Journal of Artificial Intelligence Research 63 (2018) 789-848 Submitted 04/18; published 12/18

State-Space Abstractions for Probabilistic Inference: A Systematic Review

Stefan Lüdtke STEFAN.LUEDTKE2@UNI-ROSTOCK.DE
Institute of Computer Science
University of Rostock, Germany

Max Schröder MAX.SCHROEDER@UNI-ROSTOCK.DE
Frank Krüger FRANK.KRUEGER@UNI-ROSTOCK.DE
Institute of Communications Engineering
University of Rostock, Germany

Sebastian Bader SEBASTIAN.BADER@UNI-ROSTOCK.DE
Thomas Kirste THOMAS.KIRSTE@UNI-ROSTOCK.DE
Institute of Computer Science
University of Rostock, Germany

Abstract

Tasks such as social network analysis, human behavior recognition, or modeling biochemical reactions, can be solved elegantly by using the probabilistic inference framework. However, standard probabilistic inference algorithms work at a propositional level, and thus cannot capture the symmetries and redundancies that are present in these tasks.

Algorithms that exploit those symmetries have been devised in different research fields, for example by the lifted inference-, multiple object tracking-, and modeling and simulation-communities. The common idea, that we call *state space abstraction*, is to perform inference over compact representations of sets of symmetric states. Although they are concerned with a similar topic, the relationship between these approaches has not been investigated systematically.

This survey provides the following contributions. We perform a systematic literature review to outline the state of the art in probabilistic inference methods exploiting symmetries. From an initial set of more than 4,000 papers, we identify 116 relevant papers. Furthermore, we provide new high-level categories that classify the approaches, based on common properties of the approaches. The research areas underlying each of the categories are introduced concisely. Researchers from different fields that are confronted with a state space explosion problem in a probabilistic system can use this classification to identify possible solutions. Finally, based on this conceptualization, we identify potentials for future research, as some relevant application domains are not addressed by current approaches.

1. Introduction

Many real-world problems are inherently symmetric. For example, human behavior recognition from sensor data (Fox, Hightower, Liao, Schulz, & Borriello, 2003), social network analysis (Singla & Domingos, 2008), and models of biochemical reactions (Barbuti, Levi, Milazzo, & Scatena, 2011) all have symmetric properties. These application scenarios are also probabilistic: We do not have perfect knowledge about the state of the system, and the system can develop non-deterministically over time. Performing probabilistic inference in these domains quickly leads to a combinatorial explosion, known as *state space explosion*

problem (Clarke, Grumberg, Jha, Lu, & Veith, 2001). To overcome this problem, probabilistic inference approaches that exploit symmetric properties of the system have been devised. In this survey, we systematically review the literature on these approaches and develop a new conceptual model to classify the approaches. Previous surveys on this topic (Kersting, 2012; Kimmig, Mihalkova, & Getoor, 2015) have focussed on a specific class of such algorithms, known as *lifted inference*. In this review, we put more emphasis on inference in sequential processes (known as *Bayesian filtering*, a method that is highly relevant for many different application domains), and consider algorithms devised in a number of different research fields, like control theory, modeling and simulation, and computer vision.

To give an intuition of the state space explosion problem, we give some initial examples that show how it manifests itself in different domains.

Example 1 (Friends and Smokers, Singla & Domingos, 2008)**.** The relationship of smoking habits and lung cancer is modeled. People who smoke are more likely to develop lung cancer, and friends tend to have similar smoking habits. We can model this problem as a Bayesian network with one random variable for the smoking probability of each person, one random variable for the cancer risk of each person, and one random variable for each pair of people that represents whether they are friends or not. The number of random variables and the treewidth of the graphical model grows linearly with the number of people, and thus the inference time grows exponentially (as inference is NP-hard in the treewidth of the model). ◦

Example 2 (Office, Fox et al., 2003)**.** Several people walk around in an office. The office is equipped with presence sensors that get activated when a person is nearby. The sensors do not identify the specific person that is near the sensor. The task is to keep track of the locations of each person. An inference algorithm has to track an exponential number of possible situations (all possible permutations of observations to person identities). ◦

Example 3 (Biochemical Reaction, Barbuti et al., 2011)**.** Biochemical reactions can involve many different reactants. In each specific reaction, many instances of the same molecule can participate in that reaction. A naive algorithm has to consider an exponential number of specific reactions (one for each combination of specific molecule instances) that can take place. ◦

In all of these cases, standard probabilistic inference algorithms are not suitable, due to the exponential growth in problem complexity. However, we can exploit the symmetries underlying each of these problems: In Example 1, the probability of each person having cancer is the same, as long as we have the same information about each person. We can therefore reason over all people simultaneously, by only representing the probability of a generic person having cancer. In Example 2, people are not *identified*. Thus, all states that are only different in the assignment of names to people cannot be distinguished and can be grouped together. In Example 3, it does not matter which specific molecule participates in the reaction, as the result of the reaction is the same. In all of the examples, the general idea is to represent multiple concrete (or *grounded*) states that are symmetrical by a single abstract state (also called *lifted* state). In this paper, we identify two types of symmetries, based on *exchangeability* in state variables or on variables following the same *parametric*

distribution. In the following, we call the procedure of grouping symmetrical states *state space abstraction*. To perform inference efficiently, an inference algorithm must be able to reason directly with the abstract states, without resorting to grounded states.

This systematic review aims at giving an overview of different methods of state space abstractions for probabilistic models, and inference algorithms that exploit these abstractions. The format of a *systematic* literature review has been chosen because state space abstractions have been considered in different research communities (e.g. probabilistic inference, see Kersting, 2012; control theory, see Nitti, De Laet, & De Raedt, 2014; modeling and simulation, see Maus, Rybacki, & Uhrmacher, 2011; computer vision, see Huang, Guestrin, & Guibas, 2009b, etc.). A systematic review is the appropriate tool in this case, because it reduces the chance to miss out relevant contributions from different research areas.

The contribution of this paper is a novel structure of the research field that is based on an *application-centric* classification of the approaches. That is, approaches in the same class can exploit symmetries in the same problem domain.

Recently, attempts have been made to formally structure the problem classes of lifted inference algorithms, by investigating which structures of a probabilistic model allow efficient (lifted) probabilistic inference (Jaeger & Van den Broeck, 2012). For lifted inference algorithms, this classification is precise and robust – we present this classification in Appendix A. However, it does not address algorithms for models containing continuous distributions, or dynamic models. In contrast, our classification is much more coarse-grained and informal (all problem classes they consider fall in the same category in our classification), but it applies to a broader range of algorithms.

By using this classification, for the first time, this review draws connections between previously distinct lines of research, like lifted inference, logical filtering, and multiset rewriting, and outlines the common idea shared by these approaches – the use of *state space abstractions*. We hope that this structure helps researchers from different research fields that are confronted with a state space explosion in a probabilistic system to identify possible solutions. Finally, we identify potential future research directions.

We proceed as follows. In Section 2, we introduce the basic concepts used in the rest of the paper. Section 3 contains a description of the properties that are used to characterize the approaches. In Section 4, we describe the systematic procedure we applied for retrieving, selecting and analyzing the relevant work. An empirical overview of the retrieved papers is presented in Section 4.5. Section 5 contains the analysis of the retrieved papers, regarding the criteria proposed in Section 3. This evaluation leads to a categorization of the approaches, regarding the problem class they are concerned with. Each of the resulting groups is described separately. We conclude in Section 7, by discussing the results of this review, and proposing future research directions.

2. Preliminaries

This chapter gives a brief overview over basic concepts used in the remainder of the paper. It consists of two parts: Section 2.1 and 2.2 introduce the basic concepts and algorithms used in the context of probabilistic inference. Sections 2.3 and 2.4 introduce the two basic concepts for state space abstractions that are discussed in this review: Lifted graphical models and

Rao-Blackwellization. Each state space abstraction approach that we will discuss is based on either of these two concepts.

2.1 Graphical Models and Probabilistic Inference

In this section, we introduce the basic concepts of probabilistic inference, and briefly present three algorithms that are the basis for the lifted inference algorithms discussed in Section 5.1. For a more thorough introduction to graphical models, see the book by Koller and Friedman (2009).

2.1.1 Graphical Models

Probabilistic graphical models are a way to compactly represent a joint probability distribution $P(X_1, \ldots, X_n)$ that exhibits certain independence assumptions. They represent a joint probability distribution over multiple random variables (RVs) $X_1, \ldots, X_n$ by decomposing the distribution $P(X_1, \ldots, X_n)$ into a set of factors F. Each factor $\phi \in F$ maps a vector of RV assignments to non-negative real numbers, and the product of all factors describes the joint distribution (together with a normalization constant Z ensuring that the total probability sums to one):

$$P(X_1 = x_1, \ldots, X_n = x_n) = Z^{-1} \prod_{\phi \in F} \phi(x_\phi) \tag{1}$$

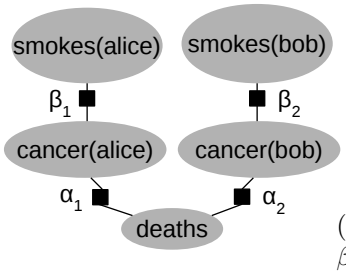
x_ϕ denotes the subset of values of RVs that is necessary to compute the factor ϕ. A factor of binary RVs is often represented as a table (for example, see Figure 1). A factor graph is a depiction of the relationship between factors and RVs. RVs are depicted by a circle, and factors by a box (see Figure 1). Edges between factors and RVs mean that the RV is part of the factor.

Thus, graphical models provide a compact representation for probability distributions: Instead of representing a distribution over, for example, n binary variables by a factor of size 2^n (a table with 2^n rows), the distribution is represented by a set of much smaller factors. This also makes reasoning about the distribution more efficient, as described later.

Bayesian networks and Markov networks can be seen as special cases of factor graphs, where the factors are defined implicitly by the graph structure. Bayesian networks are directed graphical models. The nodes represent RVs and an edge from a node X to a node Y means that the distribution of the RV Y depends on the RV X. Markov networks are undirected graphical models, where nodes represent RVs, and there is a factor for each maximal clique in the graph that takes the nodes of the clique as arguments.

Consider the scenario introduced in Example 1. We present a slightly adapted version of this scenario here (omitting the *friends* relation for simplicity).

Example 4 (Smokers)**.** Each person either smokes or does not smoke. For people who smoke, the chance of getting cancer is higher than for people who do not smoke. Whether or not at least one person died last year depends on the number of people who have cancer. ○



(a) Factor graph.

s(a)	c(a)	β_1
0	0	$\beta_1^{(00)}$
0	1	$\beta_1^{(01)}$
1	0	$\beta_1^{(10)}$
1	1	$\beta_1^{(11)}$

(b) Factor β_1. The values $\beta^{(xx)}$ are real numbers. The factor β_2 looks similar (see text).

c(a)	d	α_1
0	0	$\alpha_1^{(00)}$
0	1	$\alpha_1^{(01)}$
1	0	$\alpha_1^{(10)}$
1	1	$\alpha_1^{(11)}$

(c) Factor α_1. The values $\alpha^{(xx)}$ are real numbers. The factor α_2 looks similar (see text).

Figure 1: Factor graph of Example 4 (adapted from Richardson & Domingos, 2006).

For now, let us assume that only two people, Alice and Bob, exist. We can then model this scenario with the binary random variables *smokes*(*alice*), *cancer*(*alice*), *smokes*(*bob*), *cancer*(*bob*) and *death*[1]. The factor graph for this scenario can be seen in Figure 1.

The factor graph describes a joint probability by multiplying all of the factors, for example:

$$\begin{aligned} &P(s(a)=1, s(b)=1, c(a)=0, c(b)=0, d=0) \\ =&Z^{-1}\ \beta_1(s(a)=1, c(a)=0)\ \beta_2(s(b)=1, c(b)=0)\ \alpha_1(d=0, c(a)=0)\ \alpha_2(d=0, c(b)=0) \end{aligned} \tag{2}$$

Note that this example shows the need (and potential) for employing abstractions: We see that there is a certain redundancy in the model: The factors β_1 and β_2 as well as α_1 and α_2 are identical, when we exchange $s(a)$ and $c(a)$ for $s(b)$ and $c(b)$. If we want to add more people to the model, we need similar random variables and factors for each person. This behavior is the main motivation for employing state space abstractions: To be able to reason over these redundant variables as a group, ideally independently of the number of people (domain objects) involved.

2.1.2 Inference Algorithms

Given a graphical model, we can answer different questions. In our example, we may want to know the probability that Alice has cancer, or the expected number of deaths. These questions fall into different categories: *Conditional probability queries* $p(Q \mid E{=}e)$, where the goal is to compute the conditional probability of some variables Q, given values of *evidence* variables E, *Maximum-a-posteriori (MAP) queries* $\mathrm{MAP}(Q \mid E{=}e) = \arg\max_q p(Q{=}q, E{=}e)$ that ask for the most likely joint assignment of variables, given values of evidence variables, and *marginal MAP queries* $\mathrm{MMAP}(S \mid E{=}e)$ that ask for the most likely assignment of a subset $S \subset Q$ of variables, while the other variables $Q \setminus S$ are marginalized.

The process of calculating answers to these questions is called *probabilistic inference*. Inference can always be performed by computing the complete joint distribution, and summing out (marginalizing) the variables we are not interested in. However, the reason for

1. For readability, we use $c(a)$ and $c(b)$ instead of *cancer*(a) and *cancer*(b), $s(a)$ and $s(b)$ instead of *smokes*(a) and *smokes*(b), and d instead of *death*.

using graphical models in the first place was to avoid computing the complete joint distribution, so efficient inference algorithms avoid this. The remainder of this section will focus on conditional probability queries[2].

Variable Elimination Variable elimination (VE) (Zhang & Poole, 1994) is an inference algorithm for conditional probability queries that operates on a factor graph. It eliminates the non-query and non-evidence variables one by one without computing the entire joint probability. A variable is eliminated by multiplying all factors that contain this variable, and then summing out (marginalizing) this variable. The performance depends on the order in which the variables are eliminated, and thus heuristics for good elimination orderings have been proposed (Darwiche, 2009).

Example 5. Consider the graphical model of Example 4 and the query $P(s(a), s(b), d{=}1)$[3]. VE eliminates the non-query and non-evidence variables $c(a)$ and $c(b)$ one by one: The RV $c(a)$ is eliminated by multiplying the factor α_1 and β_1, resulting in a factor f_0 that has the following representation as a table (with 8 rows):

s(a)	c(a)	d	f_0
0	0	0	$\beta_1^{(00)}\,\alpha_1^{(00)}$
0	0	1	$\beta_1^{(00)}\,\alpha_1^{(01)}$
⋮	⋮	⋮	⋮

The RV $c(a)$ is summed out of f_0, resulting in a factor

$$f_1(s(a), d) = \sum_v f_0(s(a), c(a){=}v, d) = \sum_v \beta_1(s(a), c(a){=}v)\,\alpha_1(c(a){=}v, d)$$

that is represented by the following table:

s(a)	d	f_1
0	0	$\beta_1^{(00)}\,\alpha_1^{(00)} + \beta_1^{(01)}\,\alpha_1^{(10)}$
0	1	$\beta_1^{(00)}\,\alpha_1^{(01)} + \beta_1^{(01)}\,\alpha_1^{(11)}$
⋮	⋮	⋮

Thus, the distribution $P(s(a), s(b), c(b), d)$ can be represented by the factors α_2, β_2 and f_1 as follows:

$$P(s(a), s(b), c(b), d) = Z^{-1}\, f_1(s(a), d)\,\beta_2(s(b), c(b))\,\alpha_2(c(b), d))$$

2. MAP queries can be answered by adapting conditional probability inference algorithms (like variable elimination), or by specialized optimization algorithms. MMAP requires to calculate a marginal probability for each explored assignment of MAP variables, and thus in general is harder than the other query types. MMAP can be solved by search-based algorithms (Marinescu, Dechter, & Ihler, 2015).
3. This query is the first step in answering the conditional probability query $P(s(a), s(b) \,|\, d) = P(s(a), s(b), d)/P(d)$.

Afterwards, the same procedure is performed for $c(b)$: α_2 and β_2 are multiplied, $c(b)$ is marginalized, the result is multiplied with f_1. The result directly represents the distribution of the above query. ∘

In this example, the computations for eliminating $c(a)$ and $c(b)$ are similar, which hints to the possibility of performing the elimination more efficiently, as shown in Section 5.1.

Recursive Conditioning Recursive conditioning (RC) (Darwiche, 2001) is the search-based variant of VE. Instead of summing out RVs, it branches on the value of RVs. Once all information to evaluate a factor are present, it is evaluated directly, and the values of all branches are combined appropriately. The presentation of RC given here is based on the description of De Raedt, Kersting, Natarajan, and Poole (2016).

Given a partially instantiated factor graph, the following cases are distinguished: (i) If there is a factor that can be evaluated, i.e. all RVs of this factor are instantiated, then it is evaluated, and RC is called on the remaining factor graph. The result of the factor evaluation and the RC call are multiplied. (ii) Otherwise, an RV is selected to branch on, RC is called recursively for each possible value of the RV, and the results of all recursive calls are summed. Furthermore, caching can be used to avoid repeated evaluation of the same expression, and disconnected components can be treated independently.

Example 6. Consider the same problem as in Example 5, i.e. the graphical model of Example 4 and the query $P(s(a), s(b), d{=}1)$. RC starts with only $d = 1$ instantiated, i.e. no factor can be evaluated. The algorithm selects $c(a)$ for branching, leading to the two branches b_1 where $\{d = 1, c(a) = 0\}$ and b_2 where $\{d = 1, c(a) = 1\}$. In both cases, the factor α_1 can be evaluated, and the algorithm is called with the remaining factor graph. In the following, the algorithm branches on the other RVs $c(b)$, $s(a)$ and $s(b)$. The factor evaluations in each branch are multiplied, and the results of each branch are summed. ∘

Belief Propagation Belief propagation (BP) (Pearl, 1988) is a message-passing inference algorithm, related to the forward-backward algorithm used in Hidden Markov Models. It is exact for acyclic factor graphs, and provides an approximate solution for factor graphs with cycles. The idea is that each node (i.e. each RV node and each factor node) in a factor graph sends *messages* to its neighbors, based on the messages it receives.

Let x be an RV node (of the RV x) and f be a factor node (of the factor f). Messages are passed either from an RV node to a factor node ($\mu_{x\to f}$) or from a factor node to an RV node ($\mu_{f\to x}$). The messages are partial functions with domain $\mathrm{dom}(x)$, i.e. vectors of length $|\mathrm{dom}(x)|$. The intuition on the messages $\mu_{f\to x}(x_j)$ is that the values are proportional to how likely node f "thinks" the RV corresponding to node x is in the state x_j.

More specifically, the messages are calculated as follows: The message sent from an RV node x to a factor node f is the multiplicative summary of the message it received:

$$\mu_{x\to f}(x_i) = \prod_{f'\in n(x)\setminus\{f\}} \mu_{f'\to x}(x_i)$$

$n(x)$ denotes the set of neighboring nodes of x in the factor graph. The message sent from a factor node f to an RV node x is

$$\mu_{f\to x}(x_i) = \sum_{\mathbf{y}} \left(f(x_i, \mathbf{y}) \prod_{x' \in n(f)\setminus\{x\}} \mu_{x'\to f}(\mathbf{y}) \right)$$

The summation is over all possible assignments $\mathbf{y} \in \{dom(x') \mid x' \in n(f) \setminus \{x\}\}$ of RVs x' that are neighbors of f. All messages $\mu_{x\to f}$ are initially set to 1. Then, the messages are updated until convergence. For *acyclic* factor graphs, belief propagation converges after a message has been sent and received by each node. For factor graphs with cycles, multiple iterations of sending and receiving messages can be performed (called loopy belief propagation). Conditions for convergence of the algorithm have been investigated by Weiss (2000).

