, or LaTeX output, which ease the work on knowledge representation, reasoning, and learning.

Since KReator is written in Java [30], it is platform independent, i. e. it runs on any system with a Java runtime environment. KReator's design follows the object-oriented programming paradigm and facilitates several architectural and design patterns [26] such as model-view control, abstract factories, multithreading, and command patterns. Central aspects of the design of KReator are *modularity*, *extensibility*, *usability*, and its intended application in scientific research; see [22] for a thorough discussion.

5.2 The ProbCog System

To our knowledge, there is only one software system which takes an approach comparable to KReator in the sense that it combines different formalisms within one system. The *ProbCog* (Probabilistic Cognition for Cognitive Technical Systems) system[4] [33] is a software suite for statistical relational learning. ProbCog currently supports three knowledge representation approaches: Bayesian Logic Networks (BLNs), Adaptive Markov Logic Networks (AMLNs), and Markov Logic Networks (MLNs). For each approach, ProbCog provides several learning and inference algorithms, implemented either in Java or Python. ProbCog provides a sophisticated framework for relational data, which features, amongst others, a unified data model (which allows data conversion for all integrated approaches) and the generation of synthetic data (for learning experiments). The main focus of the ProbCog suite is on providing a comprehensive library of algorithms and powerful data structures for statistical relational learning, but it also includes some graphical interfaces for learning and querying, respectively.

ProbCog and KReator share some similarities with respect to their general approach to gather different knowledge representation approaches within one software framework, e. g. both systems feature some sort of unified data model for evidence or sample data. But the primary application focus of both systems differs significantly: ProbCog is developed for its intended practical application

[4] `http://ias.cs.tum.edu/research-areas/knowledge-processing/probcog`

and integration in cognitive technical systems. So its primary focus is on providing a versatile and efficient framework for that specific purpose, therefore some sort of unified graphical user interface to the framework is not needed. In contrast, KReator's focus is on the typical workflow of a knowledge engineer, researcher, or developer. Therefore, KReator gathers different approaches in an integrated graphical development environment to provide easy access to typical tasks and provides a plugin interface to support the study and development of further approaches.

5.3 Related Systems

For many knowledge representation formalisms, there exist prototypical software implementations, where each particular implementation is specialized for a certain formalism, e. g. Balios[5] for BLPs, Alchemy for MLNs, or Primula[6] for relational Bayesian networks.

Reasoning in probabilistic conditional logic by employing the principle of maximum entropy [54, 35] requires solving the numerical optimization problem $P^* = \arg\max_{P \models \mathcal{R}} H(P)$ (cf. Sec 2.1). **SPIRIT** [63] is an expert system shell[7] implementing maximum entropy reasoning and solving this optimization problem. In order to tame the complexity of the optimization task which grows exponentially in the number of variables, SPIRIT generates a junction-tree of variable clusters, allowing to represent the global probability distribution by a product of marginal distributions. SPIRIT has been used successfully in various application domains, like medical diagnosis, project risk management, or credit scoring. Apart from its graphical user interface, SPIRIT also features a software interface (in terms of a Java API) which allows to access its functionalities from external programs.

MEcore [20] is another system implementing reasoning for propositional probabilistic conditional logic under maximum entropy. While it does not employ a junction-tree modelling, but a straight-forward representation of the complete probability distribution, its focus is on flexibly supporting different basic knowledge and belief management functions like revising or updating probabilistic beliefs, or hypothetical reasoning in what-if mode. All functionality of MEcore can be controlled by a text command interface or by script files containing command sequences (see Sec. 6.1 for some practical applications). MEcore features an expressive command language which allows, e. g., to manipulate knowledge bases, and to automate sequences of updates and revisions. Besides this, a Java software interface allows to integrate MEcore in other programs.

The **Alchemy** system [8] [45] implements Markov logic. Alchemy provides a wide range of functionalities for statistical relational learning and probabilistic logic inference. In particular, the consequences of a Markov logic network L

[5] `http://people.csail.mit.edu/kersting/profile/`
[6] `http://www.cs.aau.dk/~jaeger/Primula/index.html`
[7] `http://www.fernuni-hagen.de/BWLOR/spirit/index.php`
[8] `http://alchemy.cs.washington.edu/`

defined via the ground Markov network $M_{L,C}$ (cf. Sec. 2.3) can be determined. With respect to learning, both weight learning as well as learning the structure of an MLN is supported. Applications of MLN realized with Alchemy include classification tasks and social network modelling. In Sec. 6, we will report on some experiments using MLNs and Alchemy in medical diagnosis.

6 Applications

In the following subsections, we will present three practical application scenarios of some of the afore described systems. All three applications cover settings from the medical domain. The first one illustrates ME-reasoning using a fictitious example, whereas the other ones describe learning experiments involving Markov logic networks and real-world data from medical studies.

6.1 Knowledge Processing with the MEcore system

In this section, we will illustrate how the MEcoRe system [20] (cf. Sec. 5.3) can process incomplete, uncertain knowledge expressed by a probabilistic knowledge base using a fictitious example from the medical domain. This example is taken from [20] and discusses the general treatment of a patient who suffers from a perilous bacterial infection. The infection will probably cause permanent neurological damage or even death if it is not treated appropriately. There are two antibiotics available that might be capable of ending the infection, provided that the bacteria are not resistant to the specific antibiotic. It must also be considered that each antibiotic might cause a life-threatening allergic reaction that could be especially dangerous for an already weakened patient. The resistance of the bacteria to a specific antibiotic can be tested, but each test is very time-consuming.

Building Up the Knowledge Base The construction of the knowledge base starts with the definition of some binary variables that describe aspects concerning antibiotic A:

`med_A`: The patient is treated with antibiotic A.
`effect_A`: Antibiotic A is effective against the bacteria.
`allergic_A`: The patient is allergic to antibiotic A.
`resistance_A`: The bacteria are resistant to antibiotic A.
`posResT_A`: The test result suggests a resistance to antibiotic A.

