Statistical Relational AI: Logic, Probability and Computation

Kristian Kersting, Sriraam Natarajan, David Poole
kristian.kersting@iais.fraunhofer.de,
snataraj@wfubmc.edu,
poole@cs.ubc.ca

August 31, 2011

Abstract

One of the key challenges in building intelligent agents is closing the gap between logical and statistical AI, so that we can have rich representations including objects, relations and uncertainty, that we can effectively learn and carry out inference with. Over the last 25 years there has been a considerable body of research into combinations of predicate logic and probability forming what has become known as statistical relational artificial intelligence (StaR-AI). We overview the foundations of the area, give some research problems, proposed solutions, outstanding issues, and clear up some misconceptions that have arisen. We discuss representations, semantics, inference, learning and applications, and provide references to the literature.

1 Introduction

Over the last 25 years there has been a considerable body of research into combining first-order logic and probability, evolving into what has come to be called *statistical relational AI*, which can be defined as:

> *the study and design of intelligent agents with imperfect sensors that act in noisy worlds composed of objects, where there can be complex relations among the objects.*

This paper aims at reviewing the foundations of the area, motivating the issues, justifying some choices that have been made and giving some open problems. Laying bare the foundations will hopefully inspire others to join us in exploring the frontiers and the yet unexplored areas.

One of the barriers to understanding this area is that it builds from multiple traditions, which often use the same vocabulary to mean different things. Common terms such as "variable", "domain", "relation", and "parameter" have

Example	*Author*	*Thread*	*Length*	*WhereRead*	*UserAction*
e_1	*known*	*new*	*long*	*home*	*skips*
e_2	*unknown*	*new*	*short*	*work*	*reads*
e_3	*unknown*	*follow_up*	*long*	*work*	*skips*
e_4	*known*	*follow_up*	*long*	*home*	*skips*
...	...	...	...	...	...

(a)

Individual	Property	Value
joe	*likes*	*resort_14*
joe	*dislikes*	*resort_35*
...	...	...
resort_14	*type*	*resort*
resort_14	*near*	*beach_18*
beach_18	*type*	*beach*
beach_18	*covered_in*	*ws*
ws	*type*	*sand*
ws	*color*	*white*
...	...	...

(b)

Figure 1: Two examples for datasets one can take advantage of to capture characteristics of interest of their unknown underlying probability distribution.

come to have accepted meanings in mathematics, computing, logic and probability, but their meanings in each of these areas is different enough to cause confusion.

Both predicate logic (e.g., the first-order predicate calculus) and Bayesian probability can be seen as extending the propositional logic, one by adding relations, individuals and quantified variables, the other by allowing for measures over possible worlds and conditional queries. Relational probabilistic models[1], which form the basis of statistical relational AI, can be seen as combinations of probability and predicate calculus to allow for individuals and relations as well as probabilities.

To understand the needs for such a combination, consider learning from the two datasets in Figure 1 (taken from [106]). Dataset (a) is the sort used in traditional supervised and unsupervised machine learning and data mining. Standard textbook supervised learning algorithms can learn a decision tree, a neural network, or a support vector machine to predict *UserAction*. A belief network learning algorithm can be used to learn a representation of the distribution over all of the features. Dataset (b), from which we may want to predict what Joe likes, is different. Many of the values in the table are meaningless

[1]Here we use this term in the broad sense, meaning any models that combine relations and probabilities.

names that can't be used directly in supervised learning. Instead, it is the relationship among the individuals in the world that provides the generalizations from which to learn. For example, we may want to learn that Joe likes resorts that are near sandy beaches. Learning from such datasets has been studied under the umbrella of inductive logic programming [84, 74, 18] mainly because logic programs provide a good representation for the generalizations required to make predictions. Specifically, inductive logic programming (ILP) is a research field at the intersection of machine learning and logic programming. It forms a formal framework for relational learning and has introduced practical algorithms for inductively learning relational descriptions (in the form of logic programs) from examples and background knowledge. ILP is one of the foundations of StaR-AI, as it provides a toolbox of techniques for structure learning in relational domains.

One confusion about the area stems from the term "relational"; after all most existing datasets are, or can be, stored in relational databases. The techniques of relational probabilistic models are applicable to cases where the values in the database are names of individuals, and it is the properties of the individuals and the relationship between the individuals that are modeled. It is sometimes also called multi-relational learning, as it is the interrelations that are important. This is a misnomer because, as can be seen in Figure 1 (b), it is not multiple relations that cause problems (and provide opportunities to exploit structure), as a single triple relation can store any relational database (in a so-called triple-store).

The term statistical relational AI comes from not only having probabilities and relations, but that the models are derived from data and prior knowledge.

2 Motivation

Artificial intelligence (AI) is the study of computational agents that act intelligently [106, 115]. The basic argument for probability as a foundation of AI is that agents that act under uncertainty are gambling, and probability is the calculus of gambling in that agents who don't use probability will lose to those that do use it [130]. While there are a number of interpretations of probability, the most suitable is a Bayesian or subjective view of probability: our agents do not encounter generic events, but have to make decisions in particular circumstances, and only have access to their percepts and their beliefs.

In probability theory, possible worlds are described in terms of so-called *random variables* (although they are neither random nor variable). A random variable has a value in every world. We can either define random variables in terms of worlds or define worlds in terms of random variables. A random variable having a particular value is a proposition. Probability is defined in terms of a non-negative measure over sets of possible worlds that follow some very intuitive axioms.

In Bayesian probability, we make explicit assumptions and the conclusions are logical consequences of the specified knowledge and assumptions. One par-

ticular explicit assumption is the assumption of conditional independence. A Bayesian network [93] is an acyclic directed graphical model of probabilistic dependence where the nodes are random variables. A Bayesian network encapsulates the independence: a variable is conditionally independent of other variables that are not its descendants in the graph given its parents in the graph. This has turned out to be a very useful assumption in practice. Undirected graphical models [93] encapsulate the assumption that a variable is independent of other variables given its neighbours.

These motivations for probability (and similar motivations for utility) do not depend on non-relational representations. We also want to be able to reason about individuals[2] and relationships among individuals. In statistical relational AI, we want to condition on properties of individuals and relations among individuals and make probabilistic predictions about properties and relationships. We often want to build the models before we know which individuals exist in a domain, so that the models can be applied to diverse populations.

The key property that is exploited is that of exchangeability: those individuals about which we have the same information should be treated identically. Formally, this means we can exchange the names, and still get the same probabilities. This implies that before we know anything particular about any of the individuals, they all should share their probabilistic parameters. Results from Statistics, particularly the celebrated De Finetti's theorem [17, 54], motivate the forms of possible models (if we allow the population to be unbounded).

3 Representation

Statistical relational models are typically defined in terms of parameterized random variables [101] which are often drawn in terms of plates [7]. A parameterized random variable corresponds to a predicate (atom) or a function symbol in logic. It can include logical variables (which form the parameters). In the following examples, we will write logical variables (which denote individuals) in upper case, and constants, function and predicate symbols in lower case. We assume that the logical variables are typed, where the domain of the type, the set of individuals of the type, is called the population.

Parameterized random variables are best described in terms of an example. Consider the case of diagnosing students' performance in adding multi-digit numbers of the form

$$\begin{array}{cccc} & & x_1 & x_0 \\ + & & y_1 & y_0 \\ \hline & z_2 & z_1 & z_0 \end{array}$$

[2]Individuals are things. They are also called "objects", but that terminology is often confusing to people who have been brought up with object-oriented programming, where objects are data structures and associated methods. For example, a person individual is a real person and not a data structure that encapsulates information about a person. We can be uncertain about the properties of a person, but a computer is not uncertain about its data structures.

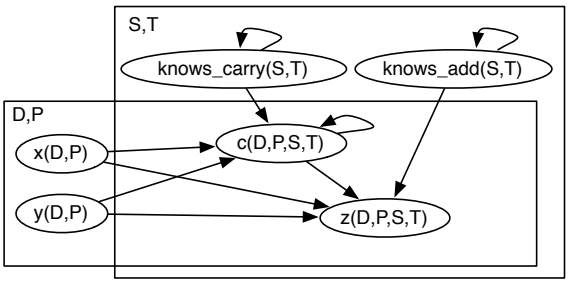

Figure 2: Belief network with plates for multidigit addition

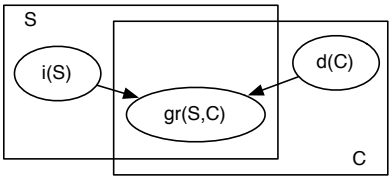

Figure 3: Plate representation of the grades model

A student, given values for the x's and the y's, provides values for the z's. The aim is to determine whether the students have mastered addition from observations of their performance.

Whether a student gets the correct answer for z_i depends on x_i, y_i, the value carried in and whether she knows addition. Whether a student gets the correct carry depends on the previous x, y and carry, and whether she knowns how to carry. This dependency can be seen in Figure 2. Here $x(D, P)$ is a parameterized random variable. There is a random variable for each digit D and each problem P. A ground instance, such as $x(d_3, problem_{57})$, is a random variable that may represent the third digit of problem 57. Similarly, there is a z-variable for each digit D, problem P, student S, and time T. The plate notation can be read as duplicating the random variable for each tuple of individual the plate is parameterized by.

As another example, Figure 3 gives a plate representation of a model to predict the grades of students in courses. In this figure, S is a logical variable that denotes a student and C is a logical variable that denotes a course. Parameterized random variable $gr(S, C)$ denotes the grade of S in course C, parameterized random variable $i(S)$ denotes the intelligence of student S, and $d(C)$ represents the difficulty of course C. Note that this figure redundantly includes the logical variables in the plates as well as arguments to the parameterized random variables.

Such parametrized models represent their grounding, where there is an instance of each random variable for each assignment of an individual to a logical

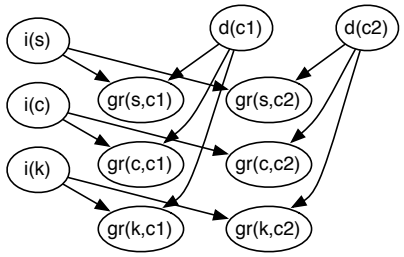

Figure 4: Grounding of the grades model for 3 people and 2 courses

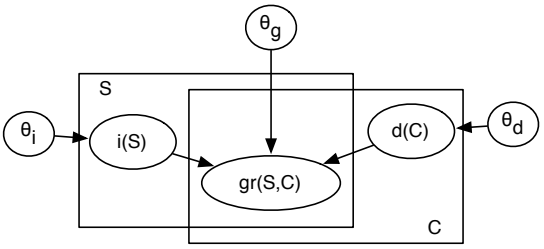

Figure 5: Plate representation of the grades model, with shared parameters explicit

variable. Figure 4 shows such a grounding where there are three students Sam (s), Chris (c) and Kim (k) and two courses ($c1$ and $c2$).

For the plate model of Figure 3 with n students and m courses, there are nm instance of the *grade* random variable, n instances of the *intelligence* random variable and m instance of the *difficulty* random variable. Thus there are $nm + n + m$ random variables in the grounding.

Note the independence assumptions in this example: The intelligence of the students are independent of each other give no observations. The difficulty of the courses are independent of each other, given no observations. The grades are independent given the intelligence and the difficulty. Given no observations, the grades are dependent on each other. Given observations on grades, the intelligence variables and the difficulty variables can be completely interdependent.