Example 7. Consider the factor graph of Example 4. Here, we will not show the complete belief propagation algorithm, but only show how some of the messages are calculated. The message $\mu_{c(a)\to\alpha_1}(x_{c(a)})$ with $x_{c(a)} \in \{0, 1\}$ is updated according to

$$\mu_{c(a)\to\alpha_1}(x_{c(a)}) = \prod_{f' \in n(c(a))\setminus\alpha_1} \mu_{f'\to c(a)}(x_{c(a)}) = \mu_{\beta_1\to c(a)}(x_{c(a)})$$

The message $\mu_{\alpha_1\to c(a)}(x_{c(a)})$ is updated according to

$$\mu_{\alpha_1\to c(a)}(x_{c(a)}) = \sum_{x_d\in\{0,1\}} \alpha_1(d{=}x_d, c(a){=}x_{c(a)})\ \mu_{d\to\alpha_1}(x_d)$$

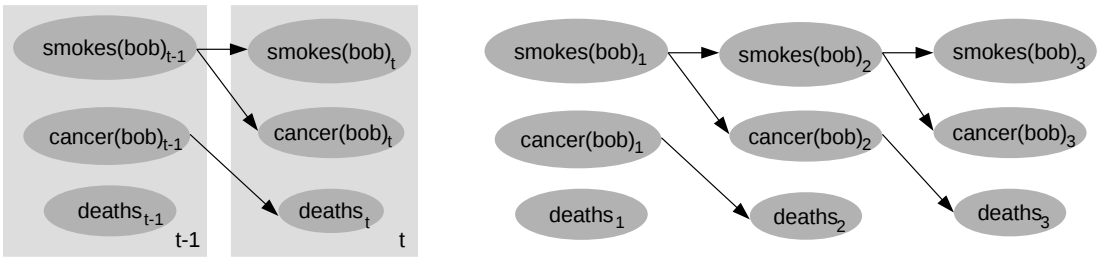
∘

2.2 Bayesian Filtering

An important subclass of probabilistic inference algorithms considers inference in cases where a distribution changes over time. They can be subsumed under the framework of *Bayesian filtering* (also called recursive Bayesian state estimation) (Särkkä, 2013). For example, consider the following extension of Example 4:

Example 8. Smoking does not cause cancer immediately, but can cause cancer in the future. Having cancer does not immediately lead to death, but can cause death in the future. Also, people who smoke tend to stay smokers, i.e. the probability of a person being a smoker depends on the person being a smoker at the previous time step. ∘

Such scenarios can be efficiently modeled by a dynamic Bayesian network (DBN). A DBN is essentially a Bayesian network with another dimension: There is a family of random variables indexed by time, and the value of each RV can depend on other RVs indexed by the same time, but also on RVs indexed by a previous time. That is, a DBN describes a stochastic process that has the Markov property. The inference goal in a DBN is to estimate the state of some (not observed, or hidden) variables, given a sequence of observations of the other variables. This task is known as *Bayesian filtering*. In the example, we might



(a) Dynamic Bayesian Network

(b) Unrolled Bayesian Network

Figure 2: Smokers domain with time dependencies (Example 8). Light grey boxes indicate that the variables share the same time index.

get information about the number of deaths for each time step, and want to estimate the number of smokers per time step.

This task can be solved by viewing the DBN as standard graphical model (known as "unrolling"), see Figure 2b. Unrolling requires a finite observation sequence, and the sequence must be completely known to construct the unrolled network. However, for applications like sensor data processing, the observations sequence is of arbitrary length, and the observation sequence is not completely present at the beginning. Instead, the inference algorithm must be able to process the observations "as they arrive", without having access to "later" observations.

Efficient algorithms for Bayesian filtering estimate the hidden state sequence $x_1, \ldots, x_t$ recursively over time, given the observation sequence $y_1, \ldots, y_t$. To do so, the DBN is factored into a *transition model* and an *observation model*. The transition model $p(x_{t+1} \mid x_t)$ describes how the hidden state at time t influences the hidden state at time $t+1$. The observation model $p(y_t \mid x_t)$ describes how the hidden state at time t influences the observation at the same time step. The inference procedure is usually decomposed into two steps: In the *prediction*, the state distribution for the next time step is calculated, based on the state distribution at the current time and the transition model, and by marginalizing over the current state:

$$p(x_{t+1} \mid y_{1:t}) = \int p(x_t \mid y_{1:t})\, p(x_{t+1} \mid x_t)\, dx_t \tag{3}$$

Afterwards, the predicted state is *updated*, based on the observation:

$$p(x_{t+1} \mid y_{1:t+1}) = \frac{p(y_{t+1} \mid x_{t+1})\, p(x_{t+1} \mid y_{1:t})}{p(y_{t+1} \mid y_{1:t})} \tag{4}$$

Two well-known algorithms that implement this framework are the Kalman filter and the Hidden Markov Model. They can only be used for linear-gaussian models or models with finite state spaces, respectively. In general, solving these equations exactly is infeasible. A popular Monte-Carlo algorithm for Bayesian filtering is the particle filter (Doucet, de Freitas, & Gordon, 2001). The idea is to approximate the distribution $p(x_{1:t} \mid y_{1:t})$ (the belief state) by a set of weighted samples. The predict and update steps are performed on these particles. That is, a new set of particles is obtained by sampling from the transition

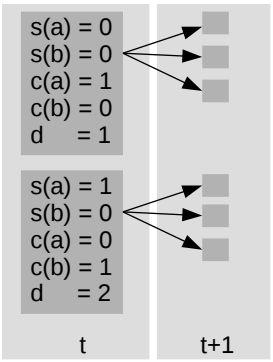
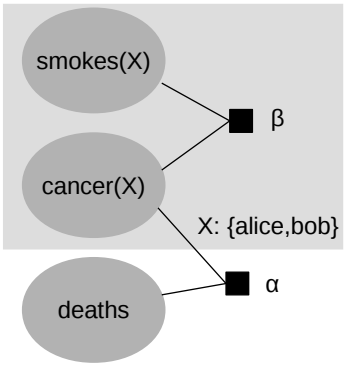


Figure 3: Predict step of the particle filter for Example 8. The example shows two particles at time t. Each particle has three successor states, leading to six particles at time $t + 1$. The update step is not shown. Light grey boxes indicate the time index.



(a) Parfactor graph.

s(X)	c(X)	β
0	0	$\beta^{(00)}$
0	1	$\beta^{(01)}$
1	0	$\beta^{(10)}$
1	1	$\beta^{(11)}$

(b) Parfactor β.

$c(X)$	d	α
0	0	$\alpha^{(00)}$
0	1	$\alpha^{(01)}$
1	0	$\alpha^{(10)}$
1	1	$\alpha^{(11)}$

(c) Parfactor α.

Figure 4: Parfactor graph for Example 4, using par-RVs and plate notation (Buntine, 1994).

distribution, conditioned on the current particles. Afterwards, each particle is updated according to the observation model. The algorithm is visualized in Figure 3.

The state space explosion problem is also evident in many dynamic models: In Example 8, the number of possible states per time step increases exponentially with the number of people.

2.3 Lifted Graphical Models

As discussed above, graphical models for situations that contain redundancies exhibit a symmetrical structure (cf. Example 4). Lifted graphical models (also known as relational graphical models) provide a more compact syntactic representation for these cases. They provide a basis for lifted inference algorithms that allow to perform inference directly on this compact syntactic representation, avoiding redundant computations. In the following, we will introduce *parfactor graphs*, one of the most common lifted graphical model formalisms.

Parfactor graphs have been introduced by Poole (2003). They are motivated by the redundancies that can occur in factor graphs. The idea of parfactor graphs is to represent the redundant factors (e.g. the factors β_1 and β_2 in Example 4) only once.

Parfactor graphs achieve this by extending factor graphs by a first-order language. Factor graphs are related to parfactor graphs in the same way that propositional logic is related to first-order logic. A *parametric random variable* (par-RV) represents a set of random variables, one for each assignment of constants the parameters. The domain of each parameter is called *population* (i.e. a set of individuals). For example, if X is a parameter with the domain $\{a, b\}$, then $s(X)$ is a par-RV, and the parameter assignments $s(a)$ and $s(b)$ both represent a random variable. We call these RVs the *groundings* of the par-RV.

A parametric factor, or *parfactor*, is a function that maps par-RV assignments to the non-negative reals. For discrete RVs, the parfactor can be represented as a table. For example, the parfactor β of Example 4 is shown in Figure 4b. Note that the factor is not indexed by the parameters of the par-RVs, i.e. the parfactor does not depend on the specific parameter assignments of the par-RVs. A parfactor represents a set of factors, one for each grounding of the par-RVs. For example, the parfactor $\beta(s(X), c(X))$ represents the two factors $\beta_1(s(a), c(a))$ and $\beta_2(s(b), c(b))$. These factors are called the *groundings* of the parfactor.

A set of par-RVs and parfactors can be represented by a *parfactor graph*. The parfactor graph for Example 4 is shown in Figure 4 (using plate notation, Buntine, 1994). A parfactor graph defines a joint probability distribution as the normalized product of all groundings of the parfactors. However, the joint distribution can also be calculated directly, without grounding all parfactors: Parfactors with the same truth assignment of variables need to be evaluated only once, raised to the power of the number of corresponding factors. For example, the probability calculated in Equation 2 can be calculated as:

$$\begin{aligned}
&P\big(s(a){=}1, s(b){=}1, c(a){=}0, c(b){=}0, d{=}0\big) \\
=&Z^{-1}\ \beta_1\big(s(a){=}1, c(a){=}0\big)\ \beta_2\big(s(b){=}1, c(b){=}0\big)\ \alpha_1\big(d{=}0, c(a){=}0\big)\ \alpha_2\big(d{=}0, c(b){=}0\big) \\
=&Z^{-1}\ \prod_{X\in\{a,b\}} \beta(s(X){=}1, c(X){=}0\big)\ \alpha\big(d{=}0, c(X){=}0\big) \qquad (5) \\
=&Z^{-1}\ \beta\big(s(X){=}1, c(X){=}0\big)^2\ \alpha\big(d{=}0, c(X){=}0\big)^2
\end{aligned}$$

Compare this with Equation 2, where the factors β_1 and β_2 are evaluated and multiplied separately. This example shows that inference operations can exploit the compact syntactic representation. Probabilistic inference algorithms that directly work on this representation are presented in Section 5.1.

Multiple other lifted graphical model formalisms have been devised. A popular formalism are *Markov logic networks* (MLNs) (Richardson & Domingos, 2006). MLNs are an extension of first-order logic with means to express uncertainty by assigning each first-order formula a weight that describes the tendency of the formula being violated. Other formalisms are based on paradigms like probabilistic logic programming (Kersting & De Raedt, 2007; Fierens, 2010), or object orientation (Koller & Pfeffer, 1997; Torti, Wuillemin, & Gonzales, 2010). A detailed description of representational formalisms is provided by Kimmig et al. (2015). In general, a main difference between these formalisms is whether they are *directed* or *undirected*. Directed models can be interpreted in terms of conditional probabil-

ities. The weights of undirected models cannot be interpreted locally, all weights together define the probabilistic model. In contrast to propositional graphical models, directed and undirected lifted models cannot be translated into each other in general. Differences of the representation formalisms are discussed by De Raedt et al. (2016).

In this review, we focus on parfactor graphs, as they are easy to understand and allow a simple description of the exemplary lifted inference algorithms shown in Section 5.1 to illustrate the basic idea of lifted inference.

2.4 Rao-Blackwellization

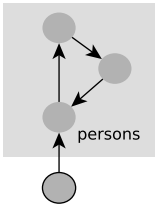
Apart from lifted graphical models, we consider a second type of state space abstraction in this review, called *Rao-Blackwellization*. Lifted graphical models exploit the fact that multiple RVs are similar, i.e. symmetries between multiple RVs. Opposed to that, Rao-Blackwellization exploits the fact that the (conditional) distribution of several (often, but not necessarily continuous) RVs follows a certain regular structure. The idea is to represent such a distribution not explicitly (e.g. as a table of all possible values or a set of samples), but *parametrically*. For example, consider a bivariate distribution $p(a, b) = p(a)\, p(b|a)$. Suppose that the conditional distribution $p(b|a)$ has some regular structure (e.g. it follows a normal distribution).

For storing and manipulating this parametric function, the function needs to have a finite representation, like the string "$\mathcal{N}(0, 1)$"[4]. The *semantics* of this *syntactic structure* is the normal distribution with mean 0 and variance 1.

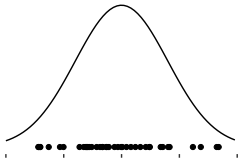
A well-known use of Rao-Blackwellization is the Rao-Blackwellized particle filter (RBPF) (Doucet, De Freitas, Murphy, & Russell, 2000). In a RBPF, the state is decomposed, such that some RVs can be represented parametrically. The transition and observation model of the RBPF have to be able to maintain this representation appropriately, i.e. it must be possible to represent the posterior distribution (the distribution after performing one predict-update step) of these variables parametrically again. This means that fewer particles are necessary to represent the belief state, because a distribution over fewer variables needs to be represented explicitly by samples. Thus, the belief state can be represented more compactly. The Kalman filter can be seen as the extreme case of a RBPF, where all variables are represented parametrically (by a normal distribution), and the transition model is linear. Note that Rao-Blackwellization is orthogonal to lifted graphical models: Lifted graphical models represent graphical models with symmetrical variables compactly by grouping them, Rao-Blackwellization represents the distribution of a single or multiple variables compactly.

Example 9. Suppose that we do not want to model whether or not a death occurred in Example 4, but the *number* n_t of deaths, i.e. n is an $\mathbb{N}$-valued RV, and we have a single factor α on all $c(X)$ RVs and the n RV. Instead of representing the factor α explicitly by a table of exponential size, we can represent the number of deaths by a binomial distribution of the number $\#_P(c(P){=}1)$ of people with cancer: $n \sim \mathit{binom}(\#_P(c(P){=}1), p_d)$. This representation is much smaller (constant size in the number of $c(X)$ RVs). However, whenever the factor α needs to be manipulated (i.e. marginalizing RVs), this either has to be done

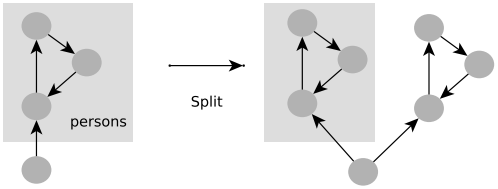
4. If we only consider normal distributions, we could also represent it by a pair of reals. However, if we allow arbitrary parametric functions (that can have different numbers of parameters), a more flexible structure like a string is required.



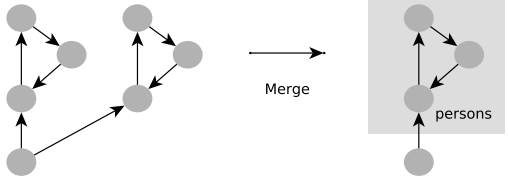
(a) Group Variables. Equivalent variables represented as a group ("lifted inference").



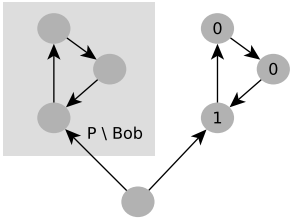
(b) Parameterization. The distribution of some state variables is represented parametrically, instead of explicitly by samples.



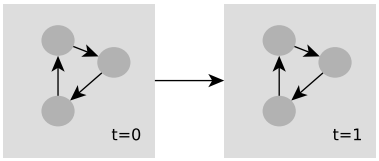
(c) Splitting. An operation that obtains a more specific representation.



(d) Merging. An operation that obtains a more abstract representation.



(e) Identification. The value of single RVs can be observed individually.



(f) Online. For each time t, a query is answered, each depending on current observations and estimate at time $t-1$.

Figure 5: Schematic depiction of properties of the algorithmic approaches.

on the parametric level (which may not be trivial), or the representation as a table has to be generated (which we try to avoid due to the exponential size of the table). ○

In general, such a parametric representation is only possible for certain distributions, more specifically distributions that can be represented syntactically by a closed form mathematical expression.

3. Properties of Inference Algorithms

In the following, we present six properties that characterize the algorithms we investigate in this review. They have been obtained by analyzing the application domains of the approaches retrieved by the systematic literature review described in Section 4. Thus, they

are a *result* of the systematic review, and one of the major contributions of this review. We chose to present them at this point in the paper because they are also used as a basis for analyzing and discussing the retrieved papers. They are depicted schematically in Figure 5.

Can the algorithm handle equivalent RVs efficiently as a group? (Group Variables) The first two properties characterize the type of abstraction that the algorithms are using. In Section 2, we presented two abstraction approaches: The first one groups multiple equivalent variables and reasons over them as a group – as for example done prominently in lifted graphical models. For example, the RVs $c(a)$ and $c(b)$, as well as the corresponding factors β_1 and β_2 in Example 4 have been grouped.

Can the algorithm handle distributions at the parametric level? (Parameterization) The second type of abstraction represents a distribution compactly by noting that the distribution follows some parametric form, and that it is sufficient to store and manipulate the parameters (which are typically far less than the enumeration of all values). The Kalman filter is a good example for this concept. The parametric distributions can also make up only some factors of the joint distribution, like in the Rao-Blackwellized particle filter, or it might be necessary to consider *mixtures* of parametric functions. In Example 4, the α factor could be represented parametrically.

Can the algorithm obtain a more specific distribution representation? (Splitting) We identify two basic operations that can be performed by an inference algorithm to modify the degree of abstraction: *Merging* and *Splitting*. Splitting is the process of obtaining a more specific (propositional) representation from an abstract representation (in logic, this operation is known as *grounding*). Splitting operations are necessary for incorporating observations: Evidence about an RV makes this RV distinct from other RVs that are part of the same par-RV, and thus requires a *split* of this par-RV and the corresponding parfactors. It can also be necessary to ensure the applicability of certain inference operations (e.g. the inversion elimination step in first-order variable elimination requires certain conditions that are ensured by splitting). In Example 4, if we obtain the information that Bob smokes, but we have no information whether Alice smokes, the corresponding par-RV $s(X)$ cannot be maintained any longer and has to be split into separate RVs $s(a)$ and $s(b)$.

Can the algorithm obtain a more abstract distribution representation? (Merging) Merging (or lifting) is the reverse process to splitting: Obtaining a more abstract or aggregated representation, by grouping equivalent variables. For example, grouping the RVs $s(a)$ and $s(b)$ into the par-RV $s(X)$ is a merging operation. Merging is necessary in all domains where either the problem is given in a propositional form, or domains where the problem degenerates over time by repeated splitting operations. Splitting and merging only change the *representation* of a distribution, they do not change the distribution itself (or at least, when approximate methods are used, they try to change it as little as possible).

Can the algorithm handle information about individuals? (Identification) In lifted models, a common problem is how information about single individuals (i.e. single RVs) is handled. For example, suppose that in the parfactor graph given in Figure 4, we are provided with the evidence that Alice has cancer. In this case, the evidence can be incorporated into the model by splitting the representation, and handling Alice differently from

the rest of the population. Not all algorithms handle identifying information by splitting. For example, when the model is given in propositional form and merging operations are applied to it, the evidence can be considered there, leaving Alice as a special case. Some methods do not allow to process evidence about individuals at all, like Multiset Rewriting Systems.

Can the algorithm perform inference in dynamic domains? (Online) This property describes the difference between probabilistic inference and Bayesian filtering. Probabilistic inference answers a single query (i.e. it estimates the state of hidden variables) for a *single* point in time, given evidence. Bayesian Filtering answers a *sequence* of queries, one for each time step. Each query depends on the current observation, and the distribution of the hidden variables of the previous point in time. In general, the observation sequence is not known in advance, but more observations are obtained as time passes. As explained in Section 2.2, such problems cannot be solved efficiently with non-sequential inference algorithms. Instead, the inference algorithms require a property that we call *online inference*: Calculating the posterior probability in a sequential fashion, with a time complexity of each step that does not depend on the total sequence length. This way, observation sequences of indeterminate length can be processed by the algorithm.

