BayesL: a Logical Framework for Bayesian Networks (Extended Abstract)

Stefano M. Nicoletti
FMT, University of Twente
Enschede, the Netherlands
DiSPeA, University of Urbino
Urbino, Italy
s.m.nicoletti@utwente.nl

E. Moritz Hahn
FMT, University of Twente
Enschede, the Netherlands
e.m.hahn@utwente.nl

Mariëlle Stoelinga
FMT, University of Twente
Enschede, the Netherlands
Radboud University
Nijmegen, the Netherlands
m.i.a.stoelinga@utwente.nl

1 Introduction

Bayesian networks are a central model in artificial intelligence for reasoning under uncertainty [3, 4]. They combine a directed acyclic graph with conditional probability tables (CPTs) to encode probabilistic dependencies among random variables.

These characteristics have made BNs particularly attractive for safety–critical and mission–critical applications in aerospace and space operations, such as failure prediction and diagnosis in satellite earth terminals and communication infrastructures [2]. In such systems, thousands of interacting hardware and environmental factors – ranging from amplifiers, converters, and generators to temperature and power conditions – must be monitored continuously, and operators rely on probabilistic models to anticipate faults, prioritise maintenance, and support rapid diagnosis from partial and noisy observations.

Motivation. Despite their expressive power and practical success, practitioners still face significant challenges in *guaranteeing that a BN behaves as intended*, particularly when networks are inferred from data [5]. Although standard BN toolkits support inference and learning [5], they provide little support for verifying high-level behavioural guarantees. Users often ask:

- Does observing variable X consistently increase the probability of outcome Y?
- Are two variables conditionally independent given certain evidence?
- How does a prediction change if a CPT entry were modified?
- Is some safety or reliability threshold always respected?

Answering such questions typically requires manual modifications of the model or ad-hoc experimentation. These approaches are error-prone and offer no formal guarantees. Inspired by probabilistic model checking [4], we seek to provide BNs with a logic in which such expectations can be stated precisely and checked automatically.

Our contribution. We introduce *BayesL*, a logical framework for specifying and verifying formal properties of BN behaviour. BayesL supports expressive probabilistic queries, reasoning about structural dependencies, and hypothetical updates to explore "what-if" scenarios. Its associated model checking procedures automatically analyse whether a BN satisfies such specifications.

Submitted to:
MISSION Workshop 2026

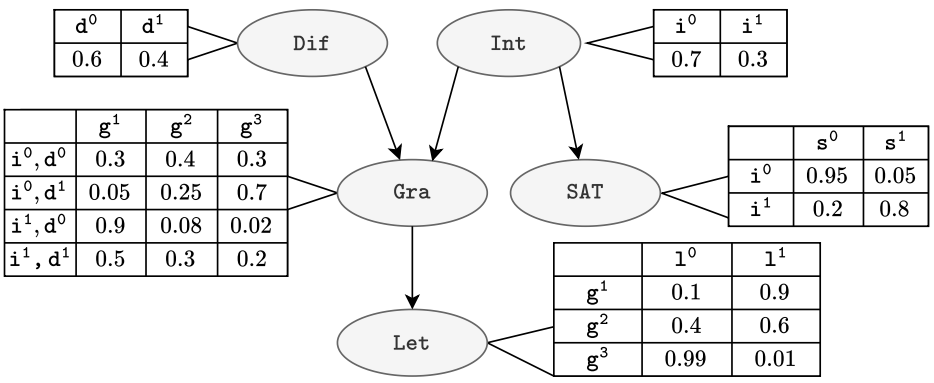

Figure 1: BN from [3] representing dependencies between the quality of a recommendation letter, the SAT score, the grade obtained in a course, the difficulty of that course and the intelligence of a student. The image includes conditional probability tables (CPTs) for every node in the BN.

Example 1. Figure 1 represents a well-known BN for the student example from [3]. This BN encodes dependencies between five random variables: the student's intelligence (`Int`), the course difficulty (`Dif`), the grade (`Gra`), the student's SAT score (`SAT`), and the quality of the recommendation letter they obtain (`Let`). All of the variables except `Gra` are binary-valued, and `Gra` is ternary-valued (i.e., grade is either *high*, *medium* or *low*). Figure 1 also represents the CPTs of each vertex representing said variables: the CPT of vertex v defines a probability distribution which determines the evaluation of v, given some evaluation of its parent nodes [4]. For example, according to the CPT of *Grade*, the probability of getting a *medium* grade ($\texttt{Gra} = \texttt{medium}$) given a *difficult* exam ($\texttt{Dif} = \texttt{high}$) and an *intelligent* student ($\texttt{Int} = \texttt{high}$) is 0.3.

2 Overview of BayesL

BayesL is a *Bayesian Network Logic* designed around three main objectives:

1. expressive querying of probabilistic and structural properties;
2. internalised what-if reasoning via CPT updates;
3. automatic verification through model checking.

Atomic events. BayesL supports atomic propositions of the form $X \bowtie x$, stating that a variable X takes a value x (or satisfies an inequality). Boolean connectives enable non-standard comparisons such as $(\texttt{Gra} < \texttt{high}) \vee (\texttt{Dif} = \texttt{easy})$.