While we have given directed models here, the same principles also apply to undirected models. The basic principle used by all methods is that of *parameter sharing*: the instances of the parameterized random variables created by substituting constants for logical variables share the same probabilistic parameters. The various languages differ in how to specify the conditional probabilities of the parametrized random variables given their parents, or the other parameters of the probabilistic model. Often in plate models [7, 52], the numerical parameters are made explicit, to emphasize that the parameters do not depend on the individuals.

Figure 5 shows the plate model of Figure 3 with the numerical parameters pulled out. θ_i specifies the shared prior probability of $i(S)$, θ_d specifies the prior probability of $d(C)$, and θ_g specifies the numerical parameters of the conditional probability $P(gr(S, C)|i(S), d(C))$. The figure makes explicit that these numerical parameters do not depend on the individuals.

The first relational probabilistic languages either had explicit languages for constructing the ground network [6, 41] or defined templates for prior and conditional probabilities [49] directly in term of tables, and required a combination function (such as noisy-and or noisy-or) when the number of parents depends on the number of individuals. Tables with combination functions turn out to be not a very flexible representation as they cannot represent the subtleties involved in how one random variable can depend on others.

For example, in the addition example of Figure 2, the carry for digit D, namely $c(D, P, S, T)$ depends, in part, on $c(D - 1, P, S, T)$, that is, on the carry from the previous digit. There is a special case for the first digit.

Another complex example is to determine the probability that two authors are collaborators, which may depend on whether they have written papers in common, or even whether they have written papers apart from each other. That is $collaborates(A_1, A_2)$ may depend on whether there is some paper P such that $author(A_1, P)$ and $author(A_2, P)$, and a representation has to be able to specify what happens when there is more than one paper that they are co-authors of. It may also depend on other co-authors who may have collaborated with each of them. Any such rules are only applicable if $A_1 \neq A_2$, that is if A_1 and A_2 are different people. Examples such as these require more sophisticated languages than the plate models specified above.

To represent such examples, it is useful to be able to specify how the logical variables interact, as is done in logic programs. The independent choice logic (ICL) [99, 103] (originally called probabilistic Horn abduction [96, 98]) allows for arbitrary (acyclic) logic programs (including negation as failure) to be used to represent the dependency. The conditional probability parameters are represented as independent probabilistic inputs to the logic program. A logic program that represents the above example is given in Chapter 14 of [106]. This idea also forms the foundation for Prism [119, 120], which has concentrated on learning, and for Problog [21], a project to build an efficient and flexible probabilistic logic programming language.

Other languages are based on entity-relationship models [34, 47], fragments of first-order logic [83, 58, 73], or even in terms of programming languages such as in IBAL [95] or Church [42]. Undirected models, exemplified by Markov logic networks (MLNs) [113], have a similar notion of parameterized random variables, but the probabilities are represented as weights of first-order clauses. These models also mean their grounding, but for MLNs the grounding is a Markov network rather than a Bayesian network (see Pearl [93] for a discussion of the relationship between Bayesian and Markov networks).

All approaches mentioned so far assume finitely many parameters to specify the prior probabilistic model. So-called nonparametric Bayesian approaches such as NP-BLOG [8], infinite (hidden) relational models [56, 141], and rela-

tional Gaussian processes [14, 144, 143, 127] allow for infinitely many parameters, for example because there could be an unbounded number of classes of individuals (or parametrized random variables with an unbounded range). As computers can only handle finite representations, these models require some process to specify how these parameters are generated.

4 Representational Issues

Many probabilistic relational representation languages already exists; see e.g. [19, 38, 22] for overviews. Rather than listing and describing the properties of the various proposal, we here descrive some issues that arise in designing representation language.

4.1 Desiderata of a Representation Language

Often a good representation is a compromise between many competing objectives. Some of the objectives of a representation language, include:

prediction accuracy: is it able to generalize to predict unseen examples? Thus the representation should be able to represent the generalized knowledge of the domain.

compactness: is the representation of the knowledge succinct? This is related to prediction accuracy, as it is conjectured that a compact representation will be best able to generalize. It is also related to introducing latent random variables, as the reason that we introduce latent variables is to make a model simpler.

expressivity: is it rich enough to represent the knowledge required to solve the problems required of it? Can it deal with discrete and continuous random variables? Can it deal with domains evolving over time?

efficient inference: can it to solve all/average/some problems efficiently? Can it efficiently deal with latent variables?

understandability: can someone understand what has been learned?

modularity : can parts of the representation be understood or learned separately? This has two aspects:

- the ability to conjoin models that are developed independently. If different teams work on individual parts of a model, can these parts easily be combined? This should be able to be done even if the teams do not know their work will be combined and even if the teams have developed both directed and undirected models.
- the ability to decompose a model into smaller parts for specific applications. If there is a big model, can someone take a subset of the

model and use it? Can someone take advantage of a hierarchical decomposition into weakly interacting parts?

ability to incorporate prior knowledge: if there is domain knowledge that has been acquired in a research community, can it be incorporated into the model?

interoperability with data: can it learn from multiple heterogeneous data sets? Heterogeneous can mean having different overlapping concepts, derived from different contexts, at different levels of abstraction (in terms of generality and specificity) or different levels of detail (in terms of parts and subparts). It should also be able to incorporate causal data that has arisen from interventions and observational data that only involved passively observing the world.

latent factors: can it learn the relationship between random variables if the relationship is a function of observed and unobserved (latent) characteristics, potentially in addition to contextual factors? Can it even introduce latent factors on the fly?

The promise of statistical relational AI is to automate decisions in complex domains. For complex domains, there are many diverse relevant pieces of information that cannot be ignored. For a system to be relied on for critical life-or-death decisions or ones that rely on large economic risks, the system needs to be able to explain or at least justify its recommendations to the people who are legally responsible, and these recommendations need to be based on the best available evidence (and preferably on all available evidence that is possibly relevant). Of course, not all of the existing representations have attempted to achieve all of these desiderata.

Sections 4.2-4.6 present some issues that have arisen in the various representations. Whether these turn out to be unimportant or whether one choice will prevail is still an open question.

4.2 Directed versus undirected models

In a manner similar to the difference between Bayesian networks and Markov networks [93], there are directed relational models (e.g., [99, 103, 47, 73]) and undirected models (e.g., [131, 113, 44, 138]). The main differences between these are:

- The probabilities of the directed models can be directly interpreted as conditional probabilities. This means that they can be explained to users, and can be locally learned if all of the relevant properties are observed.
- Because the probabilities can be interpreted locally, the directed representations are modular. The probabilities can be acquired modularly, and need not change if the model is expanded or combined with another model.

- For inference in directed models, typically only a very small part of the grounded network needs to be used for inference. In particular, only the ancestors of the observed and query relations need to be considered. This has been characterized as abduction [97], where the relevant probabilities can be extracted from proofs (in the logical sense) of the observations and queries.

- The directed models require the structure of the conditional probabilities to be acyclic. (A specification of $P(A|B)$ and $P(B|A)$ is typically inconsistent, and sometimes does not specify a unique distribution.) For relational models, this is problematic if we have conditional probabilities of the form:

  $$P(foo(X)|foo(Y)\dots)$$

  To make this acyclic, we could totally order the individuals, and enforce a "less than" operation. However, this means that the individuals are no longer exchangeable. One example where this arises is if we wanted to specify the probability that friendship is transitive:

  $$P(friends(X, Y)|\exists Z\ friends(X, Z), friends(Z, Y))$$

  Naive ways to incorporate such rules do not work as expected.

Undirected models, such as Markov logic networks [113], do not have a problem with acyclicity, but the weights of the undirected models cannot directly interpreted as probabilities, and it is often very difficult to interpret the numbers at all. However, as many of the applications where we may want to predict relations have a relation on some individual depending on relations on other individuals, the undirected models tend to be more effective for these domains.

It is an open research question to allow directed models to handle cyclic dependencies and to allow the undirected models to be modular and explainable.

4.3 Factors and Clauses

In graphical models, the next highest level of data structure is the factor or potential [125, 146, 33]. Factors represent functions from the domains of sets of variables into other values (typically reals). For example, a factor on random variables $\{A, B, C\}$ is a function from $dom(A) \times dom(B) \times dom(C) \to \Re$, where $dom(A)$ is the domain of random variable A, and $\Re$ is the set of real numbers. Factors can represent conditional probabilities, utilities, arbitrary potentials, and the intermediate results of computation. An alternative to using an explicit representation of full factors is to represent the value of one assignment to variables, or to any Boolean function of assignments to values, such as using clauses [15]; this enables local structure to be exploited including context-specific independence and zero parameters [9, 116].

In relational representations the corresponding representation is a parameterized factor or parfactor [101], which is a triple

$$\langle C, V, t\rangle$$

where C is a set of inequality constraints on logical variables, V is a set of parameterized random variables and t is a factor on V. Note that t will be the factor that is used for all assignments of individuals to logical variables in V. If the factor represents a clause, the table specifies just a single number, and then V is written as a first-order clause as for example done in MLNs.

4.4 Parametrizing Atoms

Poole [96, 98] first noticed that probabilities over atoms was adequate to represent any (conditional) probability distribution. By making these atoms assumable, the probability of any proposition could be computed through abduction: determining which of these assumable atoms were needed to imply the proposition. More recently, inference methods based on MLNs and model counting [40, 134], have used probabilities on atoms to allow probabilistic inference to be implemented by model counting. These proposed methods, although superficially similar, have very different properties when combined with languages that allow for probabilities on rules or clauses.

Suppose we want to have rules with probabilities:

$$a \leftarrow b : p_1$$
$$a \leftarrow b \wedge c : p_2$$

Both methods create atoms n_1 and n_2 with $P(n_1) = p_1$ and $P(n_2) = p_2$. Poole's method, which we will call the "independent choice" model, results in the rules:

$$a \leftarrow b \wedge n_1$$
$$a \leftarrow b \wedge c \wedge n_2$$

which is equivalent to:

$$a \leftarrow b \wedge (n_1 \vee c \wedge n_2)$$

This means its completion:

$$a \leftrightarrow (b \wedge n_1 \vee b \wedge c \wedge n_2)$$

The method of Gogate and Domingos [40], Van den Broeck et al. [134] (the MLN method) would result in:

$$(a \leftarrow b) \leftrightarrow n_1$$
$$(a \leftarrow b \wedge c) \leftrightarrow n_2$$

In the independent choice method, n_1 and n_2 can be independent.

In the MLN method, n_1 and n_2 cannot be independent as n_2 logically implies n_1. In the independent choice method p_1 and p_2 can be arbitrarily assigned, whereas in the MLN method, $P(n_2) \geq P(n_1)$. Moreover, if b has a low probability, then n_1 and n_2 both have a high probability as they represent the material

implication which is true whenever b is false. As such, in the MLN method, p_1 and p_2 are close to each other when b has a low probability. However, in the independent choice method, they represent conditional probabilities and can have arbitrary probabilities. Sato [119, 120] has done extensive work on learning for the independent choice case.