The properties describe the application domain of the approaches: Two approaches that are similar regarding these properties can (in principle) be applied to the same class of problems, while exploiting *some* symmetry of the domain. We want to point out that only two of the properties describe *state space abstraction* methods – the others describe transformations between abstract and explicit representation, and further properties that are required by some domains. They are chosen in such a way that they are meaningful for all of the approaches considered in this review[5] – but for each resulting class of approaches, we incorporate a discussion of group-specific properties, whenever necessary. Note that the properties do not describe complexity classes – in contrast to the classification proposed by Jaeger and Van den Broeck (2012) (see Appendix A), which is, however, only meaningful for a subset of all approaches, namely lifted inference algorithms. That is, two approaches that fall in the same group can still be different regarding the subproblems for which they are *tractable*.

4. Systematic Literature Review

In the following, we describe the search and evaluation methods used in this systematic review. As systematic reviews are not very common in computer science, this section starts by briefly introducing the systematic review methodology. Afterwards, we describe how each of the steps has been realized for this review.

A systematic literature review aims at finding all relevant work addressing a specific research problem by performing a reproducible and objective process. Compared to an unstructured review, a systematic review gives a broader, unbiased view of the topic. Un-

5. For example, Lifted Inference algorithms can be distinguished based on their algorithmic ideas (search-based, graph manipulation-based, MCMC-based etc.), the representation formalism, etc., but such a distinction is (1) not meaningful for some approaches, e.g. for Multiset Rewriting Systems, and (2) does not characterize the problem domain, as intended by us.

structured reviews have a higher chance to miss out contributions, either because they have not been found or because of *narrative distortion*, the observations that the author of a review is more likely to include a paper if it supports the argumentation structure of the review. A systematic review consists of the following steps (Kitchenham, 2004): (1) define the research question, (2) define the search procedure, (3) identification of research items (papers), (4) paper selection, (5) paper analysis. The PRISMA statement (Moher et al., 2009) is an established guideline that describes which items should be reported in a systematic review. In this review, we try to follow this guideline whenever possible. However, the PRISMA statement is directed towards quantitative analysis of medical research, whereas the present review is more concerned with qualitative aspects, namely assessing solution strategies to a specific problem. Therefore, some items could not be reported.

The research question (of the systematic review, not to be confused with the research question of the analyzed papers) typically consists of the following parts: (1) What research exist that solve problem P? (2) How are the solutions of P related to each other? (3) What further research topics arise from the existing research? After the research question is made clear, the search procedure to answer this question is defined. This includes the definition of search terms as well as the publication databases that are used for the literature search. A common strategy to identify search terms is to use a set of pilot papers that are known to be relevant, based on prior knowledge of the field. These pilot papers then guide the definition of the search terms, by making sure that all of them are retrieved.

Based on the search terms, the selected publication databases are searched and a list of initial papers is retrieved. These papers are then examined to assess their relevance to the research question, based on predefined *inclusion* and *exclusion criteria*. This step is performed by only considering the title, abstract and keywords of each paper. Afterwards, the full-text of the remaining papers is retrieved and their relevance regarding the inclusion and exclusion criteria is examined once again. The remaining papers are called *primary papers*. The primary papers are then analyzed with respect to the research question. This includes finding the underlying structure and relationship of the approaches and identifying possible research gaps. In the following, we describe how each of the steps has been implemented for this review.

4.1 Research Question

As described in the introduction, this review aims at giving an overview over solutions to the state space explosion problem from different research fields. More specifically, we are concerned with probabilistic inference algorithms that exploit state space abstractions. Our goal is to identify the common underlying structure of the approaches: What are common properties of the algorithms, and how does this reflect their capabilities, i.e. their applicability to different problem instances.

More formally, these questions can be stated as follows:

Q1 What methods exist to overcome the state space explosion problem in probabilistic inference?

Q2 What types of problems can different methods be applied to, and how is this reflected by the properties of the methods?

First term set
lifted
first order
higher order
symmetry
permutation
multiset

Second term set
bayesian inference
probabilistic inference
probabilistic reasoning
graphical model
bayesian network
state space model
recursive bayesian estimation
bayesian filtering
particle filter
hidden markov model
probabilistic multiset rewriting
multi-agent
multi-target
multi-object
activity recognition
plan recognition

Table 1: Search terms used to construct search query.

Q3 How are these methods related to each other, i.e. are similar concepts used in multiple approaches?

Q4 Which topics for future research can be derived?

4.2 Search Procedure

For the literature search, we used the publication databases ScienceDirect, IEEE Xplore, ACM digital library, and Scopus. These databases were chosen based on their relevance for computer science publications, and the possibility to perform a search only on title, abstract and keywords of a publication[6]. Our definition of search terms has been based on 10 pilot papers (Barbuti et al., 2011; de Salvo Braz et al., 2005; Gogate & Domingos, 2016; Huang et al., 2009b; Kersting, 2012; Kwiatkowska et al., 2006; Milch et al., 2008; Niepert, 2012; Poole, 2003; Singla & Domingos, 2008) that were the result of an explorative investigation of the literature. The search terms were defined to make sure that all of these papers have been retrieved. However, they were formulated in a general way and do not aim at specific papers or methods, to retrieve as many papers as possible that are relevant for the scope of this review. The search terms have been iteratively refined during the search process, by adding search terms to the set whenever we discovered literature that we considered relevant, and the field has not been fully covered by the current terms. The resulting terms are shown in Table 1.

The first term set describes possible state space abstractions, the second term set describes the domain where the abstractions are applied, or the research area where such

6. Another common publication database, SpringerLink, was not used because it only allowed full text searches.

abstractions are used. We constructed the query by connecting all terms in a set with logical OR and both sets with logical AND. This query describes all papers where at least one of the terms of the first set and at least one of the elements of the second set occurs. The search has been performed on the title, keywords and abstract of the publications. This way, the number of results stayed manageable, and we still retrieved all papers where any of the terms occurred prominently (i.e. that might be relevant).

4.3 Paper Selection

The search results have been assessed based on the following inclusion criteria.

I1 The paper is written in English.

I2 The paper is peer-reviewed.

I3 The full text of the paper is available via IEEExplore, the ACM Digital Library, SpringerLink, ScienceDirect, or other sources like the author's website.

I4 The paper includes a novel algorithmic contribution.

I5 The paper is considering a probabilistic model.

I6 The paper presents an inference algorithm for the probabilistic model.

I7 The paper presents an abstract representation of the state space or a method to reduce the state space.

I8 The inference algorithm exploits the state space abstraction.

Criteria **I1**-**I3** make sure that the analysis of the papers is feasible for us. This review focuses on technical approaches to handle the state space explosion problem. Therefore, **I4** ensures that application and review papers are excluded. Criterion **I5** implies that only approaches that model a probability distribution have been considered. Reduction methods in *deterministic* settings, like first-order resolution, or state space abstraction in search problems (Holte & Fan, 2015), were excluded by this criterion: Although they might contain interesting ideas on how a state space can be abstractly represented, they cannot be applied to probabilistic models in a straightforward manner. For **I6**, we defined probabilistic inference as *calculating a posterior distribution, given a prior distribution.* This definition also includes inference algorithms for dynamic domains, that might perform this step repeatedly. Criteria **I7** and **I8** ensure that only approaches that exploit a state space reduction method were included. Specifically, approaches that perform inference by grounding the abstract representation were not included, for example approaches known as knowledge-based model construction. The rationale is that this review is focused on inference algorithms that actually exploit the lifted representation, i.e. directly reason in the lifted domain.

Paper inclusion/exclusion used a three-step process. At first, only the title, abstract, and keywords of each publication have been examined. The full-text of the remaining papers has been examined in more detail. By examining the references in the remaining papers, we identified additional relevant papers (see flow diagram in Figure 6).

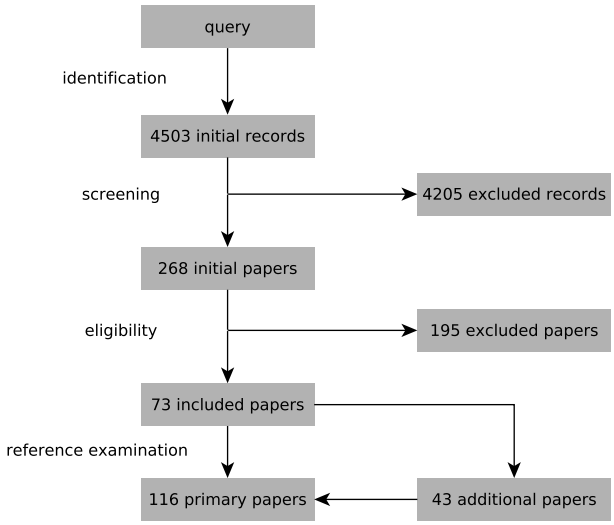


Figure 6: Flow Diagram of paper selection, oriented on PRISMA statement (Moher et al., 2009).

4.4 Analysis Procedure

We analyzed the remaining papers in order to answer research questions **Q1** – **Q4**. The analysis is based on the *properties of inference algorithms* defined in Section 3, i.e. these properties have been assessed for all approaches described in the retrieved papers. Afterwards, we performed a clustering of the approaches based on their manifestation of the properties, i.e. all approaches having the same manifestation of the properties form a cluster (or group). These groups thus define all approaches that behave similar from an application point of view, i.e. all approaches from the same group can be applied in the same problem domain (although different subclasses of the domain may be solvable efficiently).

4.5 Results

This section gives quantitative results about the retrieved papers. From the 4503 initial records that have been retrieved by the database search, 4235 have been excluded by only examining their title, keywords, and abstract. The relevance of the remaining 268 papers (regarding the inclusion criteria) has been examined based on the full-text. 195 of those papers have been excluded, based on the inclusion criteria as shown in Table 2. When multiple reasons apply to one paper, it is grouped under the the first reason, based on the order of the inclusion criteria. The high number of papers excluded because of **I6** shows that the query terms have been chosen very broadly, such that also a great number of papers that are not concerned with probabilistic inference have been retrieved. Most of the papers excluded because of **I8** are concerned with *knowledge-based model construction*, i.e. propositional inference in lifted models, a research field much older than lifted inference. In Appendix B, it is further discussed why specific approaches that might seem relevant have not been included.

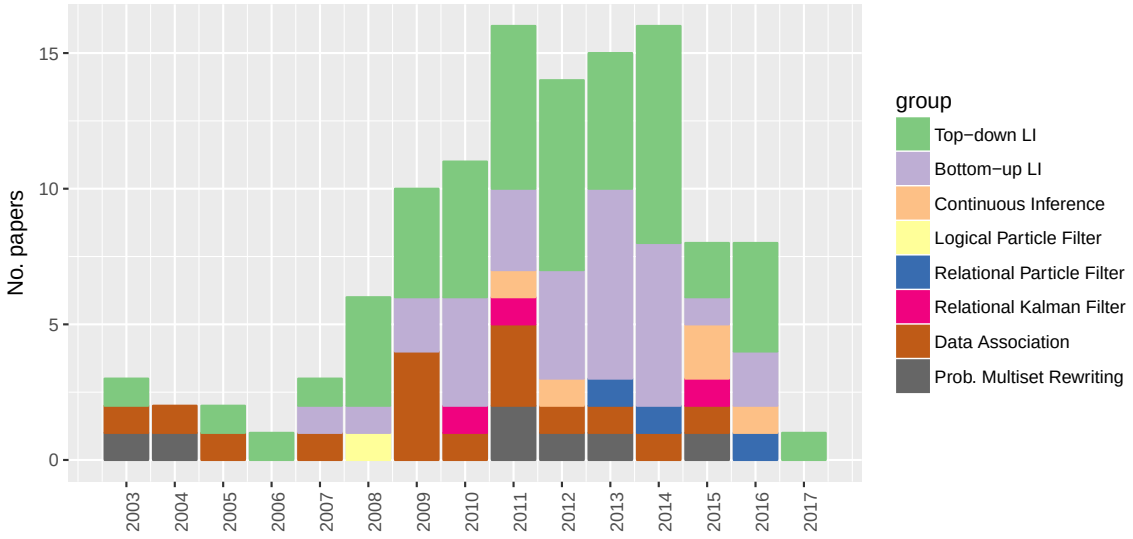


Figure 7: Number of examined papers per year. The papers have been retrieved from January to February 2017. The groups are based on the analysis and clustering of approaches, as described in the text.

Crit.	#	Explanation
I1	3	Paper not written in English
I2	0	Full-text not available
I3	9	Paper not peer reviewed
I4	31	Paper does not contain a novel algorithmic contribution (e.g. application and review papers)
I5	11	Model is not probabilistic (e.g. inference in first-order logic)
I6	77	No inference algorithm for probabilistic model (e.g. because paper presents an algorithm for learning the model structure, or something completely different, like planning or model checking)
I7	17	No lifted representation of probabilistic model (e.g. propositional models)
I8	46	Inference algorithm does not exploit abstract representation (e.g. it relies on a complete grounding)

Table 2: Reasons for excluding 195 of the 268 papers that remained after examining title, keywords and abstract of the 4503 initial records.

The remaining 73 papers were considered relevant and included into this review. The references of these papers were examined, which lead to the identification of another 43 relevant papers. Thus, 116 papers have been included in this review in total. This corresponds to a precision of $73/4503 = 1.6\%$ and a recall of $73/116 = 62.9\%$ of the initial query. These low values point to the fact that the terminology in the field is not very consistent.

Online	Identification	Group Variables	Parametrization	Splitting	Merging	No. Papers	Name	Section
□	■	■	□	■	-	50	LI Top-down	6.1.1
□	■	■	□	-	■	31	LI Bottom-up	6.1.2
□	■	□	■	■	■	5	Continuous Inference	6.2
■	□	■	□	-	-	7	Multiset Rewriting	6.3
■	■	■	□	■	□	1	Logical Particle Filter	6.4
■	■	□	■	■	□	3	Relational Particle Filter	6.5
■	■	■	■	■	■	3	Relational Kalman Filter	6.6
■	■	□	■	-	-	16	Data Association	6.7

Table 3: Groups of inference approaches, based on the properties defined in Section 3. ■: has property, □: does not have property, -: property not necessary/not meaningful.

The properties of the approaches presented in these 116 papers have been evaluated, as described in Section 4.4 (thus answering **Q1**). We then clustered the approaches, as described in Section 4.4: All approaches having the same manifestation of the properties have been put into the same group. With this process, we found eight distinct groups. We assigned names to the groups that seemed appropriate to us. The groups are shown in Table 3. The complete list of all papers per group is shown in Appendix C. We want to emphasize that the groups have not been predefined, but they are a result of the individual analysis of each paper.

As can be seen from Table 3, the "lifted inference" groups contain by far the most papers. This shows that lifted inference is a very active research area. The other groups contain fewer papers. One reason may be that they belong to a larger research area (for example, there are numerous papers on data association in general), but only a small subset of the approaches employ state space abstraction.

Figure 7 shows the chronological development of the research area. Although the first lifted inference paper was published in 2003, the majority of lifted inference papers has been published after 2008. The drop in the total number of included papers after 2014 may be due to the fact that not all papers form 2015 and 2016 are properly indexed at the used publication databases at the time of retrieving the papers (January – February 2017).

5. Categories of Inference Algorithms

As discussed in Section 4.5, we defined groups or classes of approaches that consist of all approaches that are similar regarding the six properties defined in Section 4.4 (shown in Tables 3 and Appendix C). In the following, we briefly describe the common algorithmic ideas that are shared by all approaches in the same group.

5.1 Lifted Inference

Lifted inference algorithms are concerned with probabilistic inference in lifted graphical models (Section 2.3). They aim at performing the inference directly in the first-order domain, without grounding the lifted graphical model, whenever possible. By maintaining the lifted representation, they can exploit the symmetries and redundancies that are inherent to these representations. More specifically, lifted inference algorithm can be seen as exploiting *exchangeability* in the model (Niepert & Van den Broeck, 2014): They exploit the fact that in lifted graphical models, it is not necessary to know the *specific* RVs having a certain value, but only the *number* of RVs having each value. In general, lifted inference algorithms can be viewed as performing the following steps: (1) Decompose the inference problem into similar, independent subproblems, (2) solve one representative instance, (3) count the number of instances (instead of generating all instances) (Taghipour, Fierens, Van Den Broeck, Davis, & Blockeel, 2013c).

How these steps are implemented is specific to the different lifted inference algorithms. As a high-level distinction, we distinguish between top-down and bottom-up lifted inference, following Kersting (2012). The difference of these approaches is the input they receive: Top-down lifted inference algorithms start with a lifted graphical model, while bottom-up algorithms receive a propositional model as input (thus, they are different in step (1) – the generation of subproblems). From the algorithmic viewpoint, this distinction is not always very precise, as it is just a matter of preprocessing: For several algorithms, both top-down and bottom-up versions exist – for example, lifted belief propagation has top-down (Singla & Domingos, 2008) and bottom-up (Kersting, Ahmadi, & Natarajan, 2009) variants. However, as this review is explicitly concerned with the *problem class* each approach can process, we still consider bottom-up/top-down a meaningful distinction – it is also directly reflected by the *properties* (Section 3) of the algorithms: Top-down algorithms apply splitting operations, while bottom-up algorithms need to perform merging operations on the propositional model (but never need to apply splitting operations)[7]. Top-down algorithms, on the other hand, never apply merging operations (i.e. they never explicitly search exchangeable RVs and group them).

We want to point out that lifted inference problems and algorithms can be structured further, as proposed by Jaeger and Van den Broeck (2012): Broadly speaking, the idea is to classify lifted probabilistic models by the "complexity" of their structure (in terms of numbers of parameters of par-RVs and parfactors). For some of the resulting classes, it can be shown that inference can always be performed in time that is polynomial in the domain size of the model, while in general, no such guarantees can be given. In Appendix A, we give an overview over these problem classes. However, as an in-depth discussion of lifted inference is not the focus of this review, we do not discuss this classification in detail here. From the high-level point of view of this review, all lifted inference algorithms are concerned with a similar problem: Efficient inference in graphical models containing symmetries. For a more in-depth discussion, we refer to the review papers of Kersting (2012) and Kimmig et al. (2015), as well as the books by De Raedt et al. (2016) and Getoor and Taskar (2007).

7. Search-based algorithms are also considered top-down. They branch on the value of the (par-)RVs, resulting in a simpler inference problem in each branch. We consider this branching a form of splitting.

In the following, we explain the general idea of some prominent lifted inference algorithms (first-order variable elimination, lifted recursive conditioning, lifted belief propagation).

5.1.1 Top-Down Lifted Inference

First-order Variable Elimination Poole (2003) proposed the first ideas related to lifted inference, in an algorithm known as first-order variable elimination (FOVE). The idea is to perform variable elimination directly on a parfactor graph, eliminating entire par-RVs in one step, instead of single RVs.

Example 10. Consider the graphical model of Example 4 and the query $P\big(s(X), d{=}1\big)$. Remember that inference in the propositional model (with $X = \{a, b\}$) requires two elimination steps, the elimination of $c(a)$ and $c(b)$ (Example 5). In the parfactor graph (Figure 4), we can *in principle* directly eliminate the par-RV $c(X)$ by multiplying the parfactors β and α and marginalizing $c(X)$ to get a factor

$$f(s(X), d) = \sum_v \alpha\big(c(X){=}v, d\big)\ \beta\big(s(X), c(X){=}v\big)$$

which can be represented by the table

s(X)	d	f
0	0	$\beta^{(00)}\,\alpha^{(00)} + \beta^{(01)}\,\alpha^{(10)}$
0	1	$\beta^{(00)}\,\alpha^{(01)} + \beta^{(01)}\,\alpha^{(11)}$
⋮	⋮	⋮

This factor directly leads to the query solution $P\big(s(X), d{=}1\big) = f\big(s(X), d{=}1\big)$. ∘

The elimination step performed for eliminating $c(X)$ in the example is called *inversion elimination*. Not all cases can be handled this way: For example, consider the case of eliminating d: In the ground factor graph, eliminating d means we need to multiply all α_i factors, resulting in a factor of all $c(X)$, i.e. a factor that has exponential size with respect to the domain. In general, inversion elimination can only be applied when the parameters that appear in the par-RV to be eliminated are the same as the parameters in each parfactor depending on this par-RV. Thus, for eliminating d, inversion elimination cannot be applied, and FOVE as proposed by Poole (2003) needs to ground $c(X)$ and create the the exponentially large factor.