Analogously, there are also five variables concerning antibiotic B. A three-valued variable `outcome` describes the three possible outcomes of the treatment:

`outcome=healthy`: The infection is treated successfully and the patient is healthy again.
`outcome=impaired`: The patient overcomes the infection but suffers a permanent damage to the nervous system.
`outcome=dead`: The infection is not treated effectively and the patient dies.

The available knowledge summarizing the previously made experiences about the infection and the two antibiotics is modeled by the knowledge base $\texttt{medKB} = \{R_1, \ldots, R_{22}\}$ consisting of the probabilistic rules given in Fig. 4.

R_1 : $(\neg\texttt{effect\_A} \mid \neg\texttt{med\_A} \vee \texttt{resistance\_A})[1.00]$
R_2 : $(\neg\texttt{effect\_B} \mid \neg\texttt{med\_B} \vee \texttt{resistance\_B})[1.00]$
R_3 : $(\texttt{effect\_A} \Leftrightarrow \texttt{med\_A} \mid \neg\texttt{resistance\_A})[1.00]$
R_4 : $(\texttt{effect\_B} \Leftrightarrow \texttt{med\_B} \mid \neg\texttt{resistance\_B})[1.00]$
R_5 : $(\texttt{allergic\_A})[0.10]$
R_6 : $(\texttt{allergic\_B})[0.20]$
R_7 : $(\texttt{resistance\_A})[0.01]$
R_8 : $(\texttt{resistance\_B})[0.09]$
R_9 : $(\texttt{med\_A} \wedge \texttt{med\_B})[0.00001]$
R_{10}: $(\texttt{outcome} = \texttt{dead} \mid \neg\texttt{med\_A} \wedge \neg\texttt{med\_B})[0.10]$
R_{11}: $(\texttt{outcome} = \texttt{healthy} \mid \neg\texttt{med\_A} \wedge \neg\texttt{med\_B})[0.10]$
R_{12}: $(\texttt{posResT\_A} \mid \texttt{resistance\_A})[0.97]$
R_{13}: $(\neg\texttt{posResT\_A} \mid \neg\texttt{resistance\_A})[0.99]$
R_{14}: $(\texttt{posResT\_B} \mid \texttt{resistance\_B})[0.90]$
R_{15}: $(\neg\texttt{posResT\_B} \mid \neg\texttt{resistance\_B})[0.80]$
R_{16}: $(\texttt{outcome} = \texttt{dead} \mid \texttt{med\_A} \wedge \texttt{allergic\_A})[0.99]$
R_{17}: $(\texttt{outcome} = \texttt{dead} \mid \texttt{med\_B} \wedge \texttt{allergic\_B})[0.40]$
R_{18}: $(\texttt{outcome} = \texttt{healthy} \mid \texttt{effect\_A})[0.8]$
R_{19}: $(\texttt{outcome} = \texttt{healthy} \mid \texttt{effect\_B})[0.7]$
R_{20}: $(\texttt{allergic\_A} \mid \texttt{med\_A})[0.10]$
R_{21}: $(\texttt{outcome} = \texttt{dead} \mid \texttt{effect\_B})[0.09]$
R_{22}: $(\texttt{outcome} = \texttt{healthy} \mid \texttt{med\_B} \wedge \texttt{allergic\_B})[0.001]$

Fig. 4. Probabilistic rules in the knowledge base medKB

The first four rules express very obvious correlations between the variables: R_1 and R_2 say that if a certain antibiotic is not administered or the bacteria are resistent to it, then this antibiotic has no effect. R_3 and R_4 assure that if the bacteria are not resistant to a certain antibiotic, then this antibiotic is effective if—and only if—it is administered. The facts R_5 to R_9 integrate statistical information available for antibiotic A and antibiotic B, i. e. some a priori probabilities, into the knowledge base: antibiotic B is twice as likely as antibiotic A to cause an allergic reaction (R_5, R_6); and the resistance to antibiotic B is nine times higher compared to antibiotic A (R_7, R_8). It has occurred very rarely that somebody administers both antibiotics to the patient (R_9). R_{10} and R_{11} model the prognosis for the patient if no antibiotic is administered. The result of a resistance-test, testing the resistance of the bacteria to an antibiotic, always

includes some error, but the test regarding antibiotic A is very reliable (R_{12}, R_{13}); whereas the test concerning antibiotic B has a somewhat lower sensitivity (R_{14}) and a considerably lower specificity (R_{15}).

The rules R_{16} to R_{19} express special knowledge about antibiotic A and antibiotic B, respectively: The allergic reaction caused by antibiotic A is most likely lethal (R_{16}), whereas the chance of surviving an allergy to antibiotic B is more likely than to die of it (R_{17}). If antibiotic A is effective, then the patient has a good chance to become healthy again (R_{18}), whereas the effectiveness of antibiotic B is somewhat lower (R_{19}). The following knowledge is available for antibiotic A only: R_{20} makes clear that the a priori probability of an allergy to antibiotic A (expressed by R_5 with equal probability) is not affected by the administration of antibiotic A. There is also some exclusive knowledge about antibiotic B: If antibiotic B is effective, there still remains some risk to die of the infection (R_{21}). If the patient survives an allergic reaction caused by antibiotic B, it is very unlikely that he will become healthy again (R_{22}).

Computing an Initial Epistemic State In MEcore, the computation of an epistemic state incorporating the knowledge expressed by the knowledge base `medKB` can be initiated by the command:

(1) `currState` := `epstate.initialize(medKB)`;

The calculated epistemic state `currState` represents the incomplete knowledge expressed by `medKB` inductively completed in an entropy-optimal way.

A closer look at `medKB` reveals that some additional rules can be logically deduced from the existing rules since they hold in all models satisfying `medKB`. For instance, a literal of the three-valued variable `outcome` makes up the conclusion of several rules. Hence, two rules with identical premise and an `outcome` literal as conclusion directly imply a corresponding third rule, e. g. R_{10} and R_{11} imply $(\texttt{outcome} = \texttt{impaired} \mid \neg\texttt{med\_A} \wedge \neg\texttt{med\_B})[0.8]$. Appropriate queries to MEcore in `currState` yield these expected probabilities since reasoning at optimum entropy is compatible with classical probabilistic consequences.