What seems to be a slight different in the reading of rules has a big effect on the semantics. The independent choice method allows a simple semantics in terms of independent atomic choices and a logic program that gives the consequences of the choices [98, 99]. The semantics of Gogate and Domingos [40], Van den Broeck et al. [134] does not assume independent atoms, but appeals to the maximum entropy assignment, which makes it much more difficult to interpret the numbers.

4.5 Aggregators vs Combining Rules

Suppose binary parametrized random variable a is connected to binary parametrized random variable $b(X)$ which contains an extra logical variable, X. In the grounding, a is connected to an unbounded number of instances of $b(X)$. A directed model where $b(X)$ is a child of a produces a naive Bayesian model in the grounding with a separate factor for each instance. An undirected model with a potential for a and a pairwise potential for each factor, gives the same model. In both of these models the joint probability is the product of factors. However, for a directed model with $b(X)$ as a parent of a, in the grounding a has a unbounded number of parents. There are two main approaches to deal with this "multiple-parent" problem: aggregators and combining rules. Common ways to aggregate [34, 90, 94] include logical operators such a noisy-or, noisy-and, or ways to combine the probabilities. This requirement for aggregation occurs in a directed model whenever a parent contains an extra logical variable.

For a positive model (with no zero probabilities) the aggregation model that corresponds to the naive Bayesian model is a logistic regression model. In particular, suppose that the joint probability is a product of factors:

$$P(a, b_1, \ldots, b_n) \propto \prod_i f(a, b_i) \times g(a)$$

where b_i is $b(k_i)$ for some enumeration of the population of size n.

Then, writing $a = \mathit{true}$ as a and $a = \mathit{false}$ as $\neg a$.

$$\begin{aligned} P(a|b_1, \ldots, b_n) &= \frac{P(a, b_1, \ldots, b_n)}{P(a, b_1, \ldots, b_n) + P(\neg a, b_1, \ldots, b_n)} \\ &= \frac{1}{1 + 1/\prod_i f(a, b_i)/f(\neg a, b_i) \times g(a)/g(\neg a)} \\ &= \frac{1}{1 + e^{-\sum_i \hat{f}(a,b_i) + \hat{g}(a)}} \end{aligned}$$

where $\hat{f}(a, b_i) = \log(f(a, b_i)/f(\neg a, b_i))$ and $\hat{g}(a) = \log g(a)/g(\neg a)$. We can then represent $\hat{f}(a, b_i)$ as wb_i and $\hat{g(a)}$ as w' (as a is fixed). In a standard graphical model, n is fixed and it does not matter how the values of b_i are represented. However, in a relational model where n can vary, the details of the representation affect how the model changes with n.

For example, suppose we have a logistic regression model that represents: a is true if and only if b is true for more than 5 individuals out of a population of 10. Now consider what this model predicts when the population size is 20. If the values of b are represented with $0 = \textit{false}$ and $1 = \textit{true}$, this model will have a true if b is true for more than 5 individuals out of a population of 20. However, if the values of b are represented with $-1 = \textit{false}$ and $1 = \textit{true}$, this model will have a true if b is true for more than 10 individuals out of a population of 20. If the values of b are represented with $1 = \textit{false}$ and $0 = \textit{true}$, this model will have a true if b is true for more than 15 individuals out of a population of 20. Different choices can result in arbitrary thresholds for the population.

While the dependence on population may be arbitrary when a single population is observed, it affects the ability of a model to predict when multiple populations are observed. For example, suppose we want to model whether someone is happy depends on the number of their friends that are mean to them. One model could be that people are happy as long as they have more than 5 friends who are not mean to them. Another model could be that people are happy if more than half of their friends are not mean to them. A third model could be that people are happy as long as fewer than 5 of their friends are mean to them. These three models coincide for people with 10 friends, but make different predictions for people with 20 friends. A particular logistic regression model (or a naive Bayesian model) is only able to model one of the dependencies of how predictions depend on population, and so cannot properly fit data that does not adhere to that dependence. One way to fit various dependencies is to always include the property 1 which has the value 1 for each individual (in the naive Bayesian model, this needs to be observed). Another, as typically used in Markov logic networks, is to have separate parameters for the positive and negative cases. Each of these can model linear dependence on population.

While aggregators assume that the parents of a random variable "collectively" influence the target, combining rules [50, 86, 58] take the opposite view i.e., they assume "independence of causal influence," (ICI) in that each parent of a random variable influences the target independently. This is to say that, combining rules capture the notion of causal independence [46] in directed models. In this view, multiple causes on a target variable can be decomposed into several independent causes whose effects are combined to yield a final value. Each parent or set of related parents produces a different value for the child variable, all of which are combined using a deterministic or stochastic function. Depending on how the causes are decomposed and how the effects are combined, we can express the conditional distribution of the target variable given all the causes as a function of the conditional distributions of the target variable given each independent cause using a combining rule. When analyzing the difference

between directed vs undirected models, it is important to understand how to represent causal independencies in undirected models. A first-step in this direction has been taken by Natarajan et al. [87] where an algorithm for converting a directed model with combining rules to an equivalent MLN has been presented.

For a particular class of combining rules called *decomposable combining rules*, it can be shown that there is an equivalent class of aggregators. Learning in the presence of combining rules have been considered previously [50, 86] where the EM-based algorithms use the value-based aggregation functions. At a fairly high level, decomposable combining rules are functions that yield the same target distribution irrespective of the order of the parents. The most common ones used in the literature are mean, weighted mean and noisy-or combining rules [86, 50, 58]. For these combining rules there are equivalent aggregators. For example, mean can be represented as a stochastic choice using an uniform distribution over the parent values, weighted mean corresponds to a stochastic choice given the weight distribution while noisy-or has a value-based interpretation [93] and a distribution based interpretation [86]. On the other hand, for an arbitrary combining rule, determining an equivalent aggregator is still an open question.

4.6 Probability-Specificity Tradeoff

It is often not clear whether highly complex but specific logical structures with extreme probabilities (close to 0 or 1) or simpler but less-specific logical structures with arbitrary distributions form better models. In many domains such as medical domains, using specific rules with probabilities close to 0 or 1 allows for easy interpretation. In such cases, the more "probabilistic" a rule becomes, the harder it is to interpret. On the other hand, from an inference perspective, algorithms based on sampling and belief propagation perform better if the rules are not nearly deterministic. Similar to the classic "bias-variance" tradeoff in machine learning, there seems to be a probability-specificity tradeoff in probabilistic logic methods.

If prediction accuracy were the only criterion for preference of models, whether more specific and hence deterministic or more probabilistic models is better depends on the loss function. In terms of Ockham's razor of preferring simpler models, it isn't clear how to trade off the complexity of deterministic rules with stochastic predictions, as we usually do not use the optimal information-theoretic encoding for probabilities or for the structure.

5 Inference

Inference in probabilistic relational models refers to computing the posterior distribution of some variables given some evidence.

A standard way to carry out inference in such models is to try to generate and ground as few of the parameterized random variables as possible. E.g., in the ICL, the relevant ground instances can be carried out using abduction [97].

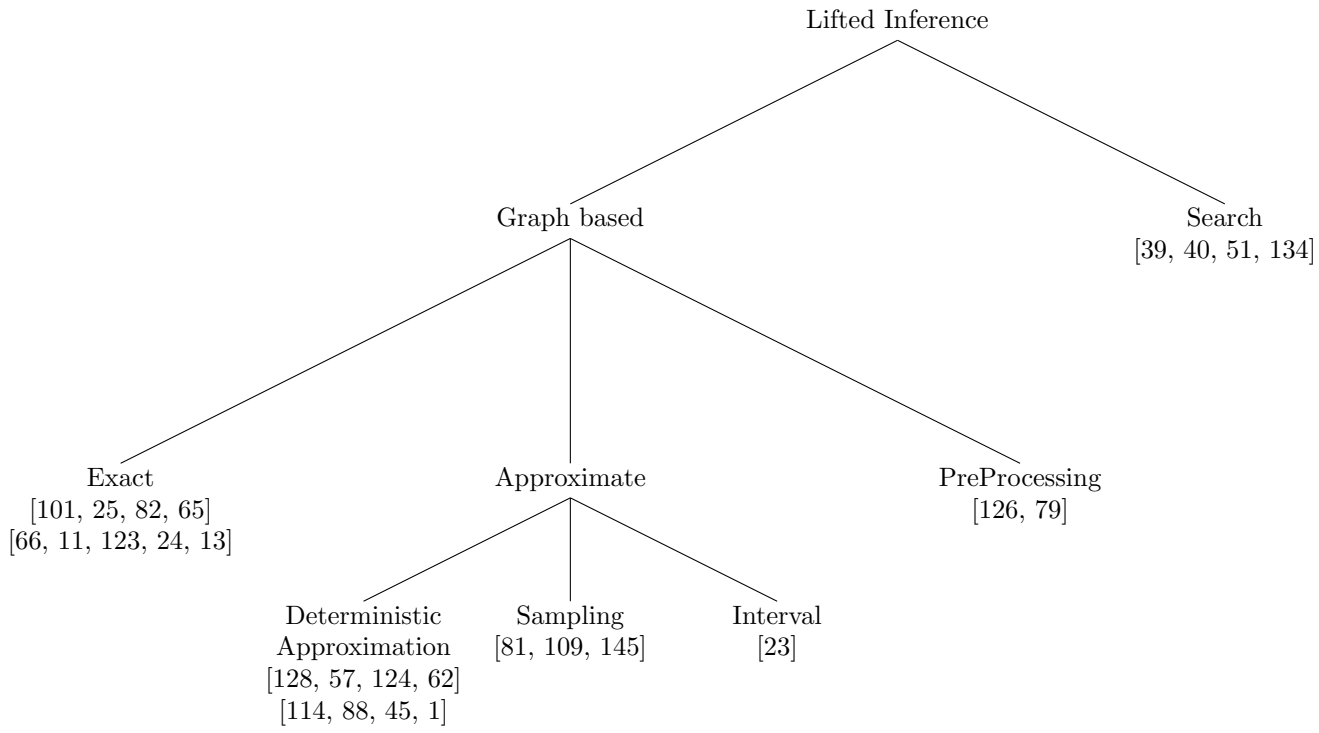

Figure 6: Classification of the *current* Lifted Inference Methods

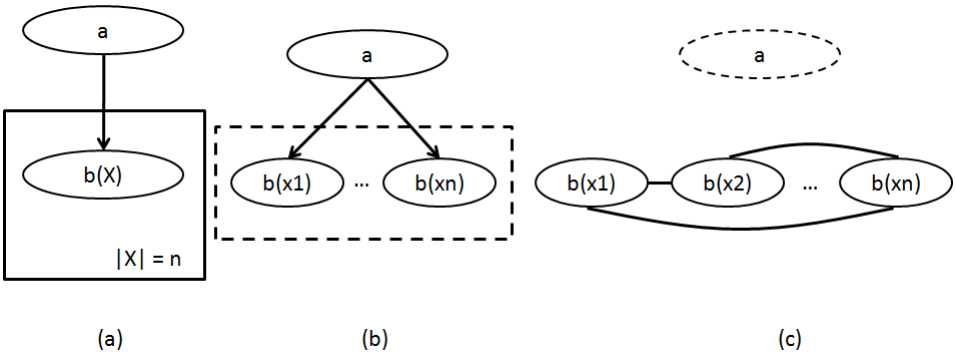

Figure 7: Lifted Inference Example. (a) Parfactor consists of two predicates a and b. (b) The case when summing out b. All ground instances of b can be grouped together. (c) The case when eliminating a. All ground instances of b are now connected.