However, the RV d (whether or not a death occurred last year) might only depend on the *number* of people having cancer, not their specific identities. Thus, it is sufficient that the resulting factor considers the number of instances of $c(X)$ that are true. This was first realized by de Salvo Braz et al. (2005), who presented an elimination operator that can handle this case. Later, Milch et al. (2008) proposed an explicit representation of such factors, called *counting formulae*, that have later been generalized by Taghipour et al. (2014). Additional elimination rules that make FOVE applicable to more cases without grounding are provided by Apsel and Brafman (2011), and Taghipour et al. (2014, 2013b). Using these rules, the class FO^2 (inference problems containing at most two parameters

per parfactor, see Appendix A for more details) can always be solved in polynomial time in the parameter domain size. The works of Taghipour et al. (2012, 2013a) allow for more general constraints in the parfactors.

Lifted Recursive Conditioning Approaches based on variable elimination have the problem that they need to represent the intermediate results of the elimination operations, that can become increasingly complex during inference. Recently, *search-based* lifted inference algorithms have emerged, that do not manipulate the representation of the parfactors directly, but branch on the values of par-RVs and combine the results of each branch appropriately. The convenient property of these algorithms is that the intermediate results (partially instantiated lifted graphical models) become simpler with each operation, instead of more complex.

For example, lifted recursive conditioning (Poole, Bacchus, & Kisynski, 2011) works similar to recursive conditioning (see Section 2.1.2), but branches on the values of par-RVs instead of (propositional) RVs. The algorithm exploits a similar idea as counting elimination: There are cases where it is sufficient to branch on the *number* of RVs having each possible value, instead of all assignments of the RVs. An extension of lifted recursive conditioning (Kazemi, Kimmig, Van den Broeck, & Poole, 2016) is able to solve all problems in the class S^2RU in polynomial time – which is currently one of the largest classes where tractable inference can be guaranteed (see Appendix A for details).

Example 11. Consider the graphical model of Example 4 and the query $P\big(s(X), d{=}1\big)$. At the beginning, only $d{=}1$ is instantiated and the algorithm needs to branch. Instead of branching on the values of a single RV, it creates one branch for each *histogram* of possible values of the instances of a par-RV. In this example, the algorithm chooses $c(X)$ to branch, leading to three recursive calls of the algorithm, where 0, 1 or 2 instances of $c(X)$ are true, respectively. In each branch, the factor α can be evaluated. For example, in the branch where 2 instances of $c(X)$ are true, it is evaluated as $\alpha\big(d{=}1, c(X){=}1\big)^2$. Afterwards, a similar branch is performed on $s(X)$. Note that the result of each branch needs to be multiplied by $\binom{n}{i}$ (where i is the number of true instance in the branch, and n is the population size), as this is the number of equivalent ground assignments represented by this branch. Compared to recursive conditioning, where we need to branch on each (ground) RV, fewer branches need to be performed. ∘

Several other search-based algorithms have been devised. The approaches of Van Den Broeck et al. (2011), and Gogate and Domingos (2016) transform the problem into a weighted model counting problem on a first-order logical theory (WFOMC): Given a first-order logical theory T and positive and negative weight functions w and $\overline{w}$ for each predicate. WFOMC is the problem of computing

$$\sum_{I \models T} \prod_{a \in I} w(\mathrm{Pred}(a)) \prod_{a \in \mathrm{HB}(T) \setminus I} \overline{w}(\mathrm{Pred}(a)) \tag{6}$$

where I is a model of T, $\mathrm{HB}(T)$ is the Herbrand base of T and Pred is the predicate of an atom a. Note that the weighted theory defined above is different form an MLN, where a weight is assigned to each formula, not to each predicate. Given a parfactor graph, one can construct a weighted theory such that the weighted model count is the probability of

some evidence in the factor graph. The basic idea is that each model relates to a value assignment of the RVs that is consistent with the evidence. WFOMC can be solved directly by a search-based algorithm (Gogate & Domingos, 2016), or by compilation into a first-order arithmetic circuit (Van Den Broeck et al., 2011).

Jha et al. (2010) propose a rewriting-rule based inference algorithm. These rules take an MLN and express it as a combination of multiple simpler MLNs until the MLNs are trivial such that the solution can be computed directly.

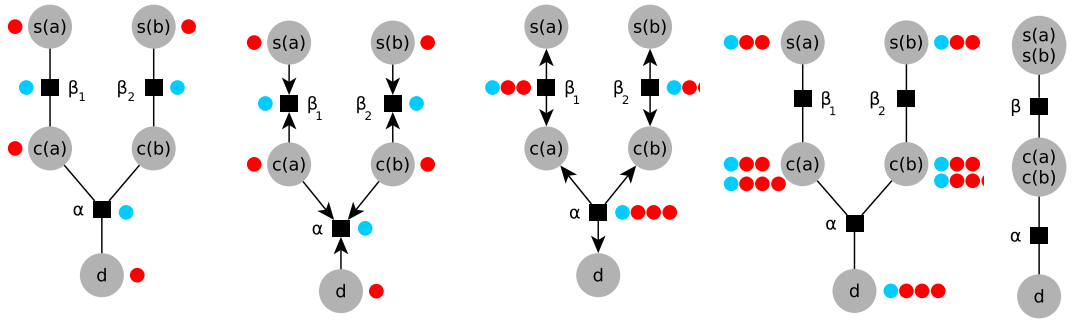
Probabilistic Databases Ideas related to lifted inference arose independently in the probabilistic database community. Probabilistic databases are relational databases where each tuple is a boolean random variable, and database queries output a probability distribution of possible answers (instead of a single answer, as in conventional relational databases). Thus, query evaluation in a probabilistic database is a probabilistic inference task. More details on probabilistic databases and query evaluation are provided in the book by Suciu, Olteanu, Ré, and Koch (2011).

Answering queries in probabilistic databases corresponds to an *asymmetric* weighted model counting task, where weights of predicates can vary, depending on the domain constant (as compared to symmetrical WFOMC defined above, where each predicate always has the same weight). Still, symmetries can be present, allowing to use methods closely related to (bottom-up) lifted inference algorithms (Sen, Deshpande, & Getoor, 2008). Dalvi and Suciu (2007) present an algorithm that rewrites a probabilistic database query in terms of combinations of simpler queries, until trivial queries can be answered directly. This approach is conceptually similar to search-based lifted inference algorithms like lifted recursive conditioning (Poole et al., 2011). Typically, probabilistic databases assume that tuples are independent, which can make inference much easier in certain cases. Jha and Suciu (2012) show how correlations can be modeled in tuple-independent databases, allowing to use lifted methods devised for tuple-independent databases (e.g., Dalvi & Suciu, 2007) in a more general setting. Dylla, Miliaraki, and Theobald (2013) devise an algorithm for finding the k most probable query results according to their marginal probabilities, without the need to first materialize all answer candidates. The symmetrical case of WFOMC has also been considered by the probabilistic database community, leading to new insights on domain-liftable inference problem classes (Beame, Van den Broeck, Gribkoff, & Suciu, 2015) – outlined in Appendix A.

5.1.2 Bottom-Up Lifted Inference

As opposed to top-down lifted inference algorithms, bottom-up approaches take a propositional model and perform merging operations to obtain a first-order structure that can be exploited. Thus, bottom-up approaches are potentially applicable to a larger class of problems, as they do not require the model to be in lifted form (or to even contain exact symmetries, instead they can approximate the model by a symmetric one). However, performing merging operations is an additional overhead: The propositional model can be very large, and merging requires at least linear time in the propositional model size.

A well-known bottom-up lifted inference algorithm is lifted belief propagation proposed by Kersting et al. (2009). The idea is to perform belief propagation (BP) on a factor graph where each node represents a set of nodes that would send and receive the same messages



(a) Initially, all RV and factor nodes have same color.

(b) RV nodes send color to factor nodes.

(c) Factor nodes send stacked colors to RV nodes.

(d) Colors are stacked at RV nodes.

(e) RV nodes with same colors are grouped.

Figure 8: From left to right, the steps of lifted BP factor graph compression (adapted from Kersting et al., 2009).

in standard BP. This lifted factor graph is obtained by simulating BP and keeping track of which nodes send and receive the same messages. In this simulation, each node sends its color (a signature) instead of the actual message. Initially, all RV and factor nodes have the same color signature. The colors a node receives extend the current color of the node. This color signature is sent in consecutive messages. After one iteration (all nodes have sent and received a message), nodes with the same color signature are grouped for the next iteration.

Example 12. Figure 8 shows the steps of simulating BP and compressing the factor graph of Example 4. The nodes $s(a)$ and $s(b)$, $c(a)$ and $c(b)$ as well as β_1 and β_2 have the same color signature after one iteration. Thus, they are grouped together in the factor graph. Afterwards, a modified BP algorithm is performed on the compressed factor graph. This algorithm needs to consider the actual number of messages sent and received by the grouped nodes. For example, a message sent from node $c(a), c(b)$ to α actually represents two identical messages. ○

For cases where it is necessary to answer multiple queries on the same graphical model with only slight changes in the evidence, it is not necessary to re-construct the lifted network from scratch each time. Instead, Nath and Domingos (2010b), and Ahmadi, Kersting, and Hadiji (2010) showed how the lifted network can be re-used, which is not trivial, as the structure of the lifted network depends on the evidence. These methods can be used to realize lifted variants of the Kalman filter and PageRank algorithm (Ahmadi, Mladenov, Kersting, & Sanner, 2011), as well as lifted linear program solvers (Mladenov, Ahmadi, & Kersting, 2012).

Other bottom-up algorithms find symmetries in the graphical model by examining graph automorphisms of the graphical model. These automorphisms can be used for lifted variational inference (Bui, Huynh, & Riedel, 2013) and lifted sampling-based inference (Niepert, 2012; Venugopal & Gogate, 2012). In general, such approximate algorithms (based on sam-

pling or belief propagation) can be feasible for very large and complex models, where exact inference (like variable elimination or recursive conditioning) is infeasible.

Another interesting property of bottom-up algorithms is that they can potentially also be applied to models that are not exactly symmetric, but exhibit *approximate* symmetries. This can happen when evidence about individuals is observed, and is a main issue for exact lifted inference algorithms. Methods have been devised that that approximate the model by a symmetric one, and then perform lifted inference in the symmetric model (Singla, Nath, & Domingos, 2014; Venugopal & Gogate, 2014a; Van Den Broeck & Niepert, 2015). Combining this with approximate inference algorithms can lead to even more efficient inference.

5.2 Continuous Inference

Most research on probabilistic inference is concerned with discrete RVs, although many practical problems require modeling continuous variables. For inference in graphical models containing continuous RVs, algorithms for discrete models cannot be used directly, as they typically rely on enumerating all values of the RV. Instead, it is necessary to describe the functional form of the factors containing continuous RVs and manipulate them analytically (this is an instance of the parameterization property introduced in Section 3). Typical operations that need to be handled are marginalization (integration) and multiplication of such continuous factors. In general, such operations can be difficult or impossible. However, recent research has focused on *piecewise polynomial* functions for describing factors, which can be manipulated efficiently. For example, in the approach by Sanner and Abbasnejad (2012), factors are represented as piecewise polynomial functions that are noted as case statements, as illustrated by the following example.

Example 13. The position of an object is observed by a noisy sensor observation. Both the position (x) and the observation (o) are continuous RVs. The sensor can either fail, or work properly (modeled as a binary RV b). In the former case, the observation density is uniform in the interval $[0, 10]$ (i.e. the density is constant $1/10$ in this interval, such that it integrates to one). In the latter case, the conditional observation density is a quadratic function, centered at the real position and truncated at a distance of one from the true position[8]. This continuous distribution can be represented by a case statement as follows:

$$p(o|x,b) = \begin{cases} -(o-x)^2 + 5/6 & b = 0 \wedge x - 1 \leq o \leq x + 1 \\ 1/10 & b = 1 \wedge 0 \leq o \leq 10 \\ 0 & \text{otherwise} \end{cases}$$

∘

In the approach by Sanner and Abbasnejad (2012), inference is defined in terms of variable elimination. When a variable is marginalized from a factor (a piecewise polynomial function), the factor is integrated on the variable to be eliminated. This integration can be calculated symbolically. The resulting factor can be more complex than the original factor (i.e. it can contain more cases), but it is always again a piecewise polynomial function and thus can be represented by case statements. These operation thus result in a more complex,

8. The added constant $5/6$ ensures that the density is always positive and integrates to one.

explicit representation of the distribution (more cases need to be distinguished explicitly) – in the context of this review, this is a splitting operation.

We can also think of an operation similar to merging for continuous inference methods: Given a distribution as case statement, a merging operation finds an equivalent case statement with fewer cases. For example, consider the case statement

$$p(a) = \begin{cases} -a & -1 \leq a \leq 0 \\ a & 0 < a \leq 1 \\ 0 & \text{otherwise} \end{cases}$$

where the first two cases can be merged into the single case $|a|$, when $-1 \leq a \leq 1$. Such operations are implicitly performed in the approach by Sanner and Abbasnejad (2012), who represent case statements as some variant of algebraic decision diagrams (ADDs) – this way, it is ensured that the case statements can be represented sufficiently compact.

Inference algorithms in continuous or hybrid models that rely on polynomial approximations have also been devised in the context of belief propagation (Shenoy & West, 2011), and weighted model counting (Belle, Passerini, & Van den Broeck, 2015a).

5.3 Probabilistic Multiset Rewriting Systems

Multiset rewriting systems (MRSs) (Calude, Paun, Rozenberg, & Salomaa, 2001) are a formalism to model dynamic systems where the state can be described as a *multiset* of entities (i.e. they perform online inference). The state transitions are defined in terms of rewriting rules having preconditions (a multiset of entities that are consumed by the reaction) and effects (a multiset of entities that are created by the reaction). They are for instance used to model biochemical reactions (Barbuti et al., 2011), population dynamics in ecological studies (Pescini, Besozzi, Mauri, & Zandron, 2006) or network protocols (Cervesato, Durgin, Lincoln, Mitchell, & Scedrov, 1999).

Example 14. A system consists of prey (x) and predators (y). Prey can reproduce, and predators can eat prey. In this simple model, eating a prey results in the death of the prey and the birth of a predator. This system can be modeled as a MRS with the two rewriting rules $r(x) \to 2x$ and $e(x, y) \to 2y$. ∘

Stochastic MRSs (Bistarelli, Cervesato, Lenzini, Marangoni, & Martinelli, 2003) assign weights to each rule, thereby specifying the probability of selecting this rule. Typically, MRSs are used for simulation studies: At each step, one of the rules is sampled according to their probabilities, leading to a sequence of multiset states.

Example 15. Consider the multiset state[9] consisting of two predators and two preys $s = \llbracket 2x, 2y \rrbracket$ and the rules $r(x) \to 2x$ and $e(x, y) \to 2y$ given in Example 14. The rules have the weight $w_r = 2$ and $w_e = 1$. Thus, their probability is $p(r) = 2/3$ and $p(e) = 1/3$ and the successor states $s_r = \llbracket 3x, 2y \rrbracket$ and $s_e = \llbracket 1x, 3y \rrbracket$ have the same probabilities. ∘

9. We use $\llbracket \cdot \rrbracket$ to denote multisets.

A popular formalism relying on MRS semantics are P Systems (Paun, 2012), where states can have a hierarchical structure (i.e. multisets can contain other multisets, and rewriting rules can also apply to the components of these inner multisets). Instead of executing one action per time step, they define the state transitions by *parallel* rule applications: At each step, a *maximal* multiset of rules (i.e. such that no more rules are applicable at the same time step, given the multiset state) is executed.

Example 16. Consider the same situation as in Example 15, but a parallel transition semantics. The following maximal rules are applicable: $c_1 = [\![2r(x)]\!]$, $c_2 = [\![1r(x), 1e(x,y)]\!]$ and $c_3 = [\![2e(x,y)]\!]$. To compute the weight of each parallel rule, we multiply the weights of the individual rules and the number of ways that entities in the state can be assigned to the preconditions of the actions. Thus, the weights of the parallel actions are $w_1 = w_r^2 * 2 = 8$, $w_2 = w_r * w_e * 2 * 2 = 8$ and $w_3 = w_e^2 * 2 = 2$. Finally, the probabilities are obtained by normalizing the weights: $p(c_1) = p(c_2) = 4/9$, $p(c_3) = 1/9$. ∘

Computing the distribution of maximally parallel rules is a search problem related to weighted model counting (WMC): Each maximally parallel rule is a model of an appropriately defined formula. Instead of the sum of all weights of all models (as in WMC), the goal is to enumerate all models and their weights.

The state space representation of MRSs groups equivalent variables, and reasons about them as a group. When computing the applicable rules (and their probabilities), we only need to reason about the *number* of entities of a species in a multiset, not their specific identities or ordering. This concept is related to counting formulae in C-FOVE, where probabilities only depend on the *number* of RVs of a parfactor with a specific value, and not the specific identities. For example, in the predator-prey scenario above, the probability of applying the reproduction rule depends only on the number of prey, and the probability of applying the eating rule depends only on the number of predator-prey pairs. However, the probability does not depend on presence of any specific predator or prey entity.

However, there is no way for existing MRS algorithms to reason about individual entities: All entities belonging to the same species are exactly identical. From our point of view, a MRS always operates on an abstract representation, and never propositionalizes the state space (by identification of specific entities). Therefore, splitting and merging operations are not meaningful for this representation.

5.4 Logical Particle Filter

The logical particle filter (LPF) (Zettlemoyer, Pasula, & Kaelbling, 2008) is a Bayesian filtering algorithm where states are described by partially instantiated first-order logical formulae. Each of those state descriptions actually represent a set of ground states (all instantiations of the formulae).

Example 17. Consider the dynamic smokers scenario (Example 8, Figure 2). Suppose we know that exactly one person has cancer, but we do not know which person. Furthermore, it is known that Bob smokes, and all other state variables are unknown. This situation can be represented by a single logical state in the LPF (representing the set of all 8 ground

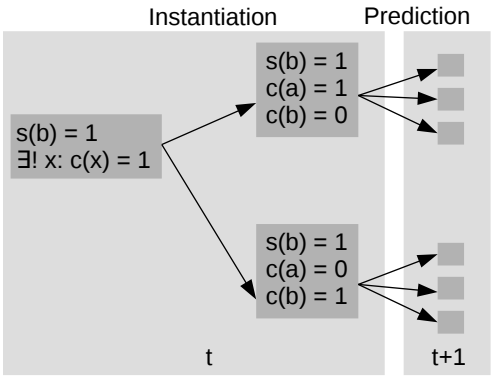


Figure 9: Depiction of the logical particle filter for Example 8. The instantiation step materializes all predicates necessary to calculate the transition model. Here, we assume that the values of $c(a)$ and $c(b)$ must be known to calculate the state transitions. Thus, all instantiations of $\exists!x : c(x)=1$ are materialized.

states that correspond to this situation):

$$s(b)=1, \exists!x : c(x)=1$$

Two examples for ground states that are represented by this logical state are:

$$s(a)=1, s(b)=0, c(a)=1, c(b)=0, d=0$$
$$s(a)=1, s(b)=0, c(a)=0, c(b)=1, d=1$$

○

The transition model is described in terms of *rules* that have preconditions and probabilistic effects. A state transition is performed as follows: First, a split operation is applied, which is necessary to determine which state transition rules are applicable in the current state.

Example 18. Suppose that the transition model requires that the *specific* person having cancer is known (for example because the probability of Bob dying from cancer is higher than the probability of Alice dying from cancer). The state

$$s(b)=1, \exists!x : c(x)=1$$

is split into two states:

$$s(b)=1, c(a)=1, c(b)=0$$
$$s(b)=1, c(a)=0, c(b)=1$$

Note that these two states still represent multiple ground states each. ○

Afterwards, the transition model is applied to each state separately (in the same way as in a standard particle filter). The situation is depicted in Figure 9.