Query Suppose we want to know the patient's chances in each case of treatment, i. e. for each of the four possible options of medical administration: no antibiotic, antibiotic A only, antibiotic B only, both antibiotics. This can be expressed by a set of twelve query formulas (i. e. conditionals of the form e. g. $(\texttt{outcome} = \texttt{healthy} \mid \texttt{med\_A} \wedge \neg\texttt{med\_B})$) which we will denote by `medQueries`. While using classical probabilistic consequences does not yield informative answers for `medQueries`, MEcore infers the following probabilities from `currState`:

	healthy	impaired	dead
no antibiotic	0.10	0.80	0.10
only A	0.79	0.06	0.15
only B	0.65	0.23	0.12
A and B	0.94	0.02	0.04

These results clearly suggest that the combined administration of both antibiotics would be the best treatment. It offers a high chance of healing accompanied by a minimal risk of permanent neurological damage or death. However, a closer look at the knowledge base reveals that it implies that there is almost no possible drug interaction. For instance, asking for the degree of belief for the conditional

$$\mathtt{C_{int}} : (\mathtt{dead} \mid \mathtt{med\_A} \wedge \mathtt{med\_B} \wedge \neg\mathtt{allergic\_A} \wedge \neg\mathtt{allergic\_B})$$

in `currState` yields the inferred drug interaction probability 0.01.

Incorporation of New Knowledge Suppose that later on, the doctors learn from an outside source that there is a severe risk (0.25) of a deadly drug interaction between both antibiotics. Executing

(2) `currState.update(medKB,` $\mathtt{C_{int}}[0.25]$`);`

incorporates this new knowledge into the current epistemic state as if it had been available already in `medKB`. In fact, this kind of belief change is a *genuine revision* (cf. [37]) which in MEcoRe can also be more easily expressed by

(2') `currState.revise(`$\mathtt{C_{int}}[0.25]$`);`

Now, asking the `medQueries` again, the probabilities have changed considerably (cf. Fig. 5(a)): With the knowledge about a deadly drug interaction, the probabilities show that the administration of antibiotic A maximizes the patient's chance to become healthy again.

(a)	healthy	impaired	dead
no antibiotic	0.10	0.80	0.10
only A	0.79	0.06	0.15
only B	0.65	0.23	0.12
A and B	0.70	0.02	0.28

(b)	healthy	impaired	dead
no antibiotic	0.10	0.80	0.10
only A	0.79	0.06	0.15
only B	0.69	0.21	0.10
A and B	0.76	0.02	0.22

(c)	healthy	impaired	dead
no antibiotic	0.10	0.80	0.10
only A	0.43	0.15	0.42
only B	0.65	0.23	0.12
A and B	0.32	0.05	0.63

(d)	healthy	impaired	dead
no antibiotic	0.10	0.80	0.10
only A	0.43	0.15	0.42
only B	0.54	0.26	0.20
A and B	0.20	0.04	0.76

Fig. 5. Probabilities for `medQueries` inferred by MEcoRe

What-If-Analysis It has to be noticed that the knowledge used for generating the epistemic state `currState` says that no resistance tests have been performed, i. e. for neither of the antibiotics any resistance test results are available. A what-if-analysis can be used to analyze what changes would occur if a negative resistance-test result concerning antibiotic B was known. That is, could this test

result make antibiotic B the better choice for treatment? In MECoRe, such a what-if-analysis is accomplished by

(3) `currState.whatif((¬posResT_B)[1.0],medQueries);`

delivering the results shown in Fig. 5(b). The probabilities show that even a negative resistance-B test would not change the general decision to administer antibiotic A. This result is, amongst others, caused by the low resistance-B test specificity.

Another what-if-analysis revealing the effects of a positive resistance-A-test

(4) `currState.whatif((posResT_A)[1.0],medQueries);`

yields the probabilities given in Fig. 5(c). This shows that a test-result suggesting the resistance to antibiotic A would change the situation: In this case, a treatment with antibiotic B becomes the only choice that offers a realistic healing chance. This is not surprising, because a resistance-test result concerning antibiotic A is very reliable. So it is clearly advisable to perform the time-consuming resistance-A test.

In case of a positive resistance-A-test result, would it also be helpful to test the resistance to antibiotic B? That is, could an additional positive resistance-B-test change the decision to administer antibiotic B? Hypothetical reasoning

(5) `currState.whatif(((posResT_A)[1.0],(posResT_B)[1.0]),medQueries);`

yields the results shown in Fig. 5(d), indicating that even a positive resistance-B-test would not change the decision to administer antibiotic B. So it is not helpful to perform a resistance-B test in any situation, since its result would never change the decision that had been made without knowing the test result.

6.2 Diagnosis of Lung Cancer

This section is based on [21], reporting on a case study of using probabilistic relational modelling and learning as provided by MLNs and the MLN system Alchemy [45] (cf. Sec. 5.3) in the field of biomedical diagnosis. We employ different algorithms to learn MLNs from sample data. Each MLN induces a probabilistic model and thereby allows probabilistic reasoning. The idea behind this diagnostical setting is to support diagnosis of bronchial carcinoma on the basis of the substances a person exhales [3, 2]. In this setting, the focus is on the early detection of bronchial carcinoma by ion mobility spectrometry, a non-invasive diagnostic method which delivers results within a few minutes and can be applied at low costs.

Ion Mobility Spectrometry In order to determine chemical substances in gaseous analytes, ion mobility spectrometry (IMS) can be used [3]. This method relies on characterizing substances in gases by their ion mobility. After ionisation, ion swarms enter the drift region through an ion shutter. The time needed to pass the drift region is called *drift time*, and the ion mobility is inversely proportional to the drift time. An ion mobility spectrum is obtained by mapping the drift

time to the signal intensity measured at the Faraday plate. If the gaseous analyte contains various substances, they may reach the Faraday plate at the same time. In order to avoid this, a multi capillary column is used for the pre-separation of different substances [3] so that they enter the spectrometer at different time points, called *retention times*; for more detailed descriptions of ion mobility spectrometry and its working principle we refer to [2] or [3].