More recently, there has been work on lifted probabilistic inference [101, 25, 128, 82], where the idea is to carry out probabilistic reasoning at the lifted level, without grounding out the parameterized random variables. Instead, we count how many of the probabilities we need, and when we need to multiply a number of identical probabilities, we can take the probability to the power of the number of individuals. Lifted inference turns out to be a very difficult problem, as the possible interactions between parameterized random variables can be very complicated. Nevertheless, there is already substantial work on lifted inference.

A taxonomy of lifted inference methods is presented in Figure 6. At a fairly high-level, lifted inference algorithms can be understood as being inspired by graphical models (shown as *graph based* in the Figure) or by logical approaches (shown as *search based*). It should be mentioned that graph vs search based classification is orthogonal to exact vs approximate classification. But, since the search based methods [39, 40, 51, 134] are more recent and fewer in number, we grouped them as a separate class and focus in this section mainly on the graph based methods. The graph based methods can be divided into *exact* inference methods (which are based on variable elimination), *approximate* methods and methods for *pre-processing* the data.

To understand the exact inference methods, consider a simple parfactor on $\langle\{\}, \{a, b(X)\}, t\rangle$, where the population of X has size n (see Figure 7(a)). When summing out all instances of $b(X)$ (as shown in Figure 7 (b)), we can note that all of the factors in the grounding have the same value and so can be taken to the power of n, which can be done in time logarithmic in n [101], whereas the grounding is linear in n. This operation invented by Poole [101] was called inversion elimination by de Salvo Braz et al. [25]. However, if we were to sum out a instead (as in Figure 7 (c)), in the resulting grounding all instances of

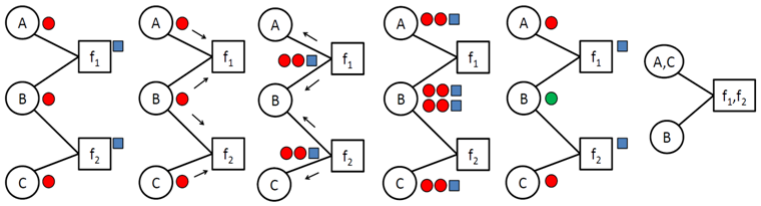

Figure 8: Illustration of lifted BP algorithm. From left to right, the steps of the lifted inference algorithm taken to compress the factor graph assuming no evidence. The shaded/colored small circles and squares denote the groups and signatures produced running the algorithm. On the right-hand side, the resulting compressed factor graph is shown. On this network a modified BP algorithm is ran.

$b(X)$ are connected, and so there would be a factor that is of size exponential in n. de Salvo Braz et al. [25] showed how, rather than representing the resulting factor, we only need to the count of the number of instances of $b(X)$ which have a certain value, and so the subsequent elimination of $b(X)$ can be done in time polynomial in n (this is linear in n if $b(X)$ is binary, and if $b(X)$ has k values, the time is $O(n^{k-1})$). Milch et al. [82] proposed counting formulae as a representation of the intermediate lifted formulae. All of these algorithms need to ground a population in certain circumstances.

In a distinct yet related work, Sen et al. [123] propose the idea of lifted variable elimination where computational trees are used to group together indistinguishable individuals. These indistinguishable individuals are determined by identifying shared factors (i.e., factors that compute the same function and have the same input and output values). This idea was later extended by the same group to approximate inference [124]. Choi et al. [13] addressed the problem of lifted inference in continuous domains. Their approach assumes that the model consists of Gaussian potentials. Their algorithm marginalizes variables by integrating out random variables using inversion elimination operation. If the elimination is not possible, they consider elimination of pairwise potentials and the marginals that are not in pairwise form are converted to pairwise form and then eliminated.

The approximate methods can be further classified into *deterministic approximation* methods that are based on variational methods such as belief propagation, *sampling* based methods, and *interval* methods. The deterministic approximate methods group random variables and factors in to sets if they have identical message passing computation trees [128, 57, 62]. Consider the factor graph presented in Figure 8 with three nodes A, B and C sending identical messages to the neighbors. As can be seen from the figure, since nodes A and C send and receive the same message, they can be clustered together. B cannot

be grouped with A or C since it sends twice the same message and hence is considered to be different from the other two. The resulting compressed graph is shown in the end. These methods have been successfully applied to several different problems and extended in several ways [88, 62, 45, 1].

There are sampling methods developed based on MCMC for certain formalisms such as MLNs [109] and BLOG [81]. Zettlemoyer et al. [145] extended particle filters to logical setting. Braz et al. [23] proposed a method that starts from the query, propagates intervals instead of point estimates and incrementally grounds the lifted network. Lastly, there are some methods for pre-processing [126, 79] that reduce the network size drastically so that ground inference can be performed efficiently on the reduced network.

The search based methods form the dual to the graph-based methods in analogy to how variable elimination is the dynamic programming variant of search-based recursive conditioning [16]. The method by Gogate and Domingos [40] reduce the problem of lifted probabilistic inference to weighted model counting in a lifted graph. Another recent approach to lifted inference compiles the lifted graph into a weighted CNF and then runs weighted model counting on the compiled network [134]. Both these approaches were developed in parallel and have promising potential to lifted inference.

Lifted inference has come a long way since Poole's lifted variable elimination and is currently a very active area of research inside Statistical Relational AI as can be evidenced by a surge in the number of algorithms and applications in the last few years. For instance, belief propagation based methods have been successfully applied to model counting [57], social networks [62] and content distribution problems [62]. Nath and Domingos used an approximate version of Lifted Belief Propagation for video segmentation [88]. Ahmadi et al. [1] applied the lifted evidence message passing to problems such as PageRank and Kalman Filters. In a related work, Choi et al. [12] developed Relational Gaussian Models to model large dynamic systems and developed an exact inference for Lifted Kalman filters. Sen et al.'s [123, 124] lifted variable elimination has been used for efficient inference in probabilistic data bases. Other applications of lifted inference techniques include information extraction and retrieval, semantic role labeling, citation matching and entity resolution [81, 109, 128, 126, 114, 89, 40]. Lifted inference in continuous models have been applied to market analysis [13].

6 Learning

Consider a single relation, that records the grade for students in courses[3]:

[3]In a real dataset, each student would be identified by unique (meaningless) identifier, such as a student number. Similarly a course would also have a unique identifier, and there would be some way to get the information about the department, number, and term taken. In a real database there would be much more information about each course and each student.

Student	Course	Grade
Sam	cs245	85
Chris	cs222	55
Sam	cs222	76
Chris	cs333	65
...	...	...

The aim might be to predict how Sam would do on another course, say cs333. This problem has been studied in terms of reference classes [111], where it has been advocated to use the narrowest reference class for which there are adequate statistics [69]. Here that might mean to use the grade of Sam, the average grade in the course cs333 or some mix of all reference classes. Suppose that the average grade of all students in all courses was 65, and that the average grade of Sam was 80 and the average grade of cs333 was also 80. One might think to predict somewhere between 65 and 80, as that is the range of the statistics available. However, it is likely that Sam will get a higher grade than 80; after all Sam is a well-above-average student and cs333 is an easy course[4].

In the simplest case, we have a model such as in Figure 3, where all of the relations have been observed. In such cases, parameter sharing can lead to learning conditional probabilities by counting or by adapting supervised learning techniques [119, 34, 61]. For example, the maximum likelihood value for the shared parameter $P(gr(S,C) = a | i(S) = high, d(C) = low)$ could be the ratio of counts

$$\frac{|\{\langle S, C\rangle : gr(S,C) = a, i(S) = high, d(C) = low\}|}{|\{\langle S, C\rangle : i(S) = high, d(C) = low\}|} .$$

This produces the probability that a randomly chosen student-course pair with a highly intelligent student and a low difficulty course, will have an "a" grade. This will typically produce a low probability in a large data base, as most students have not taken most courses. This might be what is wanted. If however, we want to predict how a intelligent student would do on an easy course, we should only consider the courses students have taken, and then the maximum likelihood value for $P(gr(S,C) = a | i(S) = high, d(C) = low)$ would be the ratio:

$$\frac{|\{\langle S, C\rangle : gr(S,C) = a, i(S) = high, d(C) = low\}|}{|\{\langle S, C\rangle : i(S) = high, d(C) = low, \exists G\ gr(S,C) = G\}|} .$$

Typically, however, we don't directly observe the difficulty of courses and the intelligence of students. A more sophisticated way to model such predictions is in terms of latent (hidden or unobserved) variables. For example, one could adopt the model of Figure 3, but where $i(S)$ and $d(C)$ are latent variables. Of course, when these are unobserved variables, it may not be the intelligence of the students and the difficulty of the courses that are discovered by the learning; a learner will discover whichever categorizations best predict the observations. Note that inducing properties of individuals is equivalent to a (soft) clustering

[4]It could also be the case that the average for cs333 is high because cs333 is only taken by the top students. We would also like the learning system to discover this.

of the individuals. For example, if there are three values for $i(S)$, assigning a value to $i(s)$ for each student s is equivalent to assigning s to one of three clusters. The use of latent variables allows for much greater modelling than can be done with reference classes [10].

Although the difficulty of the courses and the intelligence of the students are a priori independent, they become dependent once some grades are observed. For example, if students s and c have taken a course in common (and the difficulty of the course is not observed), $i(s)$ and $i(c)$ are dependent, and if c and k have also taken another course in common, all three random variables are interdependent. Thus inference in these models quickly becomes intractable. Since inference is a key step in learning (for computing the expected counts[5]) learning such models is intractable.

Next to estimating the parameter from data, we may also select the model structure, i.e., a set of parameterized factors from data [119, 35, 20, 67]. This can be viewed as adapting traditional inductive logic programming (ILP) techniques to take uncertainty explicitly into account. Whereas ILP typically employs some $0-1$ score to evaluate hypotheses (the number of correctly covered examples), learning probabilistic relational models employs some probabilistic score such as (pseudo) likelihood (the likelihood of correctly covering the examples).

With this in mind, the vanilla structure learning algorithm for probabilistic relational models might be sketched as a greedy hill-climbing search algorithm [20, 22]. Assuming some data given, we take some initial theory T_0, say gr, i, and d are not inter-connect by any edges, as starting point and compute the parameters maximizing some score such as the (pseudo) likelihood. Then, we use so-called refinement operators to compute neighbours of T_0. A refinement operator takes the current theory, makes small, syntactic modifications to it, and returns a copy of the modified theory. Whereas for Bayesian networks, typical refinements are adding, deleting, or flipping single edges, for relational models, we instead add or delete single literals to formulas, negate them, or instantiate respectively unify variables in them. For instance we may hypothesise that the grade $gr(S, C)$ of student S in course C depends on the intelligence $i(S)$ of the student S. This essentially corresponds to adding, deleting, or flipping multiple edges in the underlying ground graphical model. In our case, we add an edge for each student S and course C pair. Now, if the score for one of the neighbours, say H, is larger then the currently best theory T_0, we take H as new current best theory T_1 and iterate. The process is continued until no further improvements in score are obtained.