The LPF implicitly groups multiple RVs of a state: In the state $s(b)=1, \exists! x : c(x)=1$, it is not specified *which* specific person has cancer, only that the *number* of people having cancer is one. In a way, this representation exploits the *exchangeability* of the RVs $c(a)$ and $c(b)$ in the underlying distribution described by the state $\exists! x : c(x)=1$. However, opposed to lifted inference algorithms, this capability to exploit exchangeability is limited: There is no formalism to specify that a *specific number* of RVs have a certain value (like counting formulae in lifted inference), and no algorithmic solution to handle such cases has been proposed.

A problem not devised by the LPF is that predicates that are instantiated once stay instantiated for this particle, i.e. merging operations for LPFs have not yet been devised. This can lead to a complete propositionalization of the state space over time. Zettlemoyer et al. (2008) acknowledges that a merging operation would be necessary to apply LPF to realistic domains.

5.5 Relational Particle Filter

The relational particle filter (RPF) (Nitti et al., 2013, 2014, 2016) is a Bayesian filtering algorithm where states, as well as the transition and observation model, are described by *distributional clauses*.

Distributional clauses are a way to describe conditional probabilities, closely related to parfactors. They have the form $h \sim D \leftarrow B \sim= b$, which describes the probability $p(h|B{=}b) = D$. Each of H, B and D can have logical variables. For example, the clause

$$size(X) \sim beta(2,3) \leftarrow material(X) \sim= metal$$

describes a conditional probability $p(size(X) \,|\, material(X){=}metal)$ for each X. A *dynamic* distributional clause (DDC) furthermore allows RVs to have time indices. Thus, DDCs can be used to describe the conditional probabilities $p(x_t \,|\, x_{t-1})$ and $p(y_t \,|\, x_t)$ of Bayesian filtering models. The algorithm performs particle filtering, using distributional clauses for the transition and observation model. Each particle is an assignment of values to the RVs, where some RVs may not have a specific value, but a distribution that is assigned to them.

Example 19. Consider the dynamic smokers scenario (Example 8, Figure 2). The transition model is described in terms of a DDC. For example, the DDC

$$c(X)_t \sim bernoulli(0.5) \leftarrow s(X)_{t-1} \sim= 1$$
$$c(X)_t \sim bernoulli(0.1) \leftarrow s(X)_{t-1} \sim= 0$$

describes that the probability of each person having cancer depends on the smoking state of this person at the previous time step. Other aspects of the transition and observation model are expressed in a similar fashion. As an example of a particle, suppose that one of the particles encodes the state where both persons do not smoke, but have cancer, and where the value of d_t (whether at least one person died at time t) follows a Bernoulli distribution:

$$s(a)_t{=}0, s(b)_t{=}0, c(a)_t{=}1, c(b)_t{=}1, d_t{\sim}bernoulli(0.1)$$

∘

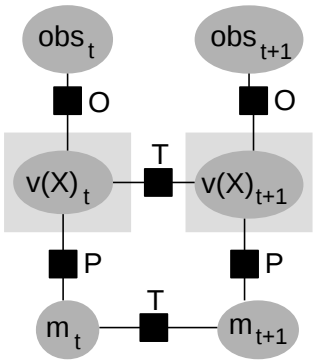


Figure 10: Parfactor graph describing the relational Kalman filter for Example 20. P parfactors describe the state distribution, T parfactors correspond to the transition model, O parfactors correspond to the observation model.

Thus, each particle actually describes a *distribution* of ground states, similar to the Rao-Blackwellized particle filter (RBPF). For example, the state above describes a distribution of two ground states with $d = 0$ and $d = 1$. A transition might require to know the specific value of an RV. This is achieved by sampling from the corresponding distribution – obtaining a new set of particles – and applying the transition model to each particle separately. This procedure is an instance of splitting. Similar to the LPF, the RPF can suffer from a complete grounding over time, as merging operations for the RPF have not yet been devised.

5.6 Relational Kalman Filter

The relational Kalman filter (Choi, Guzman-Rivera, & Amir, 2011b) is an algorithm for Bayesian filtering that is based on lifted inference, more specifically continuous FOVE (Choi, Amir, & Hill, 2010). The standard Kalman filter assumes a state that follows a multivariate normal distribution. Opposed to that, the state of the system in the relational Kalman filter is modeled as a relational pairwise model (RPM) (Choi et al., 2010), an extension of parfactor graphs where the par-RVs are continuous and the parfactors are normal distributions of arity 2 (the latter is a technical condition, as the inference operations only work for these parfactors). RPMs essentially represent a multivariate normal distribution with additional independence assumptions. The transition and observation model are also defined by RPMs. Based on this state representation, a Bayesian Filtering algorithm is defined, that is, *predict* and *update* steps are iteratively applied. Both steps are performed by employing continuous FOVE (Choi et al., 2010), i.e. by marginalizing out variables of the previous time step.

Example 20. The true value of a number of real estates is to be estimated over time, based on observations of sales prices and other factors, like the housing market index. The value of real estate i at time t is modeled as a Gaussian RV $v_t(i)$, and the housing market index is modeled as a Gaussian RV m_t. At each step, several sales prices will be observed. If

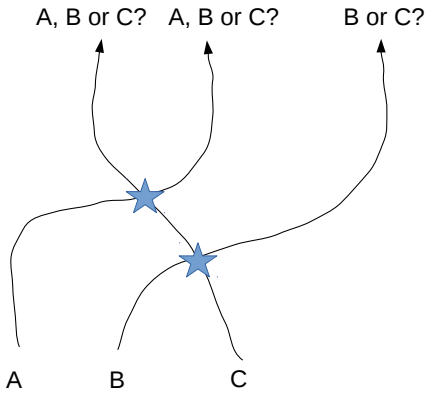


Figure 11: Data association problem. Three objects A, B and C move in 2D space. The identities of the objects cannot be observed directly. When they come too close, we get confused about the correspondence of the objects and the tracks (adapted from Huang et al., 2009b).

we initially assume each real estate to have an identical value, the estimated $v_t(i)$ will be the same for all unobserved i. Thus, all of these values can be represented by a single, parametric RV $v_t(X)$. The dependency between the state RVs at a single time step t is represented by a parfactor (specifically, an RPM) $P\big(v_t(X), m_t\big)$ and the observation model is an RPM $O\big(v_t(X), obs_t\big)$. The transition model can (for example) be described by RPMs $T_v\big(v_t(X), v_{t+1}(X)\big)$ and $T_m\big(m_t, m_{t+1}\big)$. Figure 10 shows the parfactor graph describing the situation. The predict and update steps thus have to be performed only once for each par-RV, instead of once for each RV. For the predict step, the par-RVs $v(X)_t$ and m_t are marginalized out of the joint distribution of par-RVs of time t and $t+1$. For the update step, the distribution of par-RVs is updated, based on the new observation obs_{t+1}. ∘

The key challenge of the relational Kalman filter arises when individual observations about RVs corresponding to the same par-RV are made. In this case, in general, a split operation needs to be performed to handle each observed RV individually. Interestingly, splitting is not necessary when only the means of the ground RVs become distinct, but only when the variances of the RVs become distinct. Choi, Amir, Xu, and Valocchi (2015) describe an algorithm to approximately merge variables that have become distinct due to observations.

This approach groups equivalent variables and reasons about them as a group (group variables), and also represents variables parametrically, as a Gaussian distribution (parametrization). Thus, it is the only approach we know of that exploits both types of lifting defined in this review. However, the approach is limited in its applicability, because it only allows Gaussian RVs and a linear transition model.

$$\begin{pmatrix} 2 & 12 & 4 & 4 \\ 1 & 2 & 11 & 0 \\ 10 & 4 & 4 & 15 \\ 5 & 2 & 1 & 2 \end{pmatrix}$$

(a) Information Matrix.

$$\hat{A} = \underset{A}{\operatorname{argmax}} \operatorname{tr} A^T \Omega = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

(b) Most likely association.

Figure 12: Illustration of the information form approach for data association (adapted from Schumitsch et al., 2005).

5.7 Data Association

Data Association algorithms are concerned with the following problem: Given a number of *tracks* $t_1, ..., t_n$ (e.g. radar measurements, tracks of people in a video) that correspond to objects $o_1, ..., o_n$, maintain the correct correspondence between tracks and objects (or, more general, a distribution of object-track associations). The problem is visualized in Figure 11. This problem can be viewed as performing Bayesian filtering in a state space where each state is a permutation of objects. There are $n!$ many of these permutations, so the naive approach to maintain a distribution of those permutations explicitly suffers from the state space explosion problem. Thus, the central task of Data Association algorithms is to maintain an efficient representation of distributions of permutations, and mechanisms to perform the predict and update steps of Bayesian filtering directly on this representation. Two conceptually different approaches for this goal have been devised. The first one, known as the Fourier-theoretic approach (Huang et al., 2009a, 2009b, 2009c; Jagabathula & Shah, 2011; Kondor et al., 2007), utilizes a Fourier transformation over the symmetric group $\mathbb{S}_n$ (the group that represents permutations of n objects). Instead of maintaining the complete distribution $p(\sigma)$, $\sigma \in \mathbb{S}_n$, the distribution is approximated by its first few Fourier matrices, just like a function $f(x)$, $x \in \mathcal{R}$ can be approximated by its first few Fourier coefficients.

The second approach (Schumitsch, Thrun, Bradski, & Olukotun, 2005) maintains a compact representation of the distribution over permutations matrices by an *information matrix* Ω. The information matrix contains unnormalized marginal probabilities Ω_{ij} for each association of track i with identity j. The following example illustrates the approach.

Example 21. Suppose we are tracking four objects. The distribution of object-track associations can be represented by the information matrix shown in Figure 12a. The first column corresponds to track 1, and the values imply the association of this track with the four objects, suggesting that track 1 is most strongly associated with object 3 (since this is the largest value in the column). However, the most likely permutation matrix, shown in Figure 12b, shows that actually, track 1 is most likely associated with object 4 (i.e. it is not sufficient to consider the columns separately). ◦

Given the information matrix Ω, we can calculate the probability of any permutation matrix A as $p(A) = 1/Z \exp \operatorname{tr} A^T \Omega$. Calculating the partition function Z is difficult, as it involves summing over all permutation matrices. However, the predict and update steps of the Bayesian filter can be performed directly on the information matrix: The observation of

an association of a track i with a specific object j leads to an increase of the corresponding value Ω_{ij}, and the mixing of tracks i_1 and i_2 leads to the same values in columns i_1 and i_2 in the information matrix.

Both approaches have been compared by Jiang, Huang, and Guibas (2011). They found that the Fourier-theoretic approach is better suited for scenarios with high uncertainty, while the information-theoretic approach is better suited for scenarios with low uncertainty about the data association.

To sum up, both approaches represent a (high-dimensional) distribution compactly. They do this by transforming the distribution into a different space, where it is easy to find a compact, approximate representation. This transformation can be seen as a form of parameterization, as defined in Section 3: The Fourier coefficients are the parameters of a mixture model of complex exponential functions. Operations corresponding to splitting or merging are not necessary in this setting: The distribution is always represented in this transformed space, and a grounding (in this case, an transformation back into the original space) is never necessary.

6. A Guide to Identify Suitable State-Space Abstraction Approaches

The goal of this section is to provide a useful guideline for practitioners to identify appropriate algorithms (or algorithmic ideas) for a given problem. It also summarizes our findings regarding research question **Q2**: *How can we characterize the problem classes that each of the 8 groups of approaches can solve?* We do so by rephrasing the properties of the algorithms as properties of the problem domains. At the same time, this perspective shows interesting problem domains that are not addressed by current approaches, and therefore identifies interesting directions for future research. As stated previously, the properties of the algorithms (see Table 3) directly provide a characterization of the application domain:

- Online algorithms are applicable to inference problems for sequential processes (e.g. the dynamic smokers domain of Example 8).

- Identification is necessary in two cases: Either observations about individuals are made (e.g. we observe that an individual person smokes), or the individuals need to be distinguished for some other reason (for example, because the transition model in a dynamic model requires to know the value of an individual RV, as in the variant of the dynamic smokers domain in Example 18).

- Grouping of Variables means that the algorithm can exploit exchangeability in the state space, i.e. a regular structure between multiple variables. Algorithm that have this capability can potentially solve some problems (that exhibit exchangeability) more efficiently (typically, for problems that do do not exhibit exchangeability, the algorithms simply resort to propositional inference). For example, algorithms that can group variables can potentially solve the smokers domain (Example 4) more efficiently that propositional inference algorithms.

- Parametrization allows the inference algorithm to exploit a regular structure in the distribution of a single variable. This is necessary for domains with continuous variables

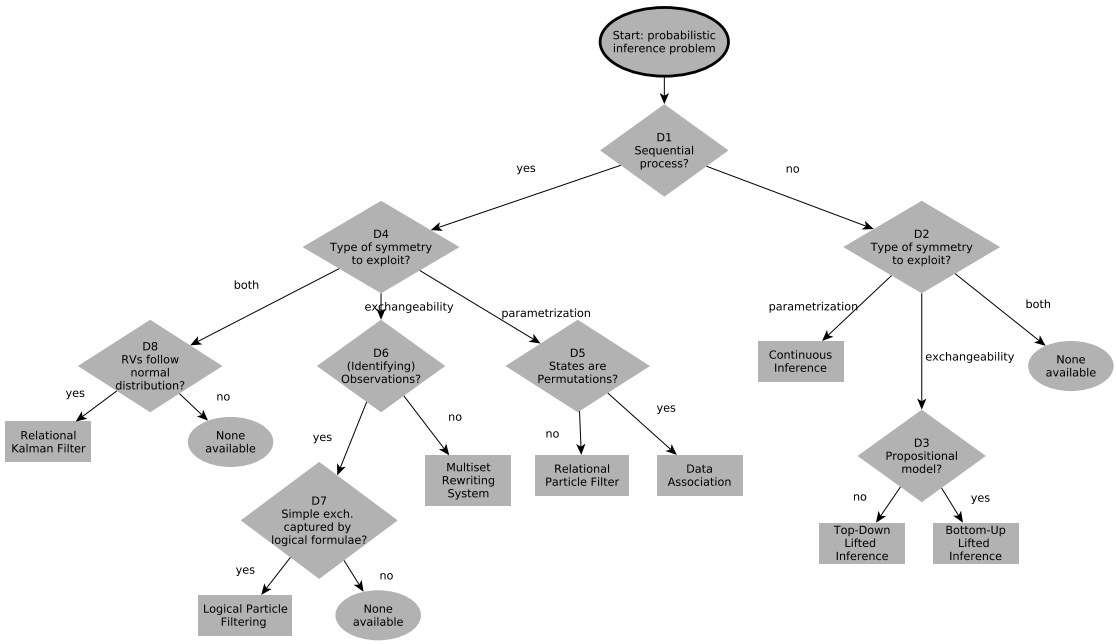


Figure 13: Decision diagram to decide on appropriate method to solve a given problem instance. Diamonds denote decisions, rectangles denote categories of inference algorithms, and paths to ellipses describe problems for which no inference algorithm (that performs state-space abstractions) exists yet.

(see Example 13), but discrete domains (like Data Association tasks) can also benefit from parametrization.

- Splitting operations are necessary for algorithms that start with a lifted representation (for example, a lifted graphical model or a first-order logic state representation, as in the Logical Particle Filter) and then need to identify individuals (as outlined for the identification property).
- Merging operations make the algorithm applicable to problems that are given in propositional form, but still contain symmetric properties (e.g. the ground factor graph for the smokers domain, Example 4). Merging is also useful in cases where the representation propositionalizes over time due to repeated splitting: In these cases, merging operations can re-introduce a compact representation (as for example done in the relational Kalman filter, see Choi et al., 2015).

Based on these considerations, we can identify approaches or algorithmic ideas that are suitable to solve a given problem in a lifted way (see Figure 13). The first decision (**D1**) is concerned with whether the system develops sequentially over time, constantly receiving new observations (requiring an approach capable of online inference), or not. Next, one needs to decide on the type of state space abstraction that the inference problem is susceptible of (**D2** and **D4**): Either the distribution exhibits exchangeability, or some (marginal) distribution can be represented in a parametric form, or both (if both is not the case, then

the inference problem does not allow any state space abstraction of the types investigated in this review). For non-sequential problems (**D2**), continuous inference algorithms can be used when the probabilistic model contains continuous variables whose distributions can be modeled (or approximated) by piecewise polynomial functions. When some of the RVs in the model are exchangeable, lifted inference algorithms can exploit this fact for efficient inference. Depending on the input format of the model (**D3**), bottom-up or top-down lifted inference algorithms can be applied. For lifted inference to be polynomial in the domain size, certain conditions have to apply, as discussed in Appendix A. However, even when these conditions do not apply, lifted inference can be more efficient than propositional inference. A combination of continuous and lifted inference algorithms, that can exploit both exchangeability and parametric distributions, has not been devised yet.

Sequential processes, on the other hand, require online algorithms to cope with indefinite observation sequences and unlimited numbers of RVs. Here again, one has to decide on the applicable type of state space abstraction (**D4**). When some of the marginal distributions can be represented in parametric form, the relational particle filter (or Rao-Blackwellized filtering in general) can be used. Some specific form of parametrization – data association methods – can be used when the state space consists of permutations (**D5**).

There are two categories of online algorithms that exploit exchangeability: When no observations are made (**D6**), Multiset rewriting systems can be used. Otherwise, the logical particle filter may be applicable, which has, however, very limited capabilities to make use of exchangeability (**D7**). The relational Kalman filter exploits exchangeability, and also makes use of parametric distributions – as long as all distributions are Gaussian (**D8**).

Finally, coming back to the three examples from the introduction, Example 1 (smokers) can be solved with a lifted inference algorithm, Example 2 (office) with a data association approach, and Example 3 (biochemical reaction) with a multiset rewriting system.

7. Conclusion and Future Work

In this section, we discuss the open problems identified in Section 6 in more detail. We propose ideas on how the methods identified in this review could be combined or extended appropriately, to devise an algorithm that can solve this problem class.

7.1 Future Work

As becomes obvious from the flow chart in Figure 13, there is no algorithm that can exploit exchangeability (as lifted inference does) *and at the same time* handle (continuous) distributions parametrically (as continuous inference algorithms do). One can easily imagine a scenario where this would be beneficial: Consider an object localization task similar to Example 13, but multiple objects are observed. A combination of both state space abstraction types is, for example, conceivable for variable elimination, for which both lifted inference (Taghipour et al., 2014) and parametric inference (Sanner & Abbasnejad, 2012) approaches exist, or weighted model counting, for which also both lifted inference (Van Den Broeck et al., 2011) and parametric inference (Belle et al., 2015a) exists.

For sequential inference tasks, there is no algorithm that can exploit exchangeability to the same extend as lifted inference does for non-sequential inference. This would, however, be relevant for sequential inference problems exhibiting exchangeability, like the dynamic

smokers domain presented in Example 8. The relational Kalman filter requires all factors to be Gaussian, and logical particle filtering cannot handle statements about counts, like "exactly 3 RVs of this group of RVs are true, and it does not matter which". Multiset rewriting systems (MRSs) can efficiently handle systems with exchangeable RVs – however, current MRSs do not provide a mechanism to incorporate evidence about specific individuals (say, we know that Bob is having cancer at $t = 10$). In the following, we sketch two ideas that seem promising for developing a general lifted Bayesian filtering algorithm.

One idea is to base such an algorithm on an MRS, i.e. a Bayesian filtering algorithm with a multiset-based state description and a transition model defined in terms of rewriting rules. Such a system directly allows to group equivalent aspects of the state by the multiset state representation. The crucial aspect for such a system is the way the state space abstractions are represented, i.e. how similar entities can be grouped, despite the fact that they may not be completely the same (e.g. because we have distinct observations about them).

The other idea is to base the algorithm on lifted inference approaches, and examine how they can be used to implement the predict and update step of Bayesian filtering. A first step in this direction are filtering algorithms for dynamic MLNs (Geier & Biundo, 2011; Papai, Kautz, & Stefankovic, 2012), that require using a probabilistic inference algorithm at each time step. However, the effects of using a *lifted* inference algorithm each time has not been evaluated yet, and it is unclear how to maintain a lifted state representation over time.