Thus, applying ion mobility spectrometry to gaseous analytes yields IMS spectra where a peak in such a spectrum corresponds to a particular substance. The determination of peaks in a measurement requires sophisticated processing of the raw spectra, see [3, 6] for details. Peak objects taken from two different measurements that correspond to the same substance occur at corresponding areas in their respective so-called *heat maps*, and in order to identify such corresponding peaks, they can be mapped to *peak clusters* [3, 21]. In our case study, we investigated an IMS database consisting of 158 measurements obtained from screening the breath of 158 patients out of which 82 had lung cancer (*bronchial carcinoma, bc*), yielding a database D_{bc} with 33 peak clusters, in the following referred to by the identifiers $pc0, \ldots, pc32$. For each peak cluster pc_i, $P(bc|pc_i)$ denotes the conditional probability that a measurement having a peak belonging to pc_i stems from a person having bronchial carcinoma. For applying methods of probabilistic relational modelling and learning to D_{bc}, we use a logic representation of D_{bc} (for convenience, also referred to as D_{bc}) involving the predicates $bc(M)$ indicating that measurement M belongs to a person having lung cancer and $pcInM(PC, M)$ stating that peak cluster PC occurs in measurement M.

In the following, we present different setups to learn MLNs from the data set D_{bc}. Our goal is to calculate the probability that a certain measurement m is from some person with a bronchial carcinoma, given the information for each of the 33 peak clusters whether or not it is contained in measurement m. That is, we want to calculate the conditional probability of $bc(m)$, given the truth values of the literals $pcInM(pc0, m)$, ..., $pcInM(pc32, m)$. This conditional probability helps to classify patients with respect to suffering from lung cancer. The corresponding classification task can be realized with MLNs. We use the software package Alchemy [45] which provides several sophisticated algorithms to perform (structure and parameter) learning and inference of MLNs. A learned MLN is validated in terms of classification accuracy, defined as the proportion of the correctly predicted (positive and negative) results on the total number of measurements in a testing set; these values are determined as the average accuracy of all tests in a 10-fold cross-validation.

Learning Logic Rules with the ILP System Aleph In a first learning setup, we use the inductive logic programming (ILP) system *Aleph* [65] for learning first-order logic rules from the data set. Besides other parameters, Aleph allows to make detailed specifications about which atoms may appear in the body or head of a rule. As we want to predict whether or not the measurement M belongs to a patient having bronchial carcinoma, we require that heads of the rules learned by Aleph must contain the *bc* predicate, whereas their body must consist of one

or more atoms of the *pcInM* predicate, with a constant in the first argument. This way, the rules predict the value of $bc(M)$, given the values of some of the $pcInM(pc_i, M)$. The two rules

$$R_1\colon\ pcInM(pc5, M) \wedge pcInM(pc8, M) \Rightarrow bc(M)$$
$$R_2\colon\ pcInM(pc7, M) \wedge pcInM(pc17, M) \wedge pcInM(pc31, M) \Rightarrow bc(M)$$

are examples of the 11 rules learned with Aleph [21]. The premises of all 11 rules consist of conjunctions of at most three positive *pcInM* literals. From the 33 different peak clusters found in the data set, only 18 occur in the rule set, so the other 15 peak clusters seem to carry no useful information with regard to lung cancer according to the Aleph result.

Learning Weights of Aleph Formulas with Alchemy In a subsequent step, we take the Aleph implications as logical base structure of an MLN and learn appropriate weights for them from the data set using Alchemy. For instance, the resulting weights for the rules R_1 and R_2 above are 4.596 and 6.004, respectively. Evaluating the MLN prediction performance results in an accuracy of 78%.

If we take the implications as if-then-rules, we can determine the conditional probabilities of these rules under the distribution induced by the MLN, i. e. we use Alchemy to calculate the conditional probability of a rule's consequent ground atom given its premise ground atoms as evidence. E. g., for rule R_1, Alchemy determines the probability $P(bc(m)|pcInM(pc5, m) \wedge pcInM(pc8, m)) = 0.9800$ in the MLN; for R_2 we get 0.996. In fact, the conditional probabilities of all rules are not exactly 1.0, as expected, but rather close to it (see [21]). This is due to the fact that Alchemy performs approximate inference.

The learned MLN allows to draw some conclusions between peak clusters (i. e. the occurrence of substances in a measurement) and bronchial carcinoma. E. g., formula R_2 relates the combined occurrence of peak clusters *pc7*, *pc17*, and *pc31* in a measurement M to the presence of bronchial carcinoma. Because of the positive (and relatively high) weight of this formula, the combined occurrence of these peak clusters can be interpreted as an indicator for bronchial carcinoma. Likewise, there are also formulas relating the combined occurrence of peak clusters to the absence of bronchial carcinoma.

Simple Classification with MLNs In a further learning setup, we predefine the formula structure of a quite simple MLN: The MLN consists of the 33 implications $pcInM(pc0, M) \Rightarrow bc(M), \ldots, pcInM(pc32, M) \Rightarrow bc(M)$. Since the Alchemy syntax allows to express such "partially grounded" formulas in a compact way, the whole predefined structural Alchemy input merely consists of a single line. With this MLN structure, we follow a straightforwardly modelled classification approach: To classify the *bc* state of a measurement, we consider each peak cluster separately, leaving out any connections or dependencies among them. To some extent, this approach resembles Naive Bayes classification, where explicit independence assumptions among classifying attributes are made. The

evaluation of the learned MLN revealed quite a high accuracy of 88% [21], although the enforced MLN structure lacks any connections between peak clusters, suggesting that those connections are not of great importance for classifying the measurements regarding *bc*.