Recently, there have been some advances to this vanilla learning approach, especially in the case of Markov Logic networks. For instance, we may view a given set of examples (a relational database) as a hypergraph. A hypergraph is a straightforward generalization of a graph, in which an edge can link any number

[5]In the presence of missing data or latent factors, the maximum likelihood estimate typically cannot be written in closed form. It is a numerical optimization problem, and all well known algorithms such as the Expectation-Maximization (EM) algorithm or gradient-based methods involve nonlinear, iterative optimization and multiple calls to inference for computing the expected counts.

of nodes, rather than just two. Now, the constants appearing in an example are the nodes and ground atoms are the hyperedges. Each hyperedge is labeled with the predicate symbol of the corresponding ground atom. Nodes (constants) are linked by a hyperedge if and only if they appear as arguments in the hyperedge. Now, any path of hyperedges can be generalized into conjunctions of relational atoms by variablizing their arguments. Mihalkova and Mooney [78] as well as Kok and Domingos [68] proposed to use relational path finding [112] for learning Markov logic networks. Each path is turned into a set of conjunctions of atoms for which we estimate the weights.

Another recent advance is triggered by the insight that finding many rough rules of thumb of how to change probabilistic models locally can be a lot easier than finding a single, highly accurate model. For relational dependency networks, for example, one can represent the conditional probability distribution associated with each predicate as a weighted sum of regression models grown in a stage-wise optimization using gradient boosting [85]. This functional gradient approach has also successfully been used to train conditional random fields for labeling relational sequences [44] and to learn relational policies [60]. The benefits of a boosted learning approach are manifold. First, being a nonparametric approach the number of parameters grows with the number of training episodes. In turn, interactions among random variables are introduced only as needed, so that the potentially large search space is not explicitly considered. Second, such an algorithm is fast and straightforward to implement. Existing off-the-shelf regression learners can be used to deal with propositional, continuous, and relational domains in a unified way. Third, it learns the structure and parameters simultaneously, which is an attractive feature as learning probabilistic relational models is computationally quite expensive.

We may also take a quite different path to learning. Whereas we can find the most likely model given the data [119, 35] using for example the vannila structure learning approach sketched above, we may also take a Bayesian perspective and average over all models. From a Bayesian point of view, learning is just a case of inference: we condition on all of the observations (all of the data), and determine the posterior distribution over some hypotheses or any query of interest. Starting from the work of Buntine [7], there has been considerable work in using relational models for Bayesian learning [52]. This work uses parameterized random variables (or the equivalent plates) and the probabilistic parameters are real-valued random variables (perhaps parameterized). Dealing with real-valued variables requires sophisticated reasoning techniques often in terms of MCMC and stochastic processes. Although these methods use relational probabilistic models for learning, the representations learned are typically not relational probabilistic models.

What is important about learning is that we want to learn general theories that can be learned before the agent know the individuals, and so before the agent knows the random variables.

Statistical relational models have been used for estimating the result size of complex database queries [37], for clustering gene expression data [122], and for discovering cellular processes from gene expression data [121]. The have also

been used for understanding tuberculosis epidemiology [36]. Probabilistic relational trees have discovered publication patterns in high-energy physics [77]. They have also been used to learn to rank brokers with respect to the probability that they would commit a serious violation of securities regulations in the near future [90]. Relational Markov networks have been used for semantic labeling of 3D scan data [2]. They have also been used to compactly represent object maps [76] and to estimate trajectories of people [75]. Relational hidden Markov models have been used for protein fold recognition [59]. Markov logic networks have been proven to be successful for joint unsupervised coreference resolution and unsupervised semantic parsing using Markov logic networks [107, 108]. Non-parametric relational models have been used for analysing social networks [140], for classification [14], link prediction [143] and for learning to rank search results [63, 139]. Most exciting, non-parametric relational models that perform probabilistic inference over hierarchies of flexibly structured representations can address some of the deepest questions about the nature and origins of human thought, see [132] and references in there.

7 Actions

There is also a large body of work on relational representations of actions under uncertainty. The initial work in this area was on representations, in terms of the event calculus [99] or the situation calculus [100, 3][6]. Representing actions in uncertain domains is challenging because to plan, an agent needs to be concerned, not only about its current uncertainty and its percepts, but also about what information will be available for future decisions. These models combined perception, action and utility to form first-order variants of fully-observable and partially-observable Markov decision processes.

Later work has concentrated on how to do planning with such representations either for the fully observable case [5, 117, 136] or the partially observable case [137, 118]. The promise of being able to carry out lifted inference much more efficiently is slowly being realized. In essence, symbolic dynamic programming — a generalization of the dynamic programming technique for solving propositional Markov decision processes — exploits the symbolic structure in the solution of relational and first-order logical Markov decision processes through a lifted version of dynamic programming. It constructs a minimal logical partition of the state space required to make all necessary value distinctions.

Consider for instance an agent acting in a simple variant of the BoxWorld problem [142]. There are several cities such as *london*, *paris* etc., trucks $truck_1$, $truck_2$ etc., and boxes box_1, box_2 etc. The agent can load a box onto a truck

- $load(Box : b, Truck : t, City : c)$:

[6]These two papers are interesting because they make the opposite design decisions on almost all of the design choices. For example, whether an agent knowns what situation it is in, and whether a situation implies what is true: we can't have both for a non-omniscient agent.

  - Success Probability: if $(BoxIn(b, c) \wedge TruckIn(t, c))$ then .9 else 0
  - Add Effects on Success: $\{BoxOn(b, t)\}$
  - Delete Effects on Success: $\{BoxIn(b, c)\}$

or unload it and can drive a truck from one city to another. Only when a particular box, say box box_1, is in a particular city, say *paris*, the agent receives a positive reward.

- Reward:
  - if $(BoxIn(b, paris)$ then 10 else 0

The agent's task is now to find a policy for action selection in each relational situation that maximizes its reward over the long term. In our example, the agent may figure out the following. To get box b to *paris*, the agent drives a truck to the city of b, loads box_1 on the truck, drives the truck to *paris*, and finally unloads the box box_1 in *paris*. This is achieved through the operations of first-order decision-theoretic regression and symbolic maximization. While the details and implementations of these operations are depending on the relational MDP framework used, see e.g. [5, 117, 136], and hence are beyond the scope of the present paper, one should note that the operations are exactly the lifted versions of the traditional dynamic programming solution to Markov decision processes. In our running example, applying them to the 0-stages-to-go value function, i.e., the reward function given previously, yields the following 1- and 2-stages-to-go value functions in the BoxWorld domain (in case notation as used by [117]; $\neg$" indicating the conjunction of the negation of all higher value partitions):

$$vCase^1 = \begin{array}{|l|r|}\hline \exists b.BoxIn(b, paris) & : 19.0 \\ \hline \neg\text{"} \wedge \exists b, t.TruckIn(t, paris) \wedge BoxOn(b, t) & : 9.0 \\ \hline \neg\text{"} & : 0.0 \\ \hline \end{array}$$

$$vCase^2 = \begin{array}{|l|r|}\hline \exists b.BoxIn(b, paris) & : 27.1 \\ \hline \neg\text{"} \wedge \exists b, t.TruckIn(t, paris) \wedge BoxOn(b, t) & : 17.1 \\ \hline \neg\text{"} \wedge \exists b, c, t.BoxOn(b, t) \wedge TruckIn(t, c) & : 8.1 \\ \hline \neg\text{"} & : 0.0 \\ \hline \end{array}$$

After sufficient iterations, the t-stages-to-go value function converges. The key features to note are the state and action abstraction in the value and policy representation that are afforded by the first-order specification and solution of the problem. That is, this solution does not refer to any specific set of domain objects, such as $City = \{paris, berlin, london\}$, but rather it provides a solution for *all possible domain object instantiations*. And while classical dynamic programming techniques could never solve these problems for large domain instantiations (since they would have to enumerate all states and actions), a domain-independent lifted solution to this particular problem is quite simple due to the power of state and action abstraction.

Since the basic symbolic dynamic programming approach, a variety of exact algorithms have been introduced to solve MDPs with relational and first-order structure. First-order value iteration [48, 55] and the relational Bellman algorithm [64] are value iteration algorithms for solving relational MDPs. In addition, first-order decision diagrams have been introduced to compactly represent case statements and to permit efficient application of symbolic dynamic programming operations to solve relational MDPs via value iteration and policy iteration [136]. All of these algorithms have some form of guarantee on convergence to the (ϵ-)optimal value function or policy. Furthermore, a class of linear-value approximation algorithms have been introduced to approximate the value function as a linear combination of weighted basis functions. First-order approximate linear programming [117] directly approximates the relational value function using a linear program. Other heuristic solutions for instance induces rule-based policies from sampled experience in small-domain instantiations of relational MDPs and generalizes these policies to larger domains [31]. In a similar vein, Gretton and Thiebaux [43] used the action regression operator in the situation calculus to provide the first-order hypothesis space for an inductive policy learning algorithm. One can also turn the relational MDP into a structured dynamic Bayesian network representation and predict the effects of action sequences using approximate inference and beliefs over world states [71]. Recently, Lang and Toussaint [70] and Joshi et al. [53] have shown that successful planning typically involves only a small subset of relevant objects and have shown how to make use of this fact to speed up symbolic dynamic programming significantly.

Solvers for relational MDPs have been successfully applied in decision-theoretic planning domains such as BlocksWorld, BoxWorld, ZenoWorld, Elevators, Drive, PitchCatch and Schedule comparing well to or even outperforming propositional counterparts. Related techniques have been used to solve path planning problems within robotics and instances of real-time strategy games, Tetris, and Digger.

Finally, there is also work on relational reinforcement learning, see [129, 135] for overviews, where an agent learns what to do before knowing what individuals will be encountered, and so before it knows what random variables exist. As an example, consider the idea of having household robots, which just need to be taken out of their shipping boxes, turned on, and then do some cleaning work. This "robot-out-of-the-box" has inspired research in robotics as well as in machine learning and artificial intelligence. Without a compact knowledge representation that supports abstraction by and unification of logical placeholders, and in turn generalization of previous experiences to the current state and potential future states, however, it seems to be difficult — if not hopeless — to explore one's home in reasonable time. There are simply too many objects a household robot may deal with such as doors, plates, boxes and water-taps. To deal with this "curse of dimensionality", a number of model-free [29, 27, 28, 26, 110, 60] as well as model-based, see e.g. [92, 71], relational reinforcement learning approaches have been developed. A key insight is that the inherent generalization of learnt knowledge in the relational representation has profound implications

also on the exploration strategy: what in a propositional setting would be considered a novel situation and worth exploration may in the relational setting be an instance of a well-known context in which exploitation is promising [72]. For instance, after having opened one or two water taps in the kitchen, say $wt(obj1), in(obj1, k1), k(k1)$ and $wt(obj2), in(obj2, k2), k(k2)$, to fill the sink with water, the household robot can expect other water-taps to behave similarly. Thus, the priority for exploring water-taps in kitchens $wt(X), in(X, Y), k(Y)$ in general should be reduced and not just the one for $wt(obj1), in(obj1, k1), k(k1)$ and $wt(obj2), in(obj2, k), in(k2)$. Moreover, our information gathered about water-taps $wt(X), in(X, Y), k(Y)$ so far should also transfer to water-taps in laundries, say $wt(obj3), in(obj3, l1), l(l1)$ since we have also learned something about $wt(X), in(X, Y)$. Without extensive feature engineering this would be difficult — if not impossible — in a propositional setting. We would simply encounter a new and therefore unexplored situation.

