Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence

A Framework for Integration of Logical and Probabilistic Knowledge

Jingsong Wang and **Marco Valtorta**
Department of Computer Science and Engineering
University of South Carolina
Columbia, SC 29208, USA
{wang82, mgv}@cse.sc.edu

Abstract

Integrating the expressive power of first-order logic with the ability of probabilistic reasoning of Bayesian networks has attracted the interest of many researchers for decades. We present an approach to integration that translates logical knowledge into Bayesian networks and uses Bayesian network composition to build a uniform representation that supports both logical and probabilistic reasoning. In particular, we propose a new way of translation of logical knowledge, relation search. Through the use of the proposed framework, without learning new languages or tools, modelers are allowed to 1) specify special knowledge using the most suitable languages, while reasoning in a uniform engine; 2) make use of pre-existing logical knowledge bases for probabilistic reasoning (to complete the model or minimize potential inconsistencies).

Introduction

Bayesian networks (BNs) are commonly used for knowledge specification and reasoning in probabilistic domain. First-order logic (FOL) has been widely used over decades for logical reasoning in many fields. The integration problem has attracted the interest of many researchers and plenty of work have been proposed. In our framework, the user is assumed to use FOL formulas to specify logical knowledge stored in a knowledge base (KB), and a Bayesian network, which we call *Probabilistic Bayesian Network (PBN)*, to specify probabilistic knowledge. The basic idea is to convert related knowledge from the logical KB into a logical model represented through a Bayesian network, which we call *Logical Bayesian Network (LBN)*. Then the LBN is composed with the PBN. The final output of the composition is a Bayesian network, *Composite Bayesian Network (CBN)*, that integrates both the logical knowledge and the probabilistic knowledge the user specified. Figure 1 summarizes the process. By using an automatic conversion and composition approach, we relieve the users from the burden of learning a new formalism that integrates logic and probabilistic knowledge. Our approach is an alternative way of both designing new expert systems and making use of existing ones.

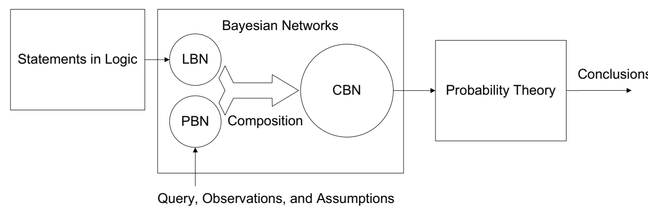

Figure 1: Reasoning systems with the combination of logic and Bayesian networks.

Logical Bayesian Network Generation

The most important part of our framework is the conversion of FOL knowledge into Bayesian network, which is based on a relation search algorithm. We present the algorithm in propositional case and extend it to FOL case.

Propositional Case Algorithm

The basic idea of relation search is very simple. We just look through the KB and extract all the logical formulas that are related to the probabilistic atoms. A formula has a *relation* with an atom if that atom appears directly or indirectly in this formula. Indirect appearance is defined the following way. If atoms A_1 and A_2 both appear in a formula R_1, and A_1 also appears in a formula R_2 while A_2 does not, we say that A_2 appears indirectly in the formula R_2. Algorithm 1 depicts the pseudocode for our relation search algorithm.

Modification of Algorithm for FOL Case

In the FOL case, instantiation and quantified formula node generation are needed to build the LBN. Even in the FOL case, queries, observations and assumptions should mostly be ground formulas. Thus we only consider ground atomic formulas from PBN and the set of such atoms is denoted by $\mathcal{P}$. We assume that all the formulas in KB are closed (i.e., no occurrence of free variables) and in Skolem form. Also, we assume that constants in KB are all the individuals in the domain, and we restrict the domain to be function-free.

1. We do the same search and find all the related formulas $\mathcal{R}$ from KB.

Algorithm 1 Relation Search

Require: the probabilistic atom set $\mathcal{P}$ from PBN containing query, observations, and assumptions, the set of logical formulas KB $= \{R_1, R_2, ..., R_z\}$, $z = |\text{KB}|$, and the sets $\mathcal{A}_1, \mathcal{A}_2, ..., \mathcal{A}_z$. $\mathcal{A}_i$ comprises all the atoms appearing in its corresponding logical formula $R_i \in$ KB, where $1 \leq i \leq z$.