MLN Structure Learning In this learning setup, we make use of Alchemy's structure learning feature to learn an MLN from scratch. Alchemy does not allow to make detailed specifications about the formulas to be learned, i. e. we cannot impose the requirement that the $pcInM(\_, \_)$ atoms have a constant in the first argument. As a consequence, Alchemy's structure learning algorithm produces no useful results when applied to D_{bc} without any further information. So we modify the relational modelling in some aspect by replacing the binary predicate $pcInM(PC, M)$ by 33 unary predicates $pc0(M), \ldots, pc32(M)$.

Using this setup, the structure (and weight) learning with Alchemy starts from an empty MLN and computes within a few minutes an MLN with 89 formulas (including 34 atomic formulas for all 34 predicates) [21]. The evaluation of this MLN shows an accuracy of 90%. Compared to the previous results, this MLN models much more connections among the peak clusters and their combined influence regarding $bc(M)$. Only 13 of the 55 non-atomic formulas involve a *bc* literal, so the other 42 formulas express connections among the peak clusters regardless of the $bc(M)$ state, and the formulas contain both positive and negative peak cluster literals. Consequently, this MLN exhibits more complex and subtle connections among the occurrences of peak clusters and the $bc(M)$ state. Here are two examples for the learned formulas:

$$R_{61}\colon\ (\neg pc10(M) \wedge pc14(M) \wedge \neg pc18(M) \wedge pc21(M) \Rightarrow bc(M), \quad 7.15)$$

$$R_{44}\colon\ (pc17(M) \wedge pc28(M) \Rightarrow pc21(M), \quad 5.05)$$

R_{61} relates the combined occurrence of peak clusters *pc14* and *pc21* and the explicit absence of peak clusters *pc10* and *pc18* in a measurement to bronchial carcinoma. With a lower, but still relatively high weight, R_{44} implies that a measurement containing peak clusters *pc17* and *pc28* also contains peak cluster *pc21*. In other words, the system has learned the relationship that the occurrence of the two substances indicated by peak clusters *pc17* and *pc28* in a measurement M leads to the presence of the substance identified by *pc21* in the same measurement. Such a relation can provide interesting insights into the general composition of substances in typical measurements.

6.3 Predicting Allergic Diseases of Children

In this section, another application of MLNs for modelling and learning in the medical domain is presented. In [64], MLNs were employed to analyze the correlations between allergic diseases of children and certain environmental factors. The data used in this analysis has been extracted from the KiGGS study of the Robert Koch-Institut [60]. The KiGGS study is a long term study which covers the health situation of 17.000 children (and adolescents) in Germany. It considers

a multitude of attributes for every child concerning medical or social aspects. For the experiments described in [64], 13 of these attributes had been chosen which represent well-known risk factors for allergies, e. g. *"the child has a pet at home"*, *"the child lives in an urban environment"*, or *"a parent suffers from an allergy"*. Each such attribute was modelled by a corresponding MLN predicate, e. g. $hasPet(X)$, $urban(X)$. Together with the information whether or not a child is allergic (represented by an $isAllergic(X)$ predicate) this allowed to model the data from the study as MLN learning data, i. e. as data samples in terms of ground atoms. The extracted and preprocessed learning data from the study consisted of about 8.000 data samples, covering allergic respectively non-allergic children in equal parts. In all experiments, subsets of these data samples were used as actual training and testing data (performing a 5-fold cross-validation).

Several learning experiments were performed on this learning data using the algorithms of the Alchemy software package [45] (cf. Sec. 5.3) for learning and inference. The goal of all experiments was to learn an MLN which can predict the risk of a child to be allergic given the presence (or absence, respectively) of each of the 13 risk factors. The learning experiments included parameter (i. e. weight) learning using a predefined MLN formula structure which consisted of 13 implications of the form e. g. $hasPet(X) \Rightarrow isAllergic(X)$. In another experiment, Alchemy's structure learning algorithm was applied to learn an MLN (formulas and weights) from scratch. The evaluation of the learned MLNs was carried out by using several of Alchemy's (approximate) inference algorithms. Additionally, the software PyMLNs (which is part of the ProbCog suite [33]) was used to perform exact inference on some MLNs in order to evaluate the deviation compared to the approximate results. The experiments showed that the results of the various Alchemy algorithms were quite similar and that there were no significant differences compared to the exact results.

Overall, the quality of the learned MLNs in terms of classification accuracy turned out to be not as good as expected. For various experiment settings, the MLNs resulting from structure as well as from parameter learning provide an accuracy of about 61% in predicting a child to be allergic. This could be improved by focusing on formulas the probabilities of which were significantly different from 0.5. However, further investigations into the evaluation of the quality of learned MLNs for prediction tasks in this domain will be necessary.

7 Conclusions and Future Work

This paper gives a brief overview on the state of the art in probabilistic reasoning, and illustrates the relevance of probabilistic methods for expert systems by describing their applications in various scenarios. The main advantage of probabilistic formalisms is a semantically clear and expressive handling of uncertainty which pervades all real world problems. Degrees of uncertainty can be conveniently obtained from statistical data and processed via probabilistic networks. Moreover, we go into more details on novel approaches combining probability theory and first-order logic which provide more expressive frameworks for

probabilistic reasoning. Using observations from theoretical foundations, implemented support systems, and a range of applications, we argue that probabilistic frameworks provide suitable and rich environments for learning, modelling, and reasoning in expert systems.