Furthermore, exploiting both exchangeability and parametric distributions is also relevant for sequential inference algorithms. As an example, consider the localization task from Example 13, but for multiple agents instead of a single agent. Here, some RVs (locations, sensor measurements) are continuous, and RVs of different agents can be exchangeable (e.g. we might know that two agents have the same location distribution, one agent has a different location distribution, but we do not know which agent is associated with which distribution).

In general, one of the most challenging aspects of inference algorithms that exploit symmetries is the question how to prevent the state representation from degenerating (become increasingly grounded), as individual observations that break the symmetries in the state space are received. Kersting (2012) notes: "Even if there are symmetries within a probabilistic model, they easily break when it comes to inference since variables become correlated by virtue of depending asymmetrically on evidence" (p. 37). This is specifically problematic for *exact* algorithms, that need to consider even slight symmetry breaks. For non-sequential models, approaches that can handle this problem by finding *approximate* symmetries in the model have been proposed (Singla et al., 2014; Venugopal & Gogate, 2014a; Van Den Broeck & Niepert, 2015) – which can gain even more efficiently by also using approximate inference algorithms (like belief propagation or sampling).

The problem is even more prevalent in sequential lifted inference algorithms: Even when evidence for a single time step leads only to a slight degeneration of the symmetries, over time, the complete model will become ground. On the other hand, the prediction step in Bayesian filtering might lead to an *increase* in symmetry: The intuition here is that the prediction in general increases the uncertainty of the state estimate, potentially (partially) revoking the effect of symmetry-breaking evidence. There is only very few research on this aspect for sequential models. For the relational Kalman filter, an approach has been

devised that approximately regroups state variables after symmetry-breaking evidence has been observed (Choi et al., 2015), relying on the fact that the difference in these variables can be bound under certain conditions. In general, the idea of occasionally performing operations that re-introduce (approximate) symmetries seems to be a promising idea for more general lifted Bayesian filtering algorithms.

7.2 Conclusion

Probabilistic inference is the task to derive the probability of certain random variables, given the values of other variables and a model for the relationship between the variables. In many cases, symmetries and redundancies are implicitly present in the model, which cannot be exploited by conventional inference algorithms. In the last 15 years, inference methods have been devised that can exploit the symmetric structure to speed up inference and thus make it feasible for much larger models.

In this article, we presented the results of a systematic review concerned with these methods. We identified eight classes of such inference algorithms, which have been grouped based on their common properties, and thus the common problems they can be applied to. For the first time, this systematic review presented a unified view of these methods, that have been devised by different research communities. Specifically, we emphasized inference algorithms for sequential processes (Bayesian filtering), a relevant application domain that has been neglected by lifted inference algorithms, and is not discussed in previous reviews of the same topic (Kersting, 2012; Kimmig et al., 2015). We found that no Bayesian filtering algorithm has been devised yet that can exploit symmetries to the same extend as lifted inference algorithms do for non-sequential inference. Developing such an algorithm might be approached by employing ideas from lifted inference or multiset rewriting systems.

One of the main problems underlying all approaches is *symmetry-breaking evidence* that makes it difficult to maintain a lifted representation. This problem is very prevalent in real-world scenarios, like sensor data processing, and solutions for non-sequential inference algorithms based on finding approximate symmetries have been proposed. Investigating how to cope with this problem in the context of Bayesian filtering is an interesting future research topic.

Acknowledgements

We are grateful to the three anonymous reviewers for their extensive comments and suggestions, which vastly improved the quality of the paper.

Appendix A. Lifted Inference Complexity Classes

Recently, attempts have been made to structure the problem classes for lifted inference algorithms, based on whether they can be solved efficiently. In general, there is no guarantee that lifted inference is tractable (i.e. has a polynomial runtime), Jaeger and Van den Broeck (2012) even showed the existence of intractable inference problems.

However, there are problem classes for which tractability guarantees can be given. To analyze them, it is useful to define inference problems in terms of *weighted model counting* on a first-order knowledge base (see Section 5.1.1). Using this representation, different

problem classes can be defined regarding the specific fragment of first-order logic used (FO: function-free first-order logic and $RFOL$: FO without constant symbols), allowed quantifiers, and the maximum number of logical variables per formula.

The central notion is that of *domain-lifted* algorithms. An algorithm is domain-lifted for a problem class, iff for all instances of this problem class, inference is polynomial in the domain size of the logical variables. Table 4 shows domain-liftability results for different algorithms and problem classes. Note that this table shows only results regarding domain-lifting. Results regarding other definitions of lifting (e.g. approximate liftability) are discussed by Jaeger and Van den Broeck (2012).

It turns out that inference on knowledge bases with at most two logical variables per formula (FO^2) is domain-liftable, i.e. all instances of this class can be solved in polynomial time with respect to the domain size (as well as a generalization, S^2FO^2). Furthermore, at least two inference algorithms are known that can actually perform inference for this problem class in polynomial time: WFOMC, as proposed by Van Den Broeck (2011) and Van Den Broeck et al. (2014), and the FOVE variant of Taghipour et al. (2013b). On the other hand, it was shown that for general FO, Lifted Inference is not polynomial in the domain size.

Another class of inference problems that is known to be domain-lifted is *recursive unary* (RU), which basically describes the problems that can be solved in polynomial time by lifted recursive conditioning (Section 5.1.1): A theory is in RU when exhaustively applying the rules of lifted recursive conditioning leads to a theory which contains a predicate that has only a single logical variable. Then, the algorithm can branch on this atom, generating branches for all *numbers* of corresponding RVs being true. It has been shown that RU subsumes FO^2 (Kazemi et al., 2016).

An example of a problem where no domain-lifted marginal inference algorithm is known is the *transitive* formula $friends(X, Y) \wedge friends(Y, Z) \Rightarrow friends(X, Z)$ (although MAP inference can be performed efficiently for this formula, see Mittal, Goyal, Gogate, & Singla, 2014).

Appendix B. Related Approaches

There are multiple approaches that touch upon related topics as the ones explored in this review, but have not been included. In the following, we will discuss their connection to the approaches examined by this review, and argue why each of them did not match our inclusion criteria.

B.1 Knowledge-Based Model Construction

Knowledge-based model construction (KBMC) is a type of inference algorithm for lifted graphical models. They work by completely grounding the model and performing standard probabilistic inference in the propositional model.

There are numerous extensions and improvements that have been proposed for these algorithms. For example, Richardson and Domingos (2006) ground only those formulae necessary to answer the query. Singla and Domingos (2006) propose a *lazy* KBMC algorithm that performs grounding on the fly. Glass and Barker (2012) propose an approximate algorithm that only produces the most relevant ground formulae, and ignores the rest.

Algorithm	KB	DL	Reference
All	$RFOL(\forall\exists =)$	□	(Jaeger, 2000)
WFOMC (Van Den Broeck et al., 2011, 2014)	$FO^2(\forall\exists =)$	■	(Van Den Broeck, 2011; Van Den Broeck et al., 2014)
WFOMC (Beame et al., 2015)	γ-acyclic query[10]	■	(Beame et al., 2015)
LRC (Poole et al., 2011)	RU	■	(Poole et al., 2011)
LRC (Kazemi et al., 2016)	$S^2FO^2(=)$[11]	■	(Kazemi et al., 2016)
LRC (Kazemi et al., 2016)	S^2RU	■	(Kazemi et al., 2016)
FOVE (de Salvo Braz et al., 2005)	$FO^1(=)$	□	(Taghipour et al., 2013b)
C-FOVE (Milch et al., 2008)	$FO^1(=)$	□	(Taghipour et al., 2013b)
C-FOVE$^\#$ (Taghipour et al., 2014; Apsel & Brafman, 2011)	$FO^1(=)$	■	(Taghipour et al., 2013b)
C-FOVE$^\#$ (Taghipour et al., 2014; Apsel & Brafman, 2011)	$FO^2(=)$	□	(Taghipour et al., 2013b)
C-FOVE$^+$ (Taghipour et al., 2013b)	$FO^2(=)$	■	(Taghipour et al., 2013b)

Table 4: Liftability results for different algorithms and problem classes. "All" algorithms mean that the result applies to all lifted inference in general. KB: Knowledge Base, DL: Domain-lifted, □: Not domain-lifted, ■: Domain-lifted (adapted from Taghipour et al., 2013c; Jaeger & Van den Broeck, 2012).

Using these methods, KBMC approaches can be more efficient than standard, propositional inference. However, at their core, they perform propositional inference. Thus, they do not match inclusion criterion 8.

B.2 Knowledge Compilation and Arithmetic Circuits

Arithmetic circuits are data structures that compactly represent functions. Early approaches like Binary Decision Diagrams (BDDs) represent boolean functions, but extensions have been devised for representing functions of the type $\mathbb{B}^n \to \mathbb{R}$. Darwiche and Marquis (2002) provide a detailed comparison of such approaches. The appeal of these methods is that they allow to efficiently answer specific classes of queries. *Knowledge compilation* exploits this fact, by tranforming logical formulae into such a formalism, which then allows efficient inference. The motivation is to perform this (potentially costly) transformation upfront, and then be able to answer a large number of queries on the compiled representation very fast.

This idea can be also be used for *probabilistic* inference: As discussed in Section 5.1.1, probabilistic inference can be transformed into a weighted model counting (WMC) problem. Knowledge compilation can then be used to solve the WMC problem efficiently. On the other hand, arithmetic circuits can also be used directly to represent distributions (instead

of using a conventional graphical model). Examples include variants of Ordered Binary Decision Diagrams (Jaeger, 2004; Dal & Lucas, 2017), Algebraic Decision Diagrams (Sanner & McAllester, 2005), and Sum-Product Networks (Gens & Domingos, 2013). In these formalisms, some probabilistic inference operations can be performed in polynomial time (in the size of the circuit).

These approaches have not been included in this review because they do not perform any of the two investigated state space abstraction methods: They do not group similar RVs, and they do not operate on parameters of the distributions (i.e. they do not match the definition of state space abstraction that we use). Still, they are related to the methods presented here insofar as they pursue the same overall goal: Compact representations of distributions, and efficient operations on this representations that lead to efficient inference.

Knowledge compilation methods and arithmetic circuits have successfully been combined with other state space abstraction methods: Combining knowledge compilation with lifted inference ideas yields *first-order* knowledge compilation (Van Den Broeck et al., 2011), discussed in Section 5.1.1. Arithmetic circuits (specifically, ADDs) are used for symbolic variable elimination (outlined in Section 5.2), to keep the representation of the case statements compact.

B.3 Markov Decision Processes

A Markov decision process (MDP) is a model for sequential decision making where an agent has to select actions based on the current environment state. Each action is associated with a reward. Given an MDP, the task is to compute an optimal *policy*, i.e. a function that assigns each state a corresponding action such that the long-term reward is maximized. The optimal policy can be obtained by computing the *value function* (that assigns a value to each state) using dynamic programming. Puterman (2014) provides a more thorough introduction into algorithms for solving MDPs.

MDPs also suffer from the state space explosion problem, and solutions similar to some of the algorithms discussed in Section 5 have been developed. These methods follow two basic ideas. The first approach is to find symmetries in the state space of an MDP and group symmetric state, thus obtaining a smaller state space (Dean & Givan, 1997; Givan, Dean, & Greig, 2003; Kang & Kim, 2012). The second approach is to perform all operations within a more compact first-order representation (Boutilier, Reiter, & Price, 2001; Kersting, Van Otterlo, & De Raedt, 2004; Hölldobler & Skvortsova, 2004; Sanner & Boutilier, 2009; Wang, Joshi, & Khardon, 2008). In these approaches, states, actions, reward functions, value functions and policies are all based on first-order logic. This way, the resulting policy can be independent of the actual domain objects, and the computations to obtain this policy can be independent of the domain size. A problem is that the (logical) representation of the value function can easily become very complex and redundant, requiring expensive first-order simplification. A first-order extension of ADDs (first-order ADDs, FOADDs) has been devised, which can be used to compactly represent the value function. Furthermore, approximate methods (like approximate first-order linear programming), that avoid this problem, can be used. Conceptually, first-order MDPs (and their solution techniques) bear strong relationships to lifted probabilistic inference: Both are concerned with first-order models, where parts of the model are redundant or identical. They both exploit these

symmetries to achieve more efficient algorithms, by performing operations "in bulk" for entire sets of redundant components.

However, there is also a more technical, intimate relationship between MDPs and probabilistic inference: It has been shown that decision problems (in terms of an MDP) can be cast into a probabilistic inference problem (Toussaint & Storkey, 2006). Thus, any probabilistic inference algorithm can be used to solve MDPs. From this point of view, first-order MDPs are an *application domain* of probabilistic inference (although research on both topics has mostly been distinct). This relationship also holds for first-order MDPs: Recently, Khardon and Sanner (2017) showed that the probabilistic inference problem that can be derived from a first-order MDP inherits its symmetric structure. This structure can be exploited by lifted inference, avoiding redundant computations. Due to the complex structure of the query, it is however not possible to use standard lifted inference algorithms here. Instead, the first-order dynamic programming approaches can be seen as performing some specialized lifted inference algorithm (that is completely independent of the domain size). An interesting perspective for future research is to combine the distinct innovations from both domains, and close the gap between the respective lines of research – a paradigm termed *generalized lifted inference* by Khardon and Sanner (2017).

Nevertheless, we chose to not consider first-order MDPs in Section 5 of this review. The reason is that the dynamic programming-based algorithms that are at the heart of solving MDPs do not *directly* involve probabilistic inference, and thus some of the properties derived in Section 3 are not meaningful for these algorithms. In other words, dynamic programming does not match inclusion criterion 6.

B.4 Statistical Relational Learning

A relevant question not discussed so far in this review is that of *learning* first-order probabilistic models. So far, we assumed that the models are given, and the only task is perform inference in these models. For many application domains, learning the model is one of the most relevant (and most challenging) aspects. For example, in tasks like link prediction (decide whether a specific relation exists between two objects) or entity resolution (decide which records in a database refer to the same real-world entity), we are given a rich, relational structure, and want to estimate a first-order probabilistic model describing this structure (to then perform inference in this model). The research field investigating this task is known as *statistical relational learning*. For an overview of the methods used in this field, we refer to the book by Getoor and Taskar (2007).

One can distinguish *parameter learning*, where the structure of the probabilistic model is given, and *structure learning*, where even the structure needs to be learned. In parameter learning, the goal is to optimize the likelihood of the model, given the data. This is, as in the propositional setting, typically done by *Expectation Maximization*: The parameters are computed in an iterative process, consisting of computing the expected likelihood of the model, given the current parameters, and maximizing this expectation function. In contrast to propositional models, however, multiple parameters may be tied in relational models (thus effectively reducing the total number of parameters). Parameter learning is a difficult task, as it requires to perform probabilistic inference (which is itself a hard problem) each time the expectation is computed. Thus, approximate methods are typically used, that

optimize easier to compute measures than the likelihood. Recently, exact (Van Haaren, b Van den Broeck, Meert, & Davis, 2016) and approximate (Ahmadi, Kersting, Mladenov, & Natarajan, 2013a) lifted inference has been used for parameter learning. Structure learning is even more challenging: The structure is also learned in an iterative process, requiring parameter learning at each step. Learning methods have been devised for a large number of probabilistic relational formalisms, including MLNs (Richardson & Domingos, 2006; Khot et al., 2011), Problog (Gutmann, Thon, & De Raedt, 2011), CP-logic (Thon, Landwehr, & De Raedt, 2011), PRISM (Sato & Kameya, 2001), probabilistic relational models (Getoor, Friedman, Koller, & Taskar, 2002) and Bayesian logic programs (Kersting & De Raedt, 2001).

As this paper focuses on *inference* rather than learning, these methods are not discussed in the main part of this review (i.e. they do not match inclusion criterion 6).

B.5 Logical Hidden Markov Models

Logical Hidden Markov Models (LHMMs) (Kersting, De Raedt, & Raiko, 2006; Natarajan, Bui, Tadepalli, Kersting, & Wong, 2008; Yue, Xu, Qin, & Yin, 2015b; Yue, Jiao, Zha, & Yin, 2015a) are similar to Hidden Markov Models (HMMs), except that each state consists of a logical atom. A LHMM transition consists of two steps. First, a ground atom is sampled based on the current state, i.e. the current logical atom. Then, an abstract transition is selected whose precondition matches the ground atom. This transition leads to a new abstract state. The filtering algorithm that has been presented for this representation requires considering all ground atoms. Thus, this approach does not match inclusion criterion 8.

B.6 Probabilistic Model Checking

Model Checking is concerned with the following problem: Given an abstract system specification, test if certain properties (defined in a temporal logic like LTL or CTL logic) are satisfied by the system. These specifications define a state space that is exhaustively searched to verify the property. A common technique is to not represent the state space explicitly, but *symbolically* as a propositional formula, that in turn is represented as a binary decision diagram (BDD). *Probabilistic* model checking furthermore models state transition probabilities.

In Model Checking, the state space explosion problem is very common. For example, when the system consists of multiple concurrent processes, each execution ordering needs to be considered, which leads to a combinatorial explosion in the state space (Clarke et al., 2001). Symbolic state space representation is one way to handle this problem. When the state space has a certain regular structure, the BDD representation can be much smaller than representing the state space explicitly. Other methods directly reduce the number of states, the most prominent ones being partial order reduction (POR) (Valmari, 1989; Peled, 1993; Godefroid, Van Leeuwen, Hartmanis, Goos, & Wolper, 1996) and symmetry reduction (Clarke, Emerson, Jha, & Sistla, 1998). These reduction methods follow similar ideas than bottom-up lifted inference algorithms: Starting with a propositional model, and finding symmetries in this model. Then, the model can be represented by a single representative of each set of symmetric state.

The reasons for excluding these approaches are similar to the reasons for excluding MDP-based approaches: Although they contain interesting ideas for state space reduction, the task and the used algorithms are completely different. This also means that the type of symmetry considered is quite different: In lifted inference, the symmetries must preserve the (conditional) probabilities of the RVs. In model checking, the symmetries must preserve the property we want to check.

B.7 Multiple Hypotheses Tracking

There is a large number of papers from the *data association* community that have not been included in this review. A prominent example for this class of algorithms is the multiple hypotheses tracker (Reid, 1979). It maintains all possible associations of measurements to objects explicitly. Therefore, it suffers from the state space explosion problem. Several approximation methods, like pruning (keeping only the most likely hypotheses) (Cox & Hingorani, 1996) have been developed. Other data association approaches have been proposed by Fortmann, Bar-Shalom, and Scheffe (1983), Han, Xu, Tao, and Gong (2004), and Oh, Russell, and Sastry (2004). None of these approaches employ state space abstractions, which is the reason why we did not consider them for this review.

B.8 Probabilistic Situation Calculus

The situation calculus (Reiter, 1991) is a first-order logic formalism to reason about dynamic domains that are changed by actions. Several approaches combine the situation calculus with some form of probabilistic model. In the works of Mateus, Pacheco, Pinto, Sernadas, and Sernadas (2001), and Hajishirzi and Amir (2008), actions have probabilistic effects. Bacchus, Halpern, and Levesque (1995, 1999), and Mateus et al. (2002) introduce uncertain observations (uncertainty about the current state). The problem that is solved by these approaches is: *Given a sequence of actions and an initial state, what is the probability that a first-order formula is true in the final state, after executing these actions?* This is done by providing an explicit distribution over all possible states (Bacchus et al., 1995, 1999), or by sampling-based approaches (Mateus et al., 2001, 2002; Hajishirzi & Amir, 2008).

This formalism provides a compact state representation, by representing states using first-order logic. However, no algorithm that can reason efficiently in this representation has been devised. In fact, the state representation can become arbitrarily complex, as noted by Boutilier et al. (2001).

Appendix C. Assignment of Papers to Groups

The following table shows the specific papers associated with each of the groups defined in Section 4.5.