$\mathcal{V} = \mathcal{P}$;
$\mathcal{E} = \emptyset$;
$\mathcal{L} = \emptyset$;
for $i = 1$ to z **do**
    Tag R_i as not visited;
end for
while $\mathtt{true}$ **do**
    $Changed \Leftarrow$ **false**;
    for $i = 1$ to z **do**
        if R_i is not visited **then**
            if $\mathcal{A}_i \cap \mathcal{V} \neq \emptyset$ **then**
                $\mathcal{V} = \mathcal{V} \cup \mathcal{A}_i \cup \{R_i\}$;
                for all $A \in \mathcal{A}_i$ **do**
                    $\mathcal{E} = \mathcal{E} \cup \{(A, R_i)\}$;
                end for
                Build CPT Θ_i for R_i based on its logical structure;
                $\mathcal{L} = \mathcal{L} \cup \{\Theta_i\}$;
                Tag R_i as visited;
                $Changed \Leftarrow \mathbf{true}$;
            end if
        end if
    end for
    if $Changed$ = **false then**
        $\mathbf{break}$;
    end if
end while
return $BN(\mathcal{V}, \mathcal{E}, \mathcal{L})$;

2. We use available constants in $\mathcal{R} \cup \mathcal{P}$ to get all the possible ground instantiations of quantified formulas in $\mathcal{R}$, and add these instantiations to $\mathcal{R}$. This new set is named $\mathcal{R}'$.

3. We build a Bayesian network based on $\mathcal{R}'$. We follow exactly the same procedure as in the propositional case for generating nodes, edges, and CPTs for the ground formulas in $\mathcal{R}'$. For a quantified formula, we put it as the child of its ground instantiations (groundings) plus an extra node O, which represents a proposition that all the other instantiations that are based on constants appearing in KB but not in $\mathcal{R}'$ hold. The CPT For such a quantified formula is an AND table, i.e., the value true has probability 1 if all parents are true and probability 0 otherwise.

Therefore for the FOL case, the generated Bayesian network will usually be a three level network if quantified formulas exist in $\mathcal{R}'$. The nodes corresponding to them are in the third level. One important change for the FOL case relation search output is an O node for each quantified formula as one additional parent.

Proof of Correctness

Theorem 1 *For a BN resulting from relation search,* $\mathbf{G} = (\mathcal{V}, \mathcal{E}, \mathcal{L})$*, for any* $U \in \mathcal{V}$ *and any* $\mathcal{V}' \subseteq \mathcal{V}$*, if* $\mathcal{V}' \models U$*, then* $\mathbf{Pr}(U = true|\mathcal{V}' = true) = 1$ *in* $\mathbf{G}$*, where* $\mathcal{V}'$ *= true means that all the nodes in* $\mathcal{V}'$ *of* $\mathbf{G}$ *are set to true.*

The proof uses weighted model counting (Darwiche 2009), over $\mathbf{G}$, to prove the joint probability of evidence $\mathbf{Pr(e)} = 0$, where $\mathbf{e} = \{U = false\} \cup \mathcal{V}' = true$, and then follow Bayes' rule to conclude the posterior probability $\mathbf{Pr}(U = true|\mathcal{V}' = true) = 1$.

Composition

The composition process of Bayesian networks (LBN and PBN) is supported by ontology (Huhns, Valtorta, and Wang 2010). Definitions of terms from an ontology is essential for combining the logical knowledge with the probabilistic one in our framework, as different definitions (of equality or inclusion) may result in different LBNs. Ontology also makes composition process much more flexible and adaptable, for example, the use of subclass relationship for more specific information analysis.

Conclusion

Most techniques for handling integration problem are designed completely as a new formalism such as BLOG (Milch et al. 2005) or an extension of either FOL such as MLN (Richardson and Domingos 2006) and SLP (Muggleton 1996) or Bayesian networks such as MEBN (Laskey 2008) and BLP (Kersting and De Raedt 2001). Users have to study new ways of modeling, a process that is hard and always comes with waste of effort as many techniques are still in the preliminary stage for practical use. We did not propose a new language, but a framework that can automatically and efficiently transform the users' knowledge expressed in these two traditional formalisms into one representation for probabilistic reasoning. We expect this framework to be easy to understand, simple to implement, and efficient to execute. Evaluation of this claim is a major topic of our ongoing work. Details and other related work including comparisons can be found at `http://www.cse.sc.edu/~wang82/doc/aaai2011`.

References

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

Huhns, M.; Valtorta, M.; and Wang, J. 2010. Design principles for ontological support of Bayesian evidence management. In Obrst, L.; Janssen, T.; and Ceusters, W., eds., *Semantic Technologies, Ontologies, and Information Sharing for Intelligence Analysis*. Amsterdam: IOS Press. 163–178.

Kersting, K., and De Raedt, L. 2001. Bayesian logic programs. *CoRR* cs.AI/0111058.

Laskey, K. B. 2008. MEBN: A language for first-order knowledge bases. *Artificial Intelligence* 172:140–178.

Milch, B.; Marthi, B.; Russell, S.; Sontag, D.; Ong, D. L.; and Kolobov, A. 2005. BLOG: Probabilistic models with unknown objects. In *In IJCAI*, 1352–1359.

Muggleton, S. 1996. Stochastic logic programs. In De Raedt, L., ed., *Advances in Inductive Logic Programming*. IOS Press. 254–264.

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