In particular, in this paper we discussed the problem of comparing formalisms for relational probabilistic knowledge representation from both a conceptual and a pragmatical point of view. This discussion was led by several focal aspects. First, we presented the framework of relational probabilistic conditional logic (RPCL) with grounding, averaging, and aggregating maximum entropy semantics, providing novel approaches to relational probabilistic knowledge representation and reasoning. This framework extends the information-theoretic principle of maximum entropy, that elegantly addresses the problem of incomplete knowledge, from propositional probabilistic conditional logic to the relational case. Secondly, we proposed a series of comparison and evaluation criteria for relational probabilistic models. These criteria describe, in an abstract fashion, diverse properties by which approaches to relational probabilistic reasoning can be distinguished. The criteria are focused on representation and reasoning issues, and we discussed them in an exemplary manner on BLPs, MLNs, and the three maximum entropy approaches $\mathrm{ME}_{\mathcal{G}_U}$, $\mathrm{ME}_{\varnothing}$, and $\mathrm{ME}_{\odot}$. Furthermore, we gave an overview of the KReator system which is a versatile toolbox for probabilistic relational reasoning. KReator alleviates the researcher's and knowledge engineer's work with different approaches to statistical relational learning by providing a unified and simple interface. Finally, using examples from the medical and biomedical domains, we illustrated the use of probabilistic knowledge representation and in particular of maximum entropy methods in application scenarios.

We expect that the discussion on comparing different approaches to relational probabilistic reasoning motivates further research and leads to application of the evaluation criteria to other formalisms. While the comparison and evaluation criteria formulated in this paper focus on the knowledge representation point of view, there are other important aspects. We already mentioned the area of learning which we deliberately left out in this paper. When knowledge bases grow larger, there should be the possibility to modularize them. Inference and learning on modularized knowledge bases should be able to reflect and exploit the modular structure. Moreover, different sources of knowledge as well as the integration of background knowledge could be supported; again, both inference and learning must take this into account. As part of our future work, we will elaborate detailed comparison and evaluation criteria for these additional aspects.

The KReator system already supports many scientific tasks in the area of relational probabilistic reasoning. Due to the open architecture of KReator and the ability to perform many tasks on abstract notions of e. g. knowledge bases, the task of implementing learning algorithms—which are already available for BLPs and MLNs—for different representation formalisms benefits from many commonalities of these algorithms. Most approaches on learning statistical relational models from data rely on established work from propositional learners.

Learning the structure of relational probabilistic models can be done using standard inductive logic programming systems like CLAUDIEN [57] or MACCENT [14]. Learning the values (probabilities) of the models can be performed using e. g. the EM-algorithm (expectation maximization, see [15]). These common components simplify implementing the ability to learn different knowledge bases from data within KReator. In order to gain the ability to learn RPCL knowledge bases (which differ significantly from other relational models which mostly rely on graphical notions and probabilistic dependence/independence assumptions) we plan to integrate an extended version of the CondorCKD system [25, 39]. CondorCKD is a propositional learning system for conditionals that relies on an algebraic characterization of interrelationships between conditionals in a knowledge base, cf. [36]. Our future work also comprises using KReator as a testbed to evaluate other approaches to relational probabilistic reasoning under maximum entropy [41].

KReator is available under the GNU General Public License and can be obtained from `http://kreator.cs.tu-dortmund.de/`.

References

1. C. Baral, M. Gelfond, and N. Rushton. Probabilistic Reasoning with Answer Sets. *Theory and Practice of Logic Programming*, 9:57–144, 2009.
2. J. Baumbach, A. Bunkowski, S. Lange, T. Oberwahrenbrock, N. Kleinbölting, S. Rahmen, and J. I. Baumbach. IMS^2 – An integrated medical software system for early lung cancer detection using ion mobility spectometry data of human breath. *J. of Integrative Bioinformatics*, 4(3), 2007.
3. J. I. Baumbach and M. Westhoff. Ion mobility spectometry to detect lung cancer and airway infections. *Spectroscopy Europe*, 18(6):22–27, 2006.
4. C. Beierle, M. Finthammer, G. Kern-Isberner, and M. Thimm. Evaluation and comparison criteria for approaches to probabilistic relational knowledge representation. In J. Bach and S. Edelkamp, editors, *KI 2011*, volume 7006 of *LNCS*, pages 63–74. Springer, 2011.
5. C. Beierle and G. Kern-Isberner. The relationship of the logic of big-stepped probabilities to standard probabilistic logics. In S. Link and H. Prade, editors, *Foundations of Information and Knowledge Systems (FoIKS 2010)*, LNCS, Vol. 5956, pages 191–210. Springer, 2010.
6. B. Bödeker, W. Vautz, and J. I. Baumbach. Peak finding and referencing in MCC/IMS-data. *International Journal for Ion Mobility Spectrometry*, 11(1-4):83–87, 2008.
7. J. S. Breese. Construction of Belief and Decision Networks. *Computational Intelligence*, 8(4):624–647, 1992.
8. M. Broecheler, G. I. Simari, and V. S. Subrahmanian. Using histograms to better answer queries to probabilistic logic programs. In *Logic Programming, 25th International Conference, ICLP 2009. Proceedings*, volume 5649 of *LNCS*, pages 40–54. Springer, 2009.
9. M. Bruynooghe, B. De Cat, J. Drijkoningen, D. Fierens, J. Goos, B. Gutmann, A. Kimmig, W. Labeeuw, S. Langenaken, N. Landwehr, W. Meert, E. Nuyts, R. Pellegrims, R. Rymenants, S. Segers, I. Thon, J. Van Eyck, G. Van den Broeck, T. Vangansewinkel, L. Van Hove, J. Vennekens, T. Weytjens, and L. De Raedt.