Name	References
Top-down LI	(Poole, 2003) (Kisyński & Poole, 2009a) (de Salvo Braz et al., 2005) (de Salvo Braz et al., 2006) (Milch et al., 2008) (Apsel & Brafman, 2011) (Taghipour et al., 2014) (Taghipour et al., 2013b) (Taghipour et al., 2013c) (Das et al., 2016) (Taghipour et al., 2012) (Taghipour et al., 2013a) (Ng et al., 2008) (Ng & Lloyd, 2009) (Takiyama & Cozman, 2014) (Kisyński & Poole, 2009b) (Choi et al., 2011a) (Singla & Domingos, 2008) (de Salvo Braz et al., 2009) (Singla et al., 2010) (Singla et al., 2014) (Gogate & Domingos, 2016) (Gogate et al., 2012) (Van Den Broeck et al., 2011) (Van Den Broeck & Davis, 2012) (Van Den Broeck, 2011) (Van Den Broeck et al., 2014) (Meert et al., 2014) (Beame et al., 2015) (Vlasselaer et al., 2016) (Bui et al., 2012) (Gogate & Domingos, 2010) (Choi & Amir, 2012) (Jha et al., 2010) (Poole et al., 2011) (Kazemi & Poole, 2014) (Kazemi et al., 2016) (Kazemi et al., 2017) (Kiddon & Domingos, 2010) (Kiddon & Domingos, 2011) (Poon et al., 2008) (Sarkhel & Gogate, 2013) (Sarkhel et al., 2014) (Venugopal et al., 2015) (Mittal et al., 2014) (Domingos & Webb, 2012) (Dalvi et al., 2010) (Dalvi & Suciu, 2007) (Dylla et al., 2013) (Jha & Suciu, 2012)
Bottom-up LI	(Kersting et al., 2010) (Jaimovich et al., 2007) (Kersting et al., 2009) (Ahmadi et al., 2013b) (Ahmadi et al., 2013a) (Venugopal et al., 2016) (Venugopal & Gogate, 2014b) (Venugopal & Gogate, 2012) (Venugopal & Gogate, 2014a) (Van Den Broeck et al., 2012) (Hadiji & Kersting, 2013) (Sen et al., 2008) (Sen et al., 2009) (Bui et al., 2013) (Bui et al., 2014) (Niepert, 2012) (Anand et al., 2016) (Van Den Broeck & Niepert, 2015) (Niepert, 2013) (Mladenov et al., 2014a) (Apsel et al., 2014) (Mladenov et al., 2012) (Mladenov & Kersting, 2013) (Mladenov et al., 2014b) (Van Den Broeck & Darwiche, 2013) (Nath & Domingos, 2010b) (Nath & Domingos, 2010a) (Ahmadi et al., 2010) (Hadiji et al., 2011) (Ahmadi et al., 2011) (Geier & Biundo, 2011)
Continuous Inference	(Belle et al., 2015a) (Belle et al., 2015b) (Belle et al., 2016) (Sanner & Abbasnejad, 2012) (Shenoy & West, 2011)
Logical Particle Filter	(Zettlemoyer et al., 2008)
Relational Particle Filter	(Nitti et al., 2013) (Nitti et al., 2016) (Nitti et al., 2014)
Relational Kalman Filter	(Choi et al., 2010) (Choi et al., 2011b) (Choi et al., 2015)

Data Association	(Schumitsch et al., 2005) (Huang et al., 2009a) (Huang et al., 2009b) (Huang et al., 2009c) (Jagabathula & Shah, 2011) (Kondor et al., 2007) (Jiang et al., 2011) (Baum & Hanebeck, 2010) (Baum & Hanebeck, 2011) (Baum & Hanebeck, 2013) (Baum et al., 2012) (Baum et al., 2014) (Hanebeck & Baum, 2015) (Leven & Lanterman, 2004) (Leven & Lanterman, 2009) (Mahler, 2003)
Prob. Multiset Rewriting	(Barbuti et al., 2011) (Barbuti et al., 2012) (Krishnamurthy et al., 2004) (Warnke et al., 2015) (Bistarelli et al., 2003) (Oury & Plotkin, 2013) (Maus et al., 2011)

Appendix D. List of Abbreviations

Abbreviation	Explanation
BP	Belief propagation
C-FOVE	Counting first-order variable elimination
DBN	Dynamic Bayesian network
FOVE	First-order variable elimination
LBP	Lifted belief propagation
LI	Lifted inference
LP	Linear program
LPF	Logical particle filter
MAP	Maximum-a-posteriori
MCMC	Markov chain Monte Carlo
MDP	Markov decision process
MLN	Markov logic network
MRS	Multiset rewriting system
RC	Recursive conditioning
RV	Random variable
VE	Variable elimination
WFOMC	Weighted first-order model counting

References

Ahmadi, B., Kersting, K., & Hadiji, F. (2010). Lifted belief propagation: Pairwise marginals and beyond. In *Proceedings of the 5th European Workshop on Probabilistic Graphical Models*, pp. 9–16.

Ahmadi, B., Kersting, K., Mladenov, M., & Natarajan, S. (2013a). Exploiting symmetries for scaling loopy belief propagation and relational training. *Machine Learning*, *92*(1), 91–132.

Ahmadi, B., Kersting, K., & Natarajan, S. (2013b). MapReduce lifting for belief propagation. In *AAAI Workshop - Technical Report*, Vol. WS-13-16, pp. 2–7.

Ahmadi, B., Mladenov, M., Kersting, K., & Sanner, S. (2011). On lifted pagerank, kalman filter and towards lifted linear program solving. In *Technical Report of the Symposium "Lernen, Wissen, Adaptivitat - Learning, Knowledge, and Adaptivity 2011" of the GI Special Interest Groups KDML, IR and WM*, pp. 35–42.

Anand, A., Grover, A., Mausam, & Singla, P. (2016). Contextual symmetries in probabilistic graphical models. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*.

Apsel, U., & Brafman, R. (2011). Extended Lifted Inference with Joint Formulas. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, UAI'11, pp. 11–18, Barcelona, Spain. AUAI Press.

Apsel, U., Kersting, K., & Mladenov, M. (2014). Lifting relational MAP-LPs using cluster signatures. In *AAAI Workshop - Technical Report*, Vol. WS-14-13, pp. 2–8.

Bacchus, F., Halpern, J., & Levesque, H. (1999). Reasoning about noisy sensors and effectors in the situation calculus. *Artificial Intelligence*, *111*(1-2), 171–208.

Bacchus, F., Halpern, J., & Levesque, J. (1995). Reasoning about noisy sensors in the situation calculus. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pp. 1933–1940.

Barbuti, R., Levi, F., Milazzo, P., & Scatena, G. (2011). Maximally Parallel Probabilistic Semantics for Multiset Rewriting. *Fundamenta Informaticae*, *112*(1), 1–17.

Barbuti, R., Levi, F., Milazzo, P., & Scatena, G. (2012). Probabilistic model checking of biological systems with uncertain kinetic rates. *Theoretical Computer Science*, *419*, 2 – 16.

Baum, M., & Hanebeck, U. (2010). Association-free tracking of two closely spaced targets. In *Proceedings of the IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems*, pp. 62–67. IEEE.

Baum, M., & Hanebeck, U. (2011). Using symmetric state transformations for multi-target tracking. In *Proceedings of the 14th International Conference on Information Fusion*, pp. 1–8. IEEE.

Baum, M., & Hanebeck, U. (2013). The kernel-sme filter for multiple target tracking. In *Proceedings of the 16th International Conference on Information Fusion*, pp. 288–295. IEEE.

Baum, M., Ruoff, P., Itte, D., & Hanebeck, U. (2012). Optimal Point Estimates for Multi-target States based on Kernel Distances. In *Proceedings of the 51st IEEE Conference on Decision and Control*, Maui, Hawaii, USA.

Baum, M., Willett, P., & Hanebeck, U. (2014). MMOSPA-based track extraction in the PHD filter-a justification for k-means clustering. In *Proceedings of the IEEE 53rd Annual Conference on Decision and Control*, pp. 1816–1821. IEEE.

Beame, P., Van den Broeck, G., Gribkoff, E., & Suciu, D. (2015). Symmetric Weighted First-Order Model Counting. In *Proceedings of the 34th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, pp. 313–328.

Belle, V., Passerini, A., & Van den Broeck, G. (2015a). Probabilistic inference in hybrid domains by weighted model integration. In *Proceedings of 24th International Joint Conference on Artificial Intelligence*, pp. 2770–2776.

Belle, V., Van den Broeck, G., & Passerini, A. (2015b). Hashing-based approximate probabilistic inference in hybrid domains. In *Proceedings of the 31st Conference on Uncertainty in Artificial Intelligence (UAI)*, pp. 141–150.

Belle, V., Van den Broeck, G., & Passerini, A. (2016). Component Caching in Hybrid Domains with Piecewise Polynomial Densities.. In *AAAI*, pp. 3369–3375.

Bistarelli, S., Cervesato, I., Lenzini, G., Marangoni, R., & Martinelli, F. (2003). On representing biological systems through multiset rewriting. In *International Conference on Computer Aided Systems Theory*, pp. 415–426. Springer.

Boutilier, C., Reiter, R., & Price, B. (2001). Symbolic dynamic programming for first-order MDPs. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, Vol. 1, pp. 690–700.

Bui, H., Huynh, T., & De Braz, R. (2012). Exact lifted inference with distinct soft evidence on every object. In *Proceedings of the National Conference on Artificial Intelligence*, Vol. 3, pp. 1875–1881.

Bui, H., Huynh, T., & Riedel, S. (2013). Automorphism groups of graphical models and lifted variational inference. In *Uncertainty in Artificial Intelligence - Proceedings of the 29th Conference*, pp. 132–141.

Bui, H., Huynh, T., & Sontag, D. (2014). Lifted tree-reweighted variational inference. In *Uncertainty in Artificial Intelligence - Proceedings of the 30th Conference*, pp. 92–101.

Buntine, W. (1994). Operations for learning with graphical models. *Journal of artificial intelligence research*, *2*, 159–225.

Calude, C., Paun, G., Rozenberg, G., & Salomaa, A. (2001). *Multiset Processing: Mathematical, Computer Science, and Molecular Computing Points of View*, Vol. 2235. Springer Science & Business Media.

Cervesato, I., Durgin, N. A., Lincoln, P. D., Mitchell, J. C., & Scedrov, A. (1999). A Meta-Notation for Protocol Analysis. In *Proceedings of the 12th IEEE Workshop on Computer Security Foundations*, CSFW '99, pp. 55–, Washington, DC, USA. IEEE Computer Society.

Choi, J., & Amir, E. (2012). Lifted Relational Variational Inference. In *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence*, UAI'12, pp. 196–206, Catalina Island, CA. AUAI Press.

Choi, J., Amir, E., & Hill, D. (2010). Lifted Inference for Relational Continuous Models. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, UAI'10, pp. 126–134, Catalina Island, CA. AUAI Press.

Choi, J., Amir, E., Xu, T., & Valocchi, A. (2015). Learning Relational Kalman Filtering.. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pp. 2539–2546.

Choi, J., de Salvo Braz, R., & Bui, H. (2011a). Efficient Methods for Lifted Inference with Aggregate Factors.. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*.

Choi, J., Guzman-Rivera, A., & Amir, E. (2011b). Lifted Relational Kalman Filtering.. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*, pp. 2092–2099.

Clarke, E., Emerson, E., Jha, S., & Sistla, A. (1998). Symmetry reductions in model checking. In *International Conference on Computer Aided Verification*, pp. 147–158. Springer.

Clarke, E., Grumberg, O., Jha, S., Lu, Y., & Veith, H. (2001). Progress on the state explosion problem in model checking. In *Informatics*, pp. 176–194. Springer.

Cox, I., & Hingorani, S. (1996). An efficient implementation of Reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking. *IEEE Transactions on pattern analysis and machine intelligence*, *18*(2), 138–150.

Dal, G. H., & Lucas, P. J. (2017). Weighted positive binary decision diagrams for exact probabilistic inference. *International Journal of Approximate Reasoning*, *90*, 411–432.

Dalvi, N., Schnaitter, K., & Suciu, D. (2010). Computing query probability with incidence algebras. In *Proceedings of the Twenty-Ninth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pp. 203–214. ACM.

Dalvi, N., & Suciu, D. (2007). Efficient query evaluation on probabilistic databases. *The International Journal on Very Large Data Bases*, *16*(4), 523–544.

Darwiche, A. (2001). Recursive Conditioning. *Artificial Intelligence*, *126*(1-2), 5–41.

Darwiche, A. (2009). *Modeling and Reasoning with Bayesian Networks*. Cambridge University Press.

Darwiche, A., & Marquis, P. (2002). A knowledge compilation map. *Journal of Artificial Intelligence Research*, *17*(1), 229–264.

Das, M., Wu, Y., Khot, T., Kersting, K., & Natarajan, S. (2016). Scaling lifted probabilistic inference and learning via graph databases. In *Proceedings of the 16th SIAM International Conference on Data Mining 2016*, pp. 738–746.

De Raedt, L., Kersting, K., Natarajan, S., & Poole, D. (2016). Statistical relational artificial intelligence: Logic, probability, and computation. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, *10*, 1–189.

de Salvo Braz, R., Amir, E., & Roth, D. (2005). Lifted first-order probabilistic inference. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, pp. 1319–1325.

de Salvo Braz, R., Amir, E., & Roth, D. (2006). MPE and partial inversion in lifted probabilistic variable elimination. In *Proceedings of the National Conference on Artificial Intelligence*, Vol. 2, pp. 1123–1130.

de Salvo Braz, R., Natarajan, S., Bui, H., Shavlik, J., & Russell, S. (2009). Anytime lifted belief propagation. In *International Workshop on Statistical Relational Learning*, Vol. 9.

Dean, T., & Givan, R. (1997). Model minimization in Markov decision processes. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence and Ninth Conference on Innovative Applications of Artificial Intelligence*, pp. 106–111.

Domingos, P., & Webb, W. (2012). A tractable first-order probabilistic logic. In *Proceedings of the National Conference on Artificial Intelligence*, Vol. 3, pp. 1902–1909.

Doucet, A., de Freitas, N., & Gordon, N. (2001). *Sequential Monte Carlo Methods in Practice*. Springer-Verlag, New York.

Doucet, A., De Freitas, N., Murphy, K., & Russell, S. (2000). Rao-Blackwellised particle filtering for dynamic Bayesian networks. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pp. 176–183. Morgan Kaufmann Publishers Inc.

Dylla, M., Miliaraki, I., & Theobald, M. (2013). Top-k query processing in probabilistic databases with non-materialized views. In *29th International Conference on Data Engineering*, pp. 122–133. IEEE.

Fagin, R. (1983). Degrees of acyclicity for hypergraphs and relational database schemes. *Journal of the ACM*, *30*(3), 514–550.

Fierens, D. (2010). Context-specific independence in directed relational probabilistic models and its influence on the efficiency of Gibbs sampling. *Frontiers in Artificial Intelligence and Applications*, *215*, 243–248.

Fortmann, T., Bar-Shalom, Y., & Scheffe, M. (1983). Sonar tracking of multiple targets using joint probabilistic data association. *IEEE journal of Oceanic Engineering*, *8*(3), 173–184.

Fox, V., Hightower, J., Liao, L., Schulz, D., & Borriello, G. (2003). Bayesian filtering for location estimation. *IEEE pervasive computing*, *2*(3), 24–33.

Geier, T., & Biundo, S. (2011). Approximate online inference for dynamic markov logic networks. In *23rd IEEE International Conference on Tools with Artificial Intelligence*, pp. 764–768. IEEE.

Gens, R., & Domingos, P. (2013). Learning the structure of sum-product networks. In *International Conference on Machine Learning*, pp. 873–880.

Getoor, L., Friedman, N., Koller, D., & Taskar, B. (2002). Learning Probabilistic Models of Link Structure. *Journal of Machine Learning Research*, *3*, 29.

Getoor, L., & Taskar, B. (2007). *Introduction to Statistical Relational Learning*. MIT press.

Givan, R., Dean, T., & Greig, M. (2003). Equivalence notions and model minimization in Markov decision processes. *Artificial Intelligence*, *147*(1-2), 163–223.

Glass, M., & Barker, K. (2012). Focused grounding for markov logic networks. In *Proceedings of the 25th International Florida Artificial Intelligence Research Society Conference*, pp. 531–536.

Godefroid, P., Van Leeuwen, J., Hartmanis, J., Goos, G., & Wolper, P. (1996). *Partial-Order Methods for the Verification of Concurrent Systems: An Approach to the State-Explosion Problem*, Vol. 1032. Springer Heidelberg.

Gogate, V., & Domingos, P. (2010). Exploiting logical structure in lifted probabilistic inference. In *AAAI Workshop - Technical Report*, Vol. WS-10-06, pp. 19–25.

Gogate, V., & Domingos, P. (2016). Probabilistic Theorem Proving. *Commun. ACM*, *59*(7), 107–115.

Gogate, V., Jha, A., & Venugopal, D. (2012). Advances in Lifted Importance Sampling.. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*.

Gutmann, B., Thon, I., & De Raedt, L. (2011). Learning the Parameters of Probabilistic Logic Programs from Interpretations. In *Machine Learning and Knowledge Discovery in Databases*, Vol. 6911, pp. 581–596. Springer Berlin Heidelberg, Berlin, Heidelberg.

Hadiji, F., Ahmadi, B., & Kersting, K. (2011). Efficient sequential clamping for lifted message passing. In *Lecture Notes in Computer Science*, Vol. 7006 LNAI, pp. 122–133.

Hadiji, F., & Kersting, K. (2013). Reduce and re-lift: Bootstrapped lifted likelihood maximization for MAP. In *AAAI Workshop - Technical Report*, Vol. WS-13-16, pp. 8–14.

Hajishirzi, H., & Amir, E. (2008). Sampling First Order Logical Particles. In *Proceedings of the Twenty-Fourth Conference on Uncertainty in Artificial Intelligence*, UAI'08, pp. 248–255, Helsinki, Finland. AUAI Press.

Han, M., Xu, W., Tao, H., & Gong, Y. (2004). An algorithm for multiple object trajectory tracking. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol. 1, pp. I–I. IEEE.

Hanebeck, U., & Baum, M. (2015). Association-free direct filtering of multi-target random finite sets with set distance measures. In *18th International Conference on Information Fusion*, pp. 1367–1374. IEEE.

Hölldobler, S., & Skvortsova, O. (2004). A logic-based approach to dynamic programming. In *Proceedings of the Workshop on "Learning and Planning in Markov Processes–Advances and Challenges" at the Nineteenth National Conference on Artificial Intelligence*, pp. 31–36.

Holte, R., & Fan, G. (2015). State Space Abstraction in Artificial Intelligence and Operations Research. In *Workshops at the Twenty-Ninth AAAI Conference on Artificial Intelligence*.

Huang, J., Guestrin, C., & Guibas, L. (2009a). Efficient inference for distributions on permutations. In *Advances in Neural Information Processing Systems*.

Huang, J., Guestrin, C., & Guibas, L. (2009b). Fourier Theoretic Probabilistic Inference over Permutations. *Journal of Machine Learning Research*, *10*, 997–1070.

Huang, J., Guestrin, C., Jiang, X., & Guibas, L. (2009c). Exploiting Probabilistic Independence for Permutations.. In *International Conference on Artificial Intelligence and Statistics*, pp. 248–255.

Jaeger, M., & Van den Broeck, G. (2012). Liftability of Probabilistic Inference: Upper and Lower Bounds. In *Proceedings of StarAI*.

Jaeger, M. (2000). On the complexity of inference about probabilistic relational models. *Artificial Intelligence*, pp. 297–308.

Jaeger, M. (2004). Probabilistic decision graphs—combining verification and AI techniques for probabilistic inference. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, *12*(supp01), 19–42.

Jagabathula, S., & Shah, D. (2011). Inferring rankings using constrained sensing. *IEEE Transactions on Information Theory*, *57*(11), 7288–7306.

Jaimovich, A., Meshi, O., & Friedman, N. (2007). Template based inference in symmetric relational Markov random fields. In *Proceedings of the Twenty-Third Conference on Uncertainty in Artificial Intelligence*.