Probabilistic inference. The logic includes marginal and conditional probability queries, marginal MAP queries, and most probable explanation (MPE) inference, following standard BN semantics [3]. With a dedicated CPT-update operator (using square brackets) we can perform local hypothetical modifications within queries.

Probabilistic constraints. Results of inference queries can be compared against thresholds and combined logically, yielding specifications such as

$$\mathrm{P}(\texttt{Let} = \texttt{strong} \mid \texttt{Gra} = \texttt{high}) \geq 0.8$$

Structural reasoning. BayesL further incorporates operators for probabilistic influence and conditional independence based on d-separation [3]. These structural properties can be freely combined with probabilistic constraints.

2.1 BayesL Queries

Following, we illustrate BayesL via some example queries for the BN in Figure 1. These examples illustrate how BayesL supports querying a BN at multiple levels (incl. non-standard ones):

1. **What-if/CPT update.** After updating the probability of a *high Grade* given *high Intelligence* and a *difficult* course to 0.9, does the probability of a *strong Letter* improve beyond 0.7?

$$\mathrm{P}(\texttt{Let} = \texttt{strong} \mid \texttt{Gra} = \texttt{high}) \geq 0.7\ [\texttt{Gra} = \texttt{high} \mid$$

$$\texttt{Int} = \texttt{high}, \texttt{Dif} = \texttt{high} \mapsto 0.9]$$

2. **Marginal Maximum A Posteriori query (MAP).** What is the most probable assignment to *Intelligence* and *SAT* given a *strong Letter*?

$$\mathrm{MAP}(\texttt{Int}, \texttt{SAT} \mid \texttt{Let} = \texttt{strong})$$

3. **Most Probable Explanation query (MPE).** What is the most probable complete explanation for the observed *strong Letter*?

$$\mathrm{MPE}(\texttt{Let} = \texttt{strong})$$

4. **Evidential reasoning.** Given that the *Letter* is *weak*, how likely is it that the course was *difficult*?

$$\mathrm{P}(\texttt{Dif} = \texttt{high} \mid \texttt{Let} = \texttt{weak})$$

5. **Causal reasoning with disjunction.** How likely is a *strong* recommendation *Letter* because of a *high SAT* score or a *high Grade*?

$$\mathrm{P}(\texttt{Let} = \texttt{strong} \mid \texttt{SAT} = \texttt{good} \vee \texttt{Gra} = \texttt{high})$$

6. **Non-standard comparison.** What is the probability that the student obtains a *Grade* lower than *high* or that the course was *easy*? (assuming an order $low < medium < high$).

$$\mathrm{P}(\texttt{Gra} < \texttt{high} \vee \texttt{Dif} = \texttt{low})$$

7. **Complex reasoning with threshold check.** Is it true that the probability of a *strong Letter* given a *high Grade* is at least 0.8 and that *Intelligence* and *Letter* are independent given *Grade*?

$$\mathrm{P}(\texttt{Let}{=}\texttt{strong} \mid \texttt{Gra}{=}\texttt{high}) \geq 0.8 \wedge \mathrm{IDP}(\texttt{Int}, \texttt{Let} \mid \texttt{Gra})$$

3 Model Checking: a Sketch

We formalise the semantics of BayesL by evaluating formulae on a BN and returning probabilities, truth values, or optimal assignments. Direct enumeration of all joint assignments is infeasible for realistic networks: our model checking algorithms instead recursively reduce BayesL formulae to standard BN inference problems, leveraging existing algorithms for conditional probabilities, MAP, and MPE [3]. CPT updates are realised by constructing modified networks on-the-fly, while Boolean combinations of atomic events are encoded via auxiliary vertices. This allows BayesL to act as a high-level specification language that conveniently re-uses existing BN inference engines.

4 Conclusion and Outlook

BayesL provides a principled framework for the formal specification and verification of Bayesian networks. By unifying probabilistic inference, structural reasoning, and hypothetical CPT updates, it enables white-box analysis of BN behaviour and strengthens trust in probabilistic (AI) systems.

Future work will focus on implementing a full BayesL model checker and exploring constraint-guided synthesis and learning of BNs, extending ideas from probabilistic verification [4] and program synthesis [1] to the automated construction of models satisfying BayesL logical requirements.

References

[1] Roman Andriushchenko, Milan Češka, Sebastian Junges, Joost-Pieter Katoen & Šimon Stupinskỳ (2021): *PAYNT: a tool for inductive synthesis of probabilistic programs*. In: *CAV*, Springer, pp. 856–869.

[2] Steven Bottone, Daniel Lee, Michael O'Sullivan & Mark Spivack (2008): *Failure prediction and diagnosis for satellite monitoring systems using Bayesian networks*. In: *MILCOM 2008-2008 IEEE Military Communications Conference*, IEEE, pp. 1–7.

[3] Daphne Koller & Nir Friedman (2009): *Probabilistic graphical models: principles and techniques*. MIT press.

[4] Bahare Salmani & Joost-Pieter Katoen (2020): *Bayesian inference by symbolic model checking*. In: *QEST 2020*, Springer, pp. 115–133.

[5] Marco Scutari (2010): *Learning Bayesian networks with the bnlearn R package*. *JSS* 35, pp. 1–22.