An Exercise with Statistical Relational Learning Systems. In P. Domingos and K. Kersting, editors, *International Workshop on Statistical Relational Learning (SRL-2009)*, Leuven, Belgium, 2009.
10. M. Chavira, A. Darwiche, and M. Jaeger. Compiling relational Bayesian networks for exact inference. *International Journal of Approximate Reasoning*, 42(1–2):4–20, May 2006.
11. J. Cussens. Logic-based formalisms for statistical relational learning. In L. Getoor and B. Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press, Cambridge, MA, 2007.
12. J. N. Darroch and D. Ratcliff. Generalized iterative scaling for log-linear models. In *Annals of Mathematical Statistics*, volume 43, pages 1470–1480. Institute of Mathematical Statistics, 1972.
13. L. De Raedt and K. Kersting. Probabilistic inductive logic programming. In L. D. Raedt, K. Kersting, N. Landwehr, S. Muggleton, and J. Chen, editors, *Probabilistic Inductive Logic Programming*, volume 4911 of *Lecture Notes in Computer Science*, pages 1–27. Springer, 2008.
14. L. Dehaspe. Maximum Entropy Modeling with Clausal Constraints. In *Proceedings of the 7th International Workshop on Inductive Logic Programming*, volume 1297 of *Lecture Notes in Artificial Intelligence*, pages 109–125. Springer, 1997.
15. A. P. Dempster, L. N. M., and D. B. Rubin. Maximum-likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.
16. P. Domingos and D. Lowd. *Markov Logic: An Interface Layer for Artificial Intelligence*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan and Claypool, San Rafael, CA, 2009.
17. R. Fagin and J. Y. Halpern. Reasoning about knowledge and probability. *J. ACM*, 41(2):340–367, 1994.
18. D. Fierens. *Learning Directed Probabilistic Logical Models from Relational Data*. PhD thesis, Katholieke Universiteit Leuven, 2008.
19. M. Finthammer. An iterative scaling algorithm for maximum entropy reasoning in relational probabilistic conditional logic. In *Scalable Uncertainty Management, 6th International Conference, Proceedings*, LNAI. Springer, 2012. (to appear).
20. M. Finthammer, C. Beierle, B. Berger, and G. Kern-Isberner. Probabilistic reasoning at optimum entropy with the MEcore system. In H. C. Lane and H. W. Guesgen, editors, *Proceedings 22nd International FLAIRS Conference, FLAIRS'09*. AAAI Press, Menlo Park, California, 2009.
21. M. Finthammer, C. Beierle, J. Fisseler, G. Kern-Isberner, and J. I. Baumbach. Using probabilistic relational learning to support bronchial carcinoma diagnosis based on ion mobility spectrometry. *International Journal for Ion Mobility Spectrometry*, 13:83–93, 2010.
22. M. Finthammer and M. Thimm. An Integrated Development Environment for Probabilistic Relational Reasoning. *International Journal of the IGPL*, 2011. . (to appear).
23. J. Fisseler. Toward Markov Logic with Conditional Probabilities. In D. C. Wilson and H. C. Lane, editors, *Proceedings of the 21st International FLAIRS Conference (FLAIRS'08)*, pages 643–648. AAAI Press, 2008.
24. J. Fisseler. *Learning and Modeling with Probabilistic Conditional Logic*, volume 328 of *Dissertations in Artificial Intelligence*. IOS Press, Amsterdam, 2010.
25. J. Fisseler, G. Kern-Isberner, C. Beierle, A. Koch, and C. Müller. Algebraic Knowledge Discovery using Haskell. In *Practical Aspects of Declarative Languages, 9th In-*

ternational Symposium, volume 4354 of *Lecture Notes in Computer Science*, pages 80–93. Springer, 2007.
26. E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1994.
27. L. Getoor, N. Friedman, D. Koller, and B. Tasker. Learning Probabilistic Models of Relational Structure. In C. E. Brodley and A. P. Danyluk, editors, *Proceedings of the 18th International Conference on Machine Learning (ICML 2001)*, pages 170–177. Morgan Kaufmann, 2001.
28. L. Getoor and J. Grant. PRL: A probabilistic relational language. *Machine Learning*, 62(1):7–31, 2006.
29. L. Getoor and B. Taskar, editors. *Introduction to Statistical Relational Learning*. MIT Press, 2007.
30. J. Gosling, B. Joy, G. Steele, and G. Bracha. *The Java Language Specification*. Addison-Wesley, third edition edition, 2005.
31. M. Jaeger. Relational Bayesian Networks: A Survey. *Electronic Transactions in Artificial Intelligence*, 6, 2002.
32. M. Jaeger. Model-Theoretic Expressivity Analysis. In L. D. Raedt, K. Kersting, N. Landwehr, S. Muggleton, and J. Chen, editors, *Probabilistic Inductive Logic Programming*, volume 4911 of *Lecture Notes in Computer Science*, pages 325–339. Springer, 2008.
33. D. Jain, L. Mösenlechner, and M. Beetz. Equipping Robot Control Programs with First-Order Probabilistic Reasoning Capabilities. In *International Conference on Robotics and Automation (ICRA)*, pages 3130–3135, 2009.
34. F. V. Jensen and T. D. Nielsen. *Bayesian Networks and Decision Graphs*. Springer-Verlag, 2007.
35. G. Kern-Isberner. Characterizing the principle of minimum cross-entropy within a conditional-logical framework. *Artificial Intelligence*, 98:169–208, 1998.
36. G. Kern-Isberner. *Conditionals in nonmonotonic reasoning and belief revision*. Number 2087 in Lecture Notes in Computer Science. Springer, 2001.
37. G. Kern-Isberner. Linking iterated belief change operations to nonmonotonic reasoning. In G. Brewka and J. Lang, editors, *Proceedings 11th International Conference on Knowledge Representation and Reasoning, KR'2008*, pages 166–176, Menlo Park, CA, 2008. AAAI Press.
38. G. Kern-Isberner, C. Beierle, M. Finthammer, and M. Thimm. Probabilistic logics in expert systems: Approaches, implementations, and applications. In *Proceedings of the 22nd International Conference on Database and Expert Systems Applications (DEXA'11)*, volume 6860 of *LNCS*, pages 27–46. Springer, 2011.
39. G. Kern-Isberner and J. Fisseler. Knowledge Discovery by Reversing Inductive Knowledge Representation. In *Proceedings of the Ninth International Conference on the Principles of Knowledge Representation and Reasoning, KR-2004*, pages 34–44. AAAI Press, 2004.
40. G. Kern-Isberner and T. Lukasiewicz. Combining probabilistic logic programming with the power of maximum entropy. *Artificial Intelligence, Special Issue on Nonmonotonic Reasoning*, 157(1-2):139–202, 2004.
41. G. Kern-Isberner and M. Thimm. Novel Semantical Approaches to Relational Probabilistic Conditionals. In *Proceedings of the Twelfth International Conference on the Principles of Knowledge Representation and Reasoning (KR'10)*, pages 382–392, May 2010.
42. G. Kern-Isberner and M. Thimm. A ranking semantics for first-order conditionals. In *Proceedings 20th European Conference on Artificial Intelligence, ECAI-2012*, 2012. (to appear).