In general, robotics is an emerging application area for StaRAI techniques [4, 133]. As robots are starting to perform everyday manipulation tasks, such as cleaning up, setting a table or preparing simple meals, they must become much more knowledgeable than they are today. Typically, everyday tasks are specified vaguely and the robot must therefore infer what are the appropriate actions to take and which are the appropriate objects involved in order to accomplish these tasks. These inferences can only be done if the robot has access to general world knowledge.

8 Identity and Existence Uncertainty

The previously outlined work assumes that an agent knows which individuals exist and can identify them. The problem of knowing whether two descriptions refer to the same individual is known as identity uncertainty [91, 102]. This arises in citation matching [91] when we need to distinguish whether two references refer to the same paper and in record linkage [30], where the aim is to determine if two hospital records refer to the same person (e.g., whether the current patient who is requesting drugs been at the hospital before). To solve this, we have the hypotheses of which descriptions refer to the same individuals, and which refer to different ones. If there are n descriptions, an assignment of equality to these descriptions corresponds to a partitioning of the descriptions (each description in a partition corresponds to the same individual, and different partitions correspond to different individuals). The number of partitions on n elements is the Bell number, which grows faster than any exponential.

The problem of knowing whether some individual exists is known as existence uncertainty [80, 102]. This is challenging because when existence is false, there is no individual to refer to, and when existence is true, there may be many individuals that fit a description. We may have to know which individual a description is referring to. In general, determining the probability of an observation requires knowing the protocol for how observations were made. For example, if an agent considers a house and declares that there is a green room,

the probability of this observation depends on what protocol they were using: did they go looking for a green room, did they report the colour of the first room found, did they report the type of the first green thing found, or did they report on the colour of the first thing they perceived?

9 Ontologies and Semantic Science

Data that are reliable and people care about, particularly in the sciences, are being reported using the vocabulary defined in formal ontologies [32]. The next stage in this line of research is to represent scientific hypotheses that also refer to formal ontologies and are able to make probabilistic predictions that can be judged against data [104]. This work combines all of the issues of relational probabilistic modelling as well as the problems of describing the world at multiple level of abstraction and detail, and handling multiple heterogenous data sets. It also requires new ways to think about ontologies [105], and new ways to think about the relationships between data, hypotheses and decisions.

10 Conclusions

Real agents need to deal with their uncertainty and reason about individuals and relations. They need to learn how the world works before they have encountered all the individuals they need to reason about. If we accept these premises, then we need to get serious about relational probabilistic models. There is a growing community under the umbrella of statistical relational learning that is tackling the problems of decision making with models that refer to individuals and relations. While there have been considerable advances in the last two decades, there are more than enough problems to go around to establish what has come to be called statistical relational AI.

Acknowledgements Krisitan Kersting was supported by the Fraunhofer ATTRACT fellowship STREAM and by the European Commission under contract number FP7-248258-First-MM. Sriraam Natarajan acknowledges the support of Translational Science Institute of Wake Forest University School of Medicine. David Poole is supported by NSERC.

References

[1] Ahmadi, B., Kersting, K., and Hadiji, F. (2010). Lifted belief propagation: Pairwise marginals and beyond. In T.J. P. Myllymaeki T. Roos (Ed.), *Proceedings of the 5th European Workshop on Probabilistic Graphical Models (PGM–10)*. Helsinki, Finland.

[2] Anguelov, D., Taskar, B., Chatalbashev, V., Koller, D., Gupta, D., Heitz, G., and Ng, A. (2005). Discriminative Learning of Markov Random Fields

for Segmentation of 3D Scan Data. In C. Schmid, S. Soatto, and C. Tomasi (Eds.), *IEEE Computer Society International Conference on Computer Vision and Pattern Recognition (CVPR-05)*, volume 2, pp. 169–176. San Diego, CA, USA.

[3] Bacchus, F., Halpern, J.Y., and Levesque, H.J. (1999). Reasoning about noisy sensors and effectors in the situation calculus. *Artificial Intelligence*, 111(1–2): 171–208. URL `http://www.lpaig.uwaterloo.ca/~fbacchus/on-line.html`.

[4] Beetz, M., Jain, D., Mösenlechner, L., and Tenorth, M. (2010). Towards performing everyday manipulation activities. *Robotics and Autonomous Systems*, 58.

[5] Boutilier, C., Reiter, R., and Price, B. (2001). Symbolic dynamic programming for first-order MDPs. In *Proc. 17th International Joint Conf. Artificial Intelligence (IJCAI-01)*.

[6] Breese, J.S. (1992). Construction of belief and decision networks. *Computational Intelligence*, 8(4): 624–647.

[7] Buntine, W.L. (1994). Operations for learning with graphical models. *Journal of Artificial Intelligence Research*, 2: 159–225.

[8] Carbonetto, P., Kisynski, J., de Freitas, N., and Poole, D. (2005). Nonparametric bayesian logic. In *Proc. 21st Conf. on Uncertainty in AI (UAI)*.

[9] Chavira, M. and Darwiche, A. (2005). Compiling bayesian networks with local structure. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 1306–1312.

[10] Chiang, M. and Poole, D. (2011). Reference classes and relational learning. *International Journal of Approximate Reasoning*. doi:DOI:10.1016/j.ijar.2011.05.002.

[11] Choi, J., de Salvo Braz, R., and Bui, H. (2011). Efficient methods for lifted inference with aggregate factors. In *AAAI 2011*.

[12] Choi, J., Guzman-Rivera, A., and Amir, E. (2011). Lifted relational kalman filtering. In *IJCAI*, pp. 2092–2099.

[13] Choi, J., Hill, D., and Amir, E. (2010). Lifted inference for relational continuous models. In *UAI'10: Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, pp. 126–134. AUAI Press, Corvallis, Oregon, USA.

[14] Chu, W., Sindhwani, V., Ghahramani, Z., and Keerthi, S. (2006). Relational learning with gaussian processes. In *Neural Information Processing Systems*.

[15] Darwiche, A. (2002). A logical approach to factoring belief networks. In *Proceedings of KR*, pp. 409–420.

[16] Darwiche, A. (2001). Recursive conditioning. *Artificial Intelligence*, 126(1-2): 5–41.

[17] de Finetti, B. (1931). Funzione caratteristica di un fenomeno aleatorio. *Atti della R. Accademia Nazionale dei Lincei, Ser. 6. Memorie, Classe di Scienze Fisiche, Matematiche e Naturali 4*, pp. 251–299.

[18] De Raedt, L. (2008). *Logical and Relational Learning*. Springer.

[19] De Raedt, L. and Kersting, K. (2003). Probabilistic Logic Learning. *ACM-SIGKDD Explorations: Special issue on Multi-Relational Data Mining*, 5(1): 31–48.

[20] De Raedt, L. and Kersting, K. (2004). Probabilistic Inductive Logic Programming. In S. Ben-David, J. Case, and A. Maruoka (Eds.), *Proceedings of the 15th International Conference on Algorithmic Learning Theory (ALT-04)*, volume 3244 of *LNCS*, pp. 19–36. Springer, Padova, Italy.

[21] De Raedt, L., Kimmig, A., and Toivonen, H. (2007). ProbLog: A probabilistic Prolog and its application in link discovery. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI-2007)*, pp. 2462–2467.

[22] De Raedt, L., Frasconi, P., Kersting, K., and Muggleton, S.H. (Eds.) (2008). *Probabilistic Inductive Logic Programming*. Springer.

[23] de Salvo Braz, R., Natarajan, S., Bui, H., Shavlik, J., and Russell, S. (2009). Anytime lifted belief propagation. In *Statistical Relational Learning Workshop*.

[24] de Salvo Braz, R., Amir, E., and Roth, D. (2006). Mpe and partial inversion in lifted probabilistic variable elimination. In *Proceedings of the 21sth National Conference on Artificial Intelligence (AAAI)*.

[25] de Salvo Braz, R., Amir, E., and Roth, D. (2007). Lifted first-order probabilistic inference. In L. Getoor and B. Taskar (Eds.), *Introduction to Statistical Relational Learning*. M.I.T. Press. URL `http://www.cs.uiuc.edu/~eyal/papers/BrazRothAmir_SRL07.pdf`.

[26] Driessens, K. and Dzeroski, S. (2005). Combining model-based and instance-based learning for first-order regression. In *Proc. International Conf. on Machine Learning*, pp. 193–200.

[27] Driessens, K. and Ramon, J. (2003). Relational instance based regression for relational reinforcement learning. In *Proc. International Conf. on Machine Learning*, pp. 123–130.

[28] Driessens, K., Ramon, J., and Gärtner, T. (2006). Graph kernels and Gaussian processes for relational reinforcement learning. *Machine Learning Journal*.

[29] Dzeroski, S., de Raedt, L., and Driessens, K. (2001). Relational reinforcement learning. *Machine Learning*, 43: 7–52.

[30] Fellegi, I. and Sunter, A. (1969). A theory for record linkage. *Journal of the American Statistical Association*, 64(328): 1183–1280.

[31] Fern, A., Yoon, S., and Givan, R. (2003). Approximate policy iteration with a policy language bias. In *NIPS-2003*. Vancouver.

[32] Fox, P., McGuinness, D., Middleton, D., Cinquini, L., Darnell, J., Garcia, J., West, P., Benedict, J., and Solomon, S. (2006). Semantically-enabled large-scale science data repositories. In *5th International Semantic Web Conference (ISWC06)*, volume 4273 of *Lecture Notes in Computer Science*, pp. 792–805. Springer-Verlag. URL `http://www.ksl.stanford.edu/KSL_Abstracts/KSL-06-19.html`.

[33] Frey, B.J., Kschischang, F.R., Loeliger, H.A., and Wiberg, N. (1997). Factor graphs and algorithms. In *Proceedings of the 35th Allerton Conference on Communication, Control, and Computing*, pp. 666–680. Champaign-Urbana, IL. URL `http://www.psi.toronto.edu/~psi/pubs2/1999%20and%20before/134.pdf`.

[34] Friedman, N., Getoor, L., Koller, D., and Pfeffer, A. (1999). Learning probabilistic relational models. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI-99)*, pp. 1300–1309. Morgan Kaufman, Stockholm, Sweden.

[35] Getoor, L., Friedman, N., Koller, D., and Pfeffer, A. (2001). Learning probabilistic relational models. In S. Dzeroski and N. Lavrac (Eds.), *Relational Data Mining*, pp. 307–337. Springer-Verlag.

[36] Getoor, L., Rhee, J., Koller, D., and Small, P. (2004). Understanding tuberculosis epidemiology using probabilistic relational models. *Journal of Artificial Intelligence in Medicine*, 30: 233–256.

[37] Getoor, L., Taskar, B., and Koller, D. (2001). Using probabilistic models for selectivity estimation. In *Proceedings of ACM SIGMOD International Conference on Management of Data*, pp. 461–472. ACM Press.

[38] Getoor, L. and Taskar, B. (Eds.) (2007). *Introduction to Statistical Relational Learning*. MIT Press, Cambridge, MA.