Jha, A., Gogate, V., Meliou, A., & Suciu, D. (2010). Lifted inference seen from the other side: The tractable features. In *24th Annual Conference on Neural Information Processing Systems*.

Jha, A., & Suciu, D. (2012). Probabilistic databases with MarkoViews. *Proceedings of the VLDB Endowment*, *5*(11), 1160–1171.

Jiang, X., Huang, J., & Guibas, L. (2011). Fourier-information duality in the identity management problem. In *Lecture Notes in Computer Science*, Vol. 6912 LNAI, pp. 97–113.

Kang, B., & Kim, K. (2012). Exploiting symmetries for single- and multi-agent Partially Observable Stochastic Domains. *Artificial Intelligence*, *182–183*, 32 – 57.

Kazemi, S., Kimmig, A., Van den Broeck, G., & Poole, D. (2016). New liftable classes for first-order probabilistic inference. In *Advances in Neural Information Processing Systems*, pp. 3117–3125.

Kazemi, S., Kimmig, A., Van Den Broeck, G., & Poole, D. (2017). Domain Recursion for Lifted Inference with Existential Quantifiers. In *arXiv Preprint arXiv:1707.07763*.

Kazemi, S., & Poole, D. (2014). Elimination ordering in lifted first-order probabilistic inference. In *Proceedings of the National Conference on Artificial Intelligence*, Vol. 2, pp. 863–870.

Kersting, K. (2012). Lifted Probabilistic Inference. In *Proceedings of the 20th European Conference on Artificial Intelligence*, Vol. 242, pp. 33–38. IOS Press.

Kersting, K., Ahmadi, B., & Natarajan, S. (2009). Counting belief propagation. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, pp. 277–284.

Kersting, K., & De Raedt, L. (2001). Adaptive Bayesian Logic Programs. In *Inductive Logic Programming*, Vol. 2157, pp. 104–117. Springer Berlin Heidelberg, Berlin, Heidelberg.

Kersting, K., & De Raedt, L. (2007). *1 Bayesian Logic Programming: Theory and Tool*. MIT Press.

Kersting, K., De Raedt, L., & Raiko, T. (2006). Logical Hidden Markov Models.. *Journal of Artificial Intelligence Research*, *25*, 425–456.

Kersting, K., El Massaoudi, Y., Hadiji, F., & Ahmadi, B. (2010). Informed Lifting for Message-Passing.. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*.

Kersting, K., Van Otterlo, M., & De Raedt, L. (2004). Bellman goes relational. In *Proceedings of the Twenty-First International Conference on Machine Learning*, p. 59. ACM.

Khardon, R., & Sanner, S. (2017). Stochastic planning and lifted inference. *arXiv preprint, arXiv:1701.01048*.

Khot, T., Natarajan, S., Kersting, K., & Shavlik, J. (2011). Learning Markov Logic Networks via Functional Gradient Boosting. In *2011 IEEE 11th International Conference on Data Mining*, pp. 320–329, Vancouver, BC, Canada. IEEE.

Kiddon, C., & Domingos, P. (2010). Leveraging ontologies for lifted probabilistic inference and learning. In *AAAI Workshop - Technical Report*, Vol. WS-10-06, pp. 40–45.

Kiddon, C., & Domingos, P. (2011). Coarse-to-fine inference and learning for first-order probabilistic models. In *Proceedings of the National Conference on Artificial Intelligence*, Vol. 2, pp. 1049–1056.

Kimmig, A., Mihalkova, L., & Getoor, L. (2015). Lifted graphical models: A survey. *Machine Learning*, *99*(1), 1–45.

Kisyński, J., & Poole, D. (2009a). Constraint processing in lifted probabilistic inference. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, pp. 293–302.

Kisyński, J., & Poole, D. (2009b). Lifted aggregation in directed first-order probabilistic models. In *Proceedings of the Twenty-First International Joint Conference on Artificial Intelligence*, pp. 1922–1929.

Kitchenham, B. (2004). Procedures for Performing Systematic Reviews. In *Keele University Technical Report TR/SE-0401*.

Koller, D., & Friedman, N. (2009). *Probabilistic Graphical Models: Principles and Techniques*. MIT press.

Koller, D., & Pfeffer, A. (1997). Object-oriented Bayesian networks. In *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence*, pp. 302–313. Morgan Kaufmann Publishers Inc.

Kondor, R., Howard, A., & Jebara, T. (2007). Multi-object tracking with representations of the symmetric group. *Journal of Machine Learning Research*, *2*, 211–218.

Krishnamurthy, E., Murthy, V., & Krishnamurthy, V. (2004). Biologically inspired rule-based multiset programming paradigm for soft-computing. In *Proceedings of the 1st Conference on Computing Frontiers*, pp. 140–149.

Kwiatkowska, M., Norman, G., & Parker, D. (2006). Symmetry reduction for probabilistic model checking. In *Lecture Notes in Computer Science*, Vol. 4144 LNCS, pp. 234–248.

Leven, W., & Lanterman, A. (2004). Multiple target tracking with symmetric measurement equations using unscented Kalman and particle filters. In *Proceedings of the Thirty-Sixth Southeastern Symposium on System Theory*, pp. 195–199. IEEE.

Leven, W., & Lanterman, A. (2009). Unscented Kalman Filters for Multiple Target Tracking With Symmetric Measurement Equations. *IEEE Transactions on Automatic Control*, *54*(2), 370–375.

Mahler, R. (2003). Multitarget Bayes filtering via first-order multitarget moments. *IEEE Transactions on Aerospace and Electronic systems*, *39*(4), 1152–1178.

Marinescu, R., Dechter, R., & Ihler, A. (2015). Pushing Forward Marginal MAP with Best-First Search. In *Proceedings of the International Joint Conference on Artificial Intelligence*.

Mateus, P., Pacheco, A., & Pinto, J. (2002). Observations and the probabilistic situation calculus. In *Proceedings of the Eighth International Conference on Principles of Knowledge Representation and Reasoning*, pp. 327–340.

Mateus, P., Pacheco, A., Pinto, J., Sernadas, A., & Sernadas, C. (2001). Probabilistic situation calculus. *Annals of Mathematics and Artificial Intelligence*, *32*(1), 393–431.

Maus, C., Rybacki, S., & Uhrmacher, A. (2011). Rule-based multi-level modeling of cell biological systems. *BMC Systems Biology*, *5*(1), 166.

Meert, W., Van Den Broeck, G., & Darwiche, A. (2014). Lifted Inference for Probabilistic Logic Programs. In *Workshop on Probabilistic Logic Programming*.

Milch, B., Zettlemoyer, L., Kersting, K., Haimes, M., & Kaelbling, L. (2008). Lifted probabilistic inference with counting formulas. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence*, pp. 1062–1068.

Mittal, H., Goyal, P., Gogate, V., & Singla, P. (2014). New rules for domain independent lifted MAP inference. In *Advances in Neural Information Processing Systems*, pp. 649–657.

Mladenov, M., Ahmadi, B., & Kersting, K. (2012). Lifted Linear Programming.. In *International Conference on Artificial Intelligence and Statistics*, pp. 788–797.

Mladenov, M., Globerson, A., & Kersting, K. (2014a). Efficient lifting of MAP LP relaxations using k-locality. *Journal of Machine Learning Research*, *33*, 623–632.

Mladenov, M., Globerson, A., & Kersting, K. (2014b). Lifted Message Passing as Reparametrization of Graphical Models.. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, pp. 603–612.

Mladenov, M., & Kersting, K. (2013). Lifted inference via k-locality. In *AAAI Workshop - Technical Report*, Vol. WS-13-16, pp. 25–30.

Moher, D., Liberati, A., Tetzlaff, J., Altman, D., Prisma Group, & others (2009). Preferred reporting items for systematic reviews and meta-analyses: The PRISMA statement. *PLoS med*, *6*(7), e1000097.

Natarajan, S., Bui, H., Tadepalli, P., Kersting, K., & Wong, W. (2008). Logical Hierarchical Hidden Markov models for modeling user activities. In *Lecture Notes in Computer Science*, Vol. 5194 LNAI, pp. 192–209.

Nath, A., & Domingos, P. (2010a). Efficient Belief Propagation for Utility Maximization and Repeated Inference.. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, Vol. 4, p. 3.

Nath, A., & Domingos, P. (2010b). Efficient lifting for online probabilistic inference. In *AAAI Workshop - Technical Report*, Vol. WS-10-06, pp. 64–69.

Ng, K., & Lloyd, J. (2009). Probabilistic reasoning in a classical logic. *Journal of Applied Logic*, *7*(2), 218–238.

Ng, K., Lloyd, J., & Uther, W. (2008). Probabilistic modelling, inference and learning using logical theories. *Annals of Mathematics and Artificial Intelligence*, *54*(1-3), 159–205.

Niepert, M. (2012). Markov chains on orbits of permutation groups. In *Uncertainty in Artificial Intelligence - Proceedings of the 28th Conference*, pp. 624–633.

Niepert, M. (2013). Symmetry-aware marginal density estimation. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence*, pp. 725–731.

Niepert, M., & Van den Broeck, G. (2014). Tractability through exchangeability: A new perspective on efficient probabilistic inference. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*, Québec City, Québec, Canada.

Nitti, D., De Laet, T., & De Raedt, L. (2013). A particle filter for hybrid relational domains. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2764–2771. IEEE.

Nitti, D., De Laet, T., & De Raedt, L. (2014). Relational object tracking and learning. In *IEEE International Conference on Robotics and Automation*, pp. 935–942.

Nitti, D., De Laet, T., & De Raedt, L. (2016). Probabilistic logic programming for hybrid relational domains. *Machine Learning*, *103*(3), 1–43.

Oh, S., Russell, S., & Sastry, S. (2004). Markov chain Monte Carlo data association for general multiple-target tracking problems. In *Proceedings of the 43rd IEEE Conference on Decision and Control*, Vol. 1, pp. 735–742. IEEE.

Oury, N., & Plotkin, G. (2013). Multi-level modelling via stochastic multi-level multiset rewriting. *Mathematical Structures in Computer Science*, *23*(02), 471–503.

Papai, T., Kautz, H., & Stefankovic, D. (2012). Slice normalized dynamic markov logic networks. In *Advances in Neural Information Processing Systems*, pp. 1907–1915.

Paun, G. (2012). *Membrane Computing: An Introduction.* Springer Science & Business Media.

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

Peled, D. (1993). All from one, one for all: On model checking using representatives. In *International Conference on Computer Aided Verification*, pp. 409–423. Springer.

Pescini, D., Besozzi, D., Mauri, G., & Zandron, C. (2006). Dynamical probabilistic P systems. *International Journal of Foundations of Computer Science*, *17*(01), 183–204.

Poole, D. (2003). First-order probabilistic inference. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, pp. 985–991.

Poole, D., Bacchus, F., & Kisynski, J. (2011). Towards completely lifted search-based probabilistic inference. *arXiv preprint*, *arXiv:1107.4035*.

Poon, H., Domingos, P., & 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*, Vol. 8, pp. 1075–1080.

Puterman, M. (2014). *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons.

Reid, D. (1979). An Algorithm for Tracking Multiple Targets. *IEEE Transactions on Automatic Control*, *24*(6).

Reiter, R. (1991). The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression. In *Artificial Intelligence and Mathematical Theory of Computation*, pp. 359–380. Academic Press Professional, Inc.

Richardson, M., & Domingos, P. (2006). Markov logic networks. *Machine Learning*, *62*(1-2 SPEC. ISS.), 107–136.

Sanner, S., & Abbasnejad, E. (2012). Symbolic Variable Elimination for Discrete and Continuous Graphical Models. In *AAAI*.

Sanner, S., & Boutilier, C. (2009). Practical solution techniques for first-order MDPs. *Artificial Intelligence*, *173*(5-6), 748–788.

Sanner, S., & McAllester, D. (2005). Affine algebraic decision diagrams (AADDs) and their application to structured probabilistic inference. In *IJCAI*, Vol. 2005, pp. 1384–1390.

Sarkhel, S., & Gogate, V. (2013). Lifting WALKSAT-based local search algorithms for MAP inference. In *AAAI Workshop - Technical Report*, Vol. WS-13-16, pp. 64–67.

Sarkhel, S., Venugopal, D., Singla, P., & Gogate, V. (2014). Lifted MAP Inference for Markov Logic Networks.. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*, pp. 859–867.

Särkkä, S. (2013). *Bayesian Filtering and Smoothing*, Vol. 3. Cambridge University Press.

Sato, T., & Kameya, Y. (2001). Parameter Learning of Logic Programs for Symbolic-Statistical Modeling. *Journal of Artificial Intelligence Research*, *15*, 391–454.

Schumitsch, B., Thrun, S., Bradski, G., & Olukotun, K. (2005). The information-form data association filter. In *NIPS*, pp. 1193–1200.

Sen, P., Deshpande, A., & Getoor, L. (2008). Exploiting shared correlations in probabilistic databases. *Proceedings of the VLDB Endowment*, *1*(1), 809–820.

Sen, P., Deshpande, A., & Getoor, L. (2009). Bisimulation-based approximate lifted inference. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pp. 496–505. AUAI Press.

Shenoy, P., & West, J. (2011). Inference in hybrid Bayesian networks using mixtures of polynomials. *International Journal of Approximate Reasoning*, *52*(5), 641–657.

Singla, P., & Domingos, P. (2006). Memory-efficient inference in relational domains. In *Proceedings of the 21st National Conference on Artificial Intelligence*, Vol. 6, pp. 488–493.

Singla, P., & Domingos, P. (2008). Lifted first-order belief propagation. In *Proceedings of the National Conference on Artificial Intelligence*, Vol. 2, pp. 1094–1099.

Singla, P., Nath, A., & Domingos, P. (2010). Approximate Lifted Belief Propagation.. In *AAAI Workshop - Technical Report*, pp. 92–97.

Singla, P., Nath, A., & Domingos, P. (2014). Approximate Lifting Techniques for Belief Propagation.. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, pp. 2497–2504.

Suciu, D., Olteanu, D., Ré, C., & Koch, C. (2011). Probabilistic Databases. *Synthesis Lectures on Data Management*, *3*(2), 1–180.

Taghipour, N., Davis, J., & Blockeel, H. (2014). Generalized counting for lifted variable elimination. In *23rd International Conference on Inductive Logic Programming*, Vol. 8812, pp. 107–122.

Taghipour, N., Fierens, D., Davis, J., & Blockeel, H. (2012). Lifted variable elimination with arbitrary constraints. *Journal of Machine Learning Research*, *22*, 1194–1202.

Taghipour, N., Fierens, D., Davis, J., & Blockeel, H. (2013a). Lifted variable elimination: Decoupling the operators from the constraint language. *Journal of Artificial Intelligence Research*, *47*, 393–439.

Taghipour, N., Fierens, D., Van Den Broeck, G., Davis, J., & Blockeel, H. (2013b). Completeness results for lifted variable elimination. In *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*.

Taghipour, N., Fierens, D., Van Den Broeck, G., Davis, J., & Blockeel, H. (2013c). On the completeness of lifted variable elimination. In *AAAI Workshop - Technical Report*, Vol. WS-13-16, pp. 74–80.

Takiyama, F., & Cozman, F. (2014). Inference with Aggregation Parfactors: Lifted Elimination with First-Order d-Separation. In *Proceedings of the Brazilian Conference on Intelligent Systems*, pp. 384–389.

Thon, I., Landwehr, N., & De Raedt, L. (2011). Stochastic relational processes: Efficient inference and applications. *Machine Learning*, *82*(2), 239–272.

Torti, L., Wuillemin, P., & Gonzales, C. (2010). Reinforcing the object-oriented aspect of probabilistic relational models. In *European Workshop on Probabilistic Graphical Models*, pp. 273–280.

Toussaint, M., & Storkey, A. (2006). Probabilistic inference for solving discrete and continuous state Markov Decision Processes. In *Proceedings of the 23rd International Conference on Machine Learning*, pp. 945–952. ACM.

Valmari, A. (1989). Stubborn sets for reduced state space generation. In *International Conference on Application and Theory of Petri Nets*, pp. 491–515. Springer.

Van Den Broeck, G. (2011). On the completeness of first-order knowledge compilation for lifted probabilistic inference. In *25th Annual Conference on Neural Information Processing Systems*.

Van Den Broeck, G., Choi, A., & Darwiche, A. (2012). Lifted relax, compensate and then recover: From approximate to exact lifted probabilistic inference. In *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence*, pp. 131–141.

Van Den Broeck, G., & Darwiche, A. (2013). On the complexity and approximation of binary evidence in lifted inference. In *Advances in Neural Information Processing Systems*, pp. 2868–2876.

Van Den Broeck, G., & Davis, J. (2012). Conditioning in first-order knowledge compilation and lifted probabilistic inference. In *Proceedings of the National Conference on Artificial Intelligence*, Vol. 3, pp. 1961–1967.

Van Den Broeck, G., Meert, W., & Darwiche, A. (2014). Skolemization for weighted first-order model counting. In *Proceedings of the 14th International Conference on Principles of Knowledge Representation and Reasoning*.

Van Den Broeck, G., & Niepert, M. (2015). Lifted probabilistic inference for asymmetric graphical models. In *Proceedings of the National Conference on Artificial Intelligence*, Vol. 5, pp. 3599–3605.

Van Den Broeck, G., Taghipour, N., Meert, W., Davis, J., & De Raedt, L. (2011). Lifted probabilistic inference by first-order knowledge compilation. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*, pp. 2178–2185.

Van Haaren, J., b Van den Broeck, G., Meert, W., & Davis, J. (2016). Lifted generative learning of Markov logic networks. *Machine Learning*, *103*(1), 27–55.

Venugopal, D., & Gogate, V. (2012). On lifting the Gibbs sampling algorithm. In *Advances in Neural Information Processing Systems*, Vol. 3, pp. 1655–1663.

Venugopal, D., & Gogate, V. (2014a). Evidence-based clustering for scalable inference in Markov logic. In *Lecture Notes in Computer Science*, Vol. 8726 LNAI, pp. 258–273.

Venugopal, D., & Gogate, V. (2014b). Scaling-up importance sampling for Markov logic networks. In *Advances in Neural Information Processing*, Vol. 4, pp. 2978–2986.

Venugopal, D., Sarkhel, S., & Cherry, K. (2016). Non-parametric domain approximation for scalable Gibbs sampling in MLNs. In *32nd Conference on Uncertainty in Artificial Intelligence*, pp. 745–754.

Venugopal, D., Sarkhel, S., & Gogate, V. (2015). Just count the satisfied groundings: Scalable local-search and sampling based inference in MLNs. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, Vol. 5, pp. 3606–3612.

Vlasselaer, J., Kimmig, A., Dries, A., Meert, W., & De Raedt, L. (2016). Knowledge Compilation and Weighted Model Counting for Inference in Probabilistic Logic Programs. In *Workshops at the Thirtieth AAAI Conference on Artificial Intelligence*.

Wang, C., Joshi, S., & Khardon, R. (2008). First order decision diagrams for relational MDPs. *Journal of Artificial Intelligence Research*, *31*, 431–472.

Warnke, T., Helms, T., & Uhrmacher, A. (2015). Syntax and Semantics of a Multi-Level Modeling Language. In *Proceedings of the 3rd ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*, pp. 133–144. ACM Press.

Weiss, Y. (2000). Correctness of local probability propagation in graphical models with loops. *Neural computation*, *12*(1), 1–41.

Yue, S., Jiao, P., Zha, Y., & Yin, Q. (2015a). A Logical Hierarchical Hidden Semi-Markov Model for team intention recognition. *Discrete Dynamics in Nature and Society*, *2015*.

Yue, S., Xu, K., Qin, L., & Yin, Q. (2015b). Filtering states with partial observations for the Logical hidden Markov model. In *IEEE International Conference on Mechatronics and Automation*, pp. 65–69.

Zettlemoyer, L., Pasula, H., & Kaelbling, L. (2008). Logical particle filtering. In *Dagstuhl Seminar Proceedings*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik.

Zhang, N. L., & Poole, D. (1994). A simple approach to Bayesian network computations. In *Proceedings of the Tenth Canadian Conference on Artificial Intelligence*.