43. K. Kersting and L. De Raedt. Bayesian Logic Programming: Theory and Tool. In L. Getoor and B. Taskar, editors, *An Introduction to Statistical Relational Learning*. MIT Press, 2007.
44. N. S. Ketkar, L. B. Holder, and D. J. Cook. Comparison of Graph-based and Logic-based Multi-relational Data Mining. *SIGKDD Explor. Newsl.*, 7(2):64–71, 2005.
45. S. Kok, P. Singla, M. Richardson, P. Domingos, M. Sumner, H. Poon, D. Lowd, and J. Wang. *The Alchemy System for Statistical Relational AI: User Manual*. Department of Computer Science and Engineering, University of Washington, 2008.
46. A. Krämer and C. Beierle. On lifted inference for a relational probabilistic conditional logic with maximum entropy semantics. In T. Lukasiewicz and A. Sali, editors, *Foundations of Information and Knowledge Systems (FoIKS 2012)*, volume 7153 of *LNCS*, pages 224–243. Springer, 2012.
47. S. Loh, M. Thimm, and G. Kern-Isberner. On the problem of grounding a relational probabilistic conditional knowledge base. In *Proceedings of the 14th International Workshop on Non-Monotonic Reasoning (NMR'10)*, Toronto, Canada, May 2010.
48. S. Loh, M. Thimm, and G. Kern-Isberner. On the problem of grounding a relational probabilistic conditional knowledge base. In T. Meyer and E. Ternovska, editors, *Proceedings 13th International Workshop on Nonmonotonic Reasoning NMR'2010, Subworkshop on NMR and Uncertainty*, 2010.
49. S. Muggleton and J. Chen. A Behavioral Comparison of some Probabilistic Logic Models. In L. D. Raedt, K. Kersting, N. Landwehr, S. Muggleton, and J. Chen, editors, *Probabilistic Inductive Logic Programming*, pages 305–324. Springer, 2008.
50. S. H. Muggleton. Stochastic Logic Programs. In L. de Raedt, editor, *Advances in Inductive Logic Programming*, pages 254–264. IOS Press, Amsterdam, Netherlands, 1996.
51. N. Nilsson. Probabilistic logic. *Artificial Intelligence*, 28:71–87, 1986.
52. D. Nute and C. Cross. Conditional Logic. In D. Gabbay and F. Guenther, editors, *Handbook of Philosophical Logic*, volume 4, pages 1–98. Kluwer Academic Publishers, 2002.
53. J. Paris. *The uncertain reasoner's companion – A mathematical perspective*. Cambridge University Press, 1994.
54. J. Paris. *The uncertain reasoner's companion – A mathematical perspective*. Cambridge University Press, 1994.
55. J. Pearl. Fusion, propagation and structuring in belief networks. *Artificial Intelligence*, 29:241–288, 1986.
56. J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.
57. L. D. Raedt and L. Dehaspe. Clausal Discovery. *Machine Learning*, 26:99–146, 1997.
58. L. D. Raedt, A. Kimmig, B. Gutmann, K. Kersting, V. S. Costa, and H. Toivonen. Probabilistic Inductive Querying Using ProbLog. Technical Report CW 552, Department of Computer Science, Katholieke Universiteit Leuven, Belgium, June 2009.
59. M. Richardson and P. Domingos. Markov Logic Networks. *Machine Learning*, 62(1–2):107–136, 2006.
60. Robert Koch-Institut. *Public Use File KiGGS, Kinder- und Jugendgesundheitssurvey 2003-2006*, Berlin, 2008.
61. W. Rödder. Conditional Logic and the Principle of Entropy. *Artificial Intelligence*, 117:83–106, 2000.

62. W. Rödder and C.-H. Meyer. Coherent Knowledge Processing at Maximum Entropy by SPIRIT. In *Proceedings UAI 1996*, pages 470–476, 1996.
63. W. Rödder, E. Reucher, and F. Kulmann. Features of the expert-system-shell SPIRIT. *Logic Journal of the IGPL*, 14(3):483–500, 2006.
64. E. Schmaußer-Hechfellner. Probabilistic logic knowledge modelling of statistical medical data by applying learning- and inference-techniques of Markov logic networks. Bachelor Thesis, Dept. of Computer Science, FernUniversität in Hagen, 2011. (in German).
65. A. Srinivasan. The Aleph Manual. `www.comlab.ox.ac.uk/activities/machinelearning/Aleph/`, 2007.
66. M. Thimm, M. Finthammer, S. Loh, G. Kern-Isberner, and C. Beierle. A system for relational probabilistic reasoning on maximum entropy. In H. W. Guesgen and R. C. Murray, editors, *Proceedings 23rd International FLAIRS Conference, FLAIRS'10*, pages 116–121, Menlo Park, California, 2010. AAAI Press.
67. M. Thimm, G. Kern-Isberner, and J. Fisseler. Relational probabilistic conditional reasoning at maximum entropy. In *ECSQARU*, volume 6717 of *LNCS*, pages 447–458. Springer, 2011.
68. M. P. Wellman, J. S. Breese, and R. P. Goldman. From Knowledge Bases to Decision Models. *The Knowledge Engineering Review*, 7(1):35–53, 1992.
69. A. Yue, W. Liu, and A. Hunter. Measuring the ignorance and degree of satisfaction for answering queries in imprecise probabilistic logic programs. In *Scalable Uncertainty Management, Second International Conference, Proceedings*, volume 5291 of *LNCS*, pages 386–400. Springer, 2008.