[39] Gogate, V. and Domingos, P. (2010). Exploiting logical structure in lifted probabilistic inference. In *AAAI 2010 Workshop on Statististical and Relational Artificial Intelligence (STAR-AI)*. URL `http://aaai.org/ocs/index.php/WS/AAAIW10/paper/view/2049`.

[40] Gogate, V. and Domingos, P. (2011). Probabilistic theorem proving. In *Proc. 27th Conf. Uncertainty in AI*.

[41] Goldman, R.P. and Charniak, E. (1990). Dynamic construction of belief networks. In *Proc. 6th Conference on Uncertainty in Artificial Intelligence*, pp. 90–97.

[42] Goodman, N., Mansinghka, V., Roy, D.M., Bonawitz, K., and Tenenbaum, J. (2008). Church: a language for generative models. In *Proc. Uncertainty in Artificial Intelligence (UAI)*. URL `http://web.mit.edu/droy/www/papers/church\_GooManRoyBonTenUAI2008.pdf`.

[43] Gretton, C. and Thiebaux, S. (2004). Exploiting first-order regression in inductive policy selection. In *UAI-04*, pp. 217–225. Banff, Canada.

[44] Gutmann, B. and Kersting, K. (2006). Tildecrf: Conditional random fields for logical sequences. In M.S. J. Fuernkranz T. Scheffer (Ed.), *Proceedings of the 17th European Conference on Machine Learning (ECML–2006)*, pp. 174–185. Berlin, Germany.

[45] Hadiji, F., Kersting, K., and Ahmadi, B. (2010). Lifted message passing for satisfiability. In *Working Notes of the AAAI10 Workshop on Statistical Relational AI (StarAI)*. AAAI Press.

[46] Heckerman, D. and Breese, J. (1994). A new look at causal independence. In *UAI*.

[47] Heckerman, D., Meek, C., and Koller, D. (2004). Probabilistic models for relational data. Technical Report MSR-TR-2004-30, Microsoft Research.

[48] Hölldobler, S. and Skvortsova, O. (2004). A logic-based approach to dynamic programming. In *In AAAI-04 Workshop on Learning and Planning in MDPs*, pp. 31–36. Menlo Park, CA.

[49] Horsch, M. and Poole, D. (1990). A dynamic approach to probabilistic inference using Bayesian networks. In *Proc. Sixth Conference on Uncertainty in AI*, pp. 155–161. Boston.

[50] Jaeger, M. (2007). Parameter learning for Relational Bayesian networks. In *ICML*.

[51] Jha, A., Gogate, V., Meliou, A., and Suciu, D. (2010). Lifted inference from the other side: The tractable features. In *Twenty-Fourth Annual Conference on Neural Information Processing Systems (NIPS)*.

[52] Jordan, M.I. (2010). Bayesian nonparametric learning: Expressive priors for intelligent systems. In R. Dechter, H. Geffner, and J.Y. Halpern (Eds.), *Heuristics, Probability and Causality: A Tribute to Judea Pearl*, pp. 167–186. College Publications.

[53] Joshi, S., Kersting, K., and Khardon, R. (2010). Self-taught decision theoretic planning with first order decision diagrams. In *Proceedings of ICAPS-10*.

[54] Kallenberg, O. (2005). *Probabilistic symmetries and invariance principles*. Springer.

[55] Karabaev, E. and Skvortsova, O. (2005). A heuristic search algorithm for solving first-order MDPs. In *UAI-2005*, pp. 292–299. Edinburgh, Scotland.

[56] Kemp, C., Tenenbaum, J., Griffiths, T., Yamada, T., and Ueda, N. (2006). Learning systems of concepts with an infinite relational model. In *Proc. 21st AAAI*.

[57] Kersting, K., Ahmadi, B., and Natarajan, S. (2009). Counting belief propagation. In J.B. A. Ng (Ed.), *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI–09)*. Montreal, Canada.

[58] Kersting, K. and De Raedt, L. (2007). Bayesian logic programming: Theory and tool. In L. Getoor and B. Taskar (Eds.), *An Introduction to Statistical Relational Learning*. MIT Press.

[59] Kersting, K., De Raedt, L., and Raiko, T. (2006). Logial Hidden Markov Models. *Journal of Artificial Intelligence Research (JAIR)*, 25: 425–456.

[60] Kersting, K. and Driessens, K. (2008). Non–parametric policy gradients: A unified treatment of propositional and relational domains. In S.R. A. McCallum (Ed.), *Proceedings of the 25th International Conference on Machine Learning (ICML 2008)*. Helsinki, Finland.

[61] Kersting, K. and Driessens, K. (2008). Non-parametric policy gradients: A unified treatment of propositional and relational domains. In *Proceedings of the 25th International Conference on Machine Learning (ICML)*.

[62] Kersting, K., Massaoudi, Y.E., Ahmadi, B., and Hadiji, F. (2010). Informed lifting for message–passing. In D.P. M. Fox (Ed.), *Twenty–Fourth AAAI Conference on Artificial Intelligence (AAAI–10)*. AAAI Press, Atlanta, USA.

[63] Kersting, K. and Xu, Z. (2009). Learning preferences with hidden common cause relations. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD 09)*, LNAI. Springer, Bled, Slovenia.

[64] Kersting, K., van Otterlo, M., and de Raedt, L. (2004). Bellman goes relational. In *ICML-04*. ACM Press, Banff, Alberta, Canada. doi:http://doi.acm.org/10.1145/1015330.1015401.

[65] Kisynski, J. and Poole, D. (2009). Constraint processing in lifted probabilistic inference. In *Proc. 25th Conference on Uncertainty in AI, (UAI-2009)*, pp. 293–302. Montreal, Quebec. URL `http://www.cs.ubc.ca/~poole/papers/KisynskiUAI2009.pdf`.

[66] Kisynski, J. and Poole, D. (2009). Lifted aggregation in directed first-order probabilistic models. In *Proc. Twenty-first International Joint Conference on Artificial Intelligence (IJCAI-09)*, pp. 1922–1929. Pasadena, California.

[67] Kok, S. and Domingos, P. (2007). Learning the structure of Markov logic networks. In *Proc. of the Intern. Conf. on Machine Learning (ICML-05)*, pp. 441–448.

[68] Kok, S. and Domingos, P. (2009). Learning Markov Logic Network Structure via Hypergraph Lifting. In *Proc. of the International Conference on Machine Learning (ICML-09)*.

[69] Kyburg, H. (1983). The reference class. *Philosophy of Science*, 50: 374–397.

[70] Lang, T. and Toussaint, M. (2009). Relevance grounding for planning in relational domains. In *Proc. European Conf. on Machine Learning*.

[71] Lang, T. and Toussaint, M. (2010). Planning with noisy probabilistic relational rules. *Journal of Artificial Intelligence Research (JAIR)*, 39: 1–49.

[72] Lang, T., Toussaint, M., and Kersting, K. (2010). Exploration in relational worlds. In J. Balcazar, F. Bonchi, A. Gionis, and M. Sebag (Eds.), *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD-10)*. Springer, Barcelona, Spain.

[73] Laskey, K.B. (2008). MEBN: A language for first-order Bayesian knowledge bases. *Artificial Intelligence*, 172(2-3). doi:doi:10.1016/j.artint.2007.09.006. URL `http://www.sciencedirect.com/science/article/B6TYF-4PTMXXP-1/2/ce6bcf1c5a5fecfd805501056e9b62a1`.

[74] Lavrac, N. and Dzeroski, S. (1994). *Inductive Logic Programming: Techniques and Applications*. Ellis Horwood, NY.

[75] Liao, L., Fox, D., and Kautz, H. (2005). Location-Based Activity Recognition using Relational Markov Networks. In F. Giunchiglia and L.P. Kaelbling (Eds.), *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI-05)*, pp. 773–778. AAAI Press, Edinburgh, Scotland.

[76] Limketkai, B., Liao, L., and Fox, D. (2005). Relational Object Maps for Mobile Robots. In F. Giunchiglia and L.P. Kaelbling (Eds.), *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI-05)*, pp. 1471–1476. AAAI Press, Edinburgh, Scotland.

[77] McGovern, A., Friedland, L., Hay, M., Gallagher, B., Fast, A., Neville, J., and Jensen, D. (2003). Exploiting relational structure to understand publication patterns in high-energy physics. *SIGKDD Explorations*, 5(2): 165–173.

[78] Mihalkova, L. and Mooney, R. (2007). Bottom-Up Learning of Markov Logic Network Structure. In *Proc. of the Intern. Conf. on Machine Learning (ICML-07)*.

[79] Mihalkova, L. and Richardson, M. (2009). Speeding up inference in statistical relational learning by clustering similar query literals. In *In Proceedings of the 19th International Conference on Inductive Logic Programming (ILP-09*.

[80] Milch, B., Marthi, B., Russell, S., Sontag, D., Ong, D.L., and Kolobov, A. (2005). BLOG: Probabilistic models with unknown objects. In *Proc. 19th International Joint Conf. Artificial Intelligence (IJCAI-05)*. Edinburgh.

[81] Milch, B. and Russell, S. (2006). General-purpose mcmc inference over relational structures. In *In Proceedings of the Proceedings of the Twenty-Second Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI06*, pp. 349–358. AUAI Press.

[82] Milch, B., Zettlemoyer, L.S., Kersting, K., Haimes, M., and Kaelbling, L.P. (2008). Lifted probabilistic inference with counting formulas. In *Proceedings of the Twenty Third Conference on Artificial Intelligence (AAAI)*. URL `http://people.csail.mit.edu/lpk/papers/mzkhk-aaai08.pdf`.

[83] Muggleton, S. (1996). Stochastic logic programs. In L. De Raedt (Ed.), *Advances in Inductive Logic Programming*, pp. 254–264. IOS Press.

[84] Muggleton, S. and De Raedt, L. (1994). Inductive logic programming: Theory and methods. *Journal of Logic Programming*, 19,20: 629–679.

[85] Natarajan, S., Khot, T., Kersting, K., Gutmann, B., and Shavlik, J. (2011). Gradient–based boosting for statistical relational learning: The relational dependency network case. *Machine Learning Journal*. (Online First).

[86] Natarajan, S., Tadepalli, P., Dietterich, T.G., and Fern, A. (2009). Learning first-order probabilistic models with combining rules. *Special Issue on Probabilistic Relational Learning, AMAI*.

[87] Natarajan, S., Khot, T., Lowd, D., Tadepalli, P., Kersting, K., Shavlik, J., and Iais, F. (2010). Exploiting causal independence in markov logic networks: Combining undirected and directed models. In *Proceedings of ECML*.

[88] Nath, A. and Domingos, P. (2010). Efficient lifting for online probabilistic inference. In *AAAI*.

[89] Neumann, M., Kersting, K., and Ahmadi, B. (2011). Markov logic sets: Towards lifted information retrieval using pagerank and label propagation. In D.R. W. Burgard (Ed.), *Proceedings of the Twenty–Fifth AAAI Conference on Artificial Intelligence (AAAI–11)*. AAAI Press, San Francisco, CA, USA.

[90] Neville, J., Simsek, Ö., Jensen, D., Komoroske, J., Palmer, K., and Goldberg, H. (2005). Using relational knowledge discovery to prevent securities fraud. In *Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM Press.

[91] Pasula, H., Marthi, B., Milch, B., Russell, S., and Shpitser, I. (2003). Identity uncertainty and citation matching. In *NIPS*, volume 15.

[92] Pasula, H., Zettlemoyer, L., and Pack Kaelbling, L. (2007). Learning symbolic models of stochastic domains. *Artificial Intelligence Research*, 29: 309–352.

[93] Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo, CA.

[94] Perlich, C. and Provost, F. (2006). Distribution-based aggregation for relational learning with identifier attributes. In *Machine Learning*.

[95] Pfeffer, A. (2007). The design and implementation of IBAL: A general-purpose probabilistic language. In L. Getoor and B. Taskar (Eds.), *Statistical Relational Learning*. MIT Press.

[96] Poole, D. (1991). Representing diagnostic knowledge for probabilistic Horn abduction. In *Proc. 12th International Joint Conf. on Artificial Intelligence (IJCAI-91)*, pp. 1129–1135. Sydney.

[97] Poole, D. (1993). Logic programming, abduction and probability: A top-down anytime algorithm for computing prior and posterior probabilities. *New Generation Computing*, 11(3–4): 377–400.

[98] Poole, D. (1993). Probabilistic Horn abduction and Bayesian networks. *Artificial Intelligence*, 64(1): 81–129.

[99] Poole, D. (1997). The independent choice logic for modelling multiple agents under uncertainty. *Artificial Intelligence*, 94: 7–56. URL `http://cs.ubc.ca/~poole/abstracts/icl.html`. Special issue on economic principles of multi-agent systems.

[100] Poole, D. (1998). Decision theory, the situation calculus and conditional plans. *Electronic Transactions on Artificial Intelligence*, 2(1–2). URL `http://www.etaij.org`.

[101] Poole, D. (2003). First-order probabilistic inference. In *Proc. Eighteenth International Joint Conference on Artificial Intelligence (IJCAI-03)*, pp. 985–991. Acapulco, Mexico.

[102] Poole, D. (2007). Logical generative models for probabilistic reasoning about existence, roles and identity. In *22nd AAAI Conference on AI (AAAI-07)*. URL `http://cs.ubc.ca/~poole/papers/AAAI07-Poole.pdf`.

[103] Poole, D. (2008). The independent choice logic and beyond. In L. De Raedt, P. Frasconi, K. Kersting, and S. Muggleton (Eds.), *Probabilistic Inductive Logic Programming: Theory and Application*, LNCS 4911. Springer Verlag. URL `http://cs.ubc.ca/~poole/papers/ICL-Beyond.pdf`.

[104] Poole, D., Smyth, C., and Sharma, R. (2008). Semantic science: Ontologies, data and probabilistic theories. In P.C. da Costa, C. d'Amato, N. Fanizzi, K.B. Laskey, K. Laskey, T. Lukasiewicz, M. Nickles, and M. Pool (Eds.), *Uncertainty Reasoning for the Semantic Web I*, LNAI/LNCS. Springer. URL `http://cs.ubc.ca/~poole/papers/SemSciChapter2008.pdf`.

[105] Poole, D., Smyth, C., and Sharma, R. (2009). Ontology design for scientific theories that make probabilistic predictions. *IEEE Intelligent Systems*, 24(1): 27–36. URL `http://www2.computer.org/portal/web/computingnow/2009/0209/x1poo`.

[106] Poole, D.L. and Mackworth, A.K. (2010). *Artificial Intelligence: foundations of computational agents*. Cambridge University Press, New York, NY. URL `http://artint.info`.

[107] Poon, H. and Domingos, P. (2008). Joint unsupervised coreference resolution with markov logic. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Honolulu, HI, USA.

[108] Poon, H. and Domingos, P. (2009). Unsupervised semantic parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Singapore.

[109] Poon, H., Domingos, P., and Sumner, M. (2008). A general method for reducing the complexity of relational inference and its application to mcmc. In *Proceedings of the 23rd national conference on Artificial intelligence - Volume 2*, pp. 1075–1080.

[110] Ramon, J., Driessens, K., and Croonenborghs, T. (2007). Transfer learning in reinforcement learning problems through partial policy recycling. In *Proc. European Conf. on Machine Learning*, pp. 699–707.

[111] Reichenbach, H. (1949). *The Theory of Probability*. University of California Press,.

[112] Richards, B. and Mooney, R. (1992). Learning relations by pathfinding. In M.S. J. Fuernkranz T. Scheffer (Ed.), *Proceedings of the 10th National Conference on Artificial Intelligence (AAAI-92)*, pp. 174–185. San Jose, CA, USA.

[113] Richardson, M. and Domingos, P. (2006). Markov logic networks. *Machine Learning*, 62: 107–136.

[114] Riedel, S. (2008). Improving the accuracy and efficiency of map inference for markov logic. In *Proceedings of the 24th Annual Conference on Uncertainty in AI (UAI '08)*, pp. 468–475.

[115] Russell, S. and Norvig, P. (2010). *Artificial Intelligence: A Modern Approach*. Series in Artificial Intelligence. Prentice Hall, Upper Saddle River, NJ, third edition. URL `http://aima.cs.berkeley.edu/`.

[116] Sang, T., Beame, P., and Kautz, H. (2005). Solving Bayesian networks by weighted model counting. In *Proc. Twentieth National Conference on Artificial Intelligence (AAAI-05)*, pp. 475–482.

[117] Sanner, S. and Boutilier, C. (2009). Practical solution techniques for first order mdps. *AIJ*, 173: 748–788.

[118] Sanner, S. and Kersting, K. (2010). Symbolic dynamic programming for first-order POMDPs. In *Proc. AAAI-2010*.

[119] Sato, T. and Kameya, Y. (1997). PRISM: A symbolic-statistical modeling language. In *Proceedings of the 15th International Joint Conference on Artificial Intelligence (IJCAI-97)*, pp. 1330–1335.

[120] Sato, T. and Kameya, Y. (2008). New advances in logic-based probabilistic modeling by PRISM. In L. De Raedt, P. Frasconi, K. Kersting, and S. Muggleton (Eds.), *Probabilistic Inductive Logic Programming*, volume LNCS 4911, pp. 118–155. Springer. URL `http://www.springerlink.com/content/1235t75977x62038/`.

[121] Segal, E., Battle, A., and Koller, D. (2003). Decomposing gene expression into cellular processes. In *Proc. Pacific Symposium on Biocomputing (PSB)*, pp. 89–100. World Scientific.

[122] Segal, E., Taskar, B., Gasch, A., Friedman, N., and Koller, D. (2001). Rich probabilistic models for gene expression. *Bioinformatics*, 17(Suppl 1): S243–52. Proc. ISMB 2001.

[123] Sen, P., Deshpande, A., and Getoor, L. (2008). Exploiting shared correlations in probabilistic databases. *Proc. VLDB Endow.*, 1: 809–820.

[124] Sen, P., Deshpande, A., and Getoor, L. (2009). Bisimulation-based approximate lifted inference. In *Uncertainty in Artificial Intelligence*.

[125] Shafer, G. and Shenoy, P. (1990). Probability propagation. *Annals of Mathematics and Artificial Intelligence*, 2: 327–352.

[126] Shavlik, J. and Natarajan, S. (2009). Speeding up inference in markov logic networks by preprocessing to reduce the size of the resulting grounded network. In *Proceedings of the 21st international jont conference on Artifical intelligence*, pp. 1951–1956.

[127] Silva, R., Chu, W., and Ghahramani, Z. (2007). Hidden common cause relations in relational learning. In *Neural Information Processing Systems*.

[128] Singla, P. and Domingos, P. (2008). Lifted first-order belief propagation. In *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence*, pp. 1094–1099.

[129] Tadepalli, P., Givan, R., and Driessens, K. (2004). Relational reinforcement learning: An overview. In *Proc. ICML Workshop on Relational Reinforcement Learning*.

[130] Talbott, W. (2008). Bayesian epistemology. In E.N. Zalta (Ed.), *The Stanford Encyclopedia of Philosophy*. URL `http://plato.stanford.edu/archives/fall2008/entries/epistemology-bayesian/`.

[131] Taskar, B., Abbeel, P., and Koller, D. (2002). Discriminative Probabilistic Models for Relational Data. In A. Darwiche and N. Friedman (Eds.), *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence (UAI-02)*, pp. 485–492. Edmonton, Alberta, Canada.

[132] Tenenbaum, J., Kemp, C., Griffiths, T., and Goodman, N. (2011). How to grow a mind: Statistics, structure, and abstraction. *Science*, 331(11 March 2011): 1279–1285.

[133] Tenorth, M., Klank, U., Pangercic, D., and Beetz, M. (2011). Web-enabled robots – robots that use the web as an information resource. *IEEE Robotics and Automation Magazine*, 18.

[134] Van den Broeck, G., Taghipour, N., Meert, W., Davis, J., and De Raedt, L. (2011). Lifted probabilistic inference by first-order knowledge compilation. In *Proc. International Joint Conference on AI (IJCAI)*, pp. 2178–2185.

[135] van Otterlo, M. (2009). *The Logic of Adaptive Behavior - Knowledge Representation and Algorithms for Adaptive Sequential Decision Making under Uncertainty in First-Order and Relational Domains*. IOS Press. URL `http://people.cs.kuleuven.be/~martijn.vanotterlo/phdbook_vanOtterlo_v2010a.pdf`.

[136] Wang, C., Joshi, S., and Khardon, R. (2008). First order decision diagrams for relational mdps. *JAIR*, 31: 431–472.

[137] Wang, C. and Khardon, R. (2010). Relational partially observable MDPs. In *Proc. AAAI-2010*.

[138] Wick, M., McCallum, A., and Miklau, G. (2010). Scalable probabilistic databases with factor graphs and mcmc. 3(1): 794–804.

[139] Xu, Z., Kersting, K., and Joachims, T. (2010). Fast active exploration for link–based preference learning using gaussian processes. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD–10)*. Springer, Barcelona, Spain.

[140] Xu, Z., Tresp, V., Rettinger, A., and Kersting, K. (2009). Social network mining with nonparametric relational models. In *Advances in Social Network Mining and Analysis*, LNCS. Springer.

[141] Xu, Z., Tresp, V., Yu, K., and Kriegel, H.P. (2006). Infinite hidden relational models. In *Proc. 22nd UAI*.

[142] Younes, H., Littman, M., Weissman, D., and Asmuth, J. (2005). The first probabilistic track of the international planning competition. *JAIR*, 24: 851–887.

[143] Yu, K. and Chu, W. (2007). Gaussian process models for link analysis and transfer learning. In *Neural Information Processing Systems*.

[144] Yu, K., Chu, W., Yu, S., Tresp, V., and Xu, Z. (2006). Stochastic relational models for discriminative link prediction. In *Neural Information Processing Systems*.

[145] Zettlemoyer, L.S., Pasula, H.M., and Kaelbling, L.P. (2007). Logical particle filtering. In *In Proceedings of the Dagstuhl Seminar on Probabilistic, Logical, and Relational Learning*.

[146] Zhang, N.L. and Poole, D. (1994). A simple approach to Bayesian network computations. In *Proc. 10th Canadian Conference on AI*, pp. 171–178.