The Descriptive Complexity of Bayesian Network Specifications

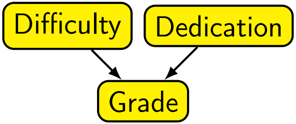
Fabio G. Cozman and Denis D. Mauá
Universidade de São Paulo

July 11, 2017

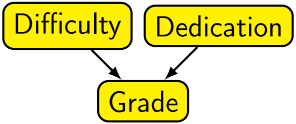
Overview

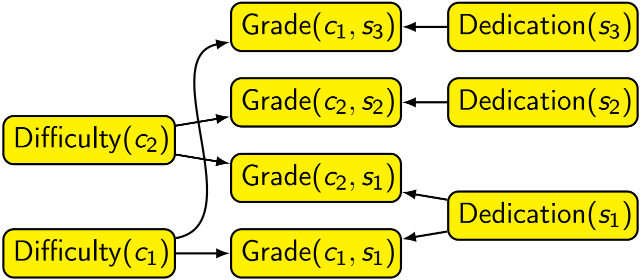
1. A bit of motivation and context.
2. Setting up relational Bayesian network specifications.
3. The descriptive complexity results.
4. Conclusion: A model theory of Bayesian networks?

Bayesian networks, and repetitive patterns...

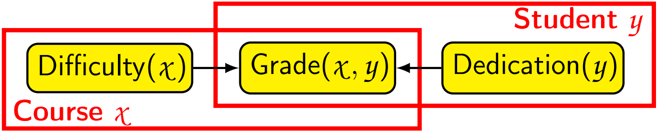


Bayesian networks, and repetitive patterns...



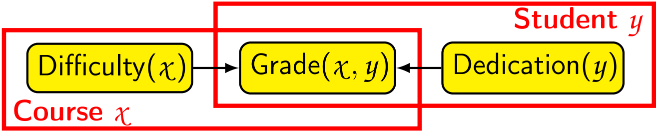


Plates and Probabilistic Relational Models

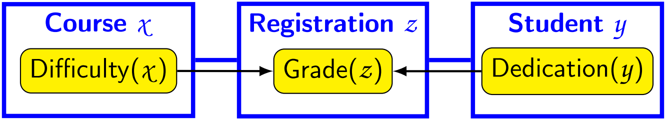


(BUGS)

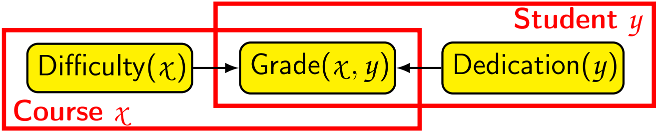
Plates and Probabilistic Relational Models



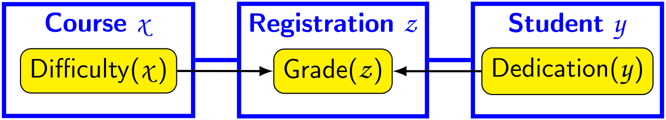
(BUGS)

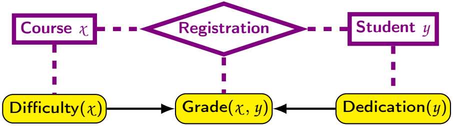


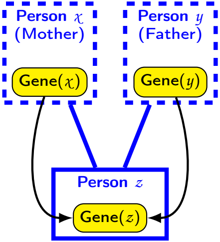
Plates and Probabilistic Relational Models



(BUGS)







And also textual languages...

Logical Bayesian Networks (LBNs)

$$
\begin{aligned}
\text{carrier}(x) \mid\; & \textbf{founder}(x) \\
\text{carrier}(x) \mid\; & \textbf{mother}(m, x), \textbf{father}(f, x), \text{carrier}(m), \text{carrier}(f) \\
\text{suffers}(x) \mid\; & \text{carrier}(x)
\end{aligned}
$$

And also textual languages...

Logical Bayesian Networks (LBNs)

$$\begin{aligned} \text{carrier}(x) &\mid \ \textbf{founder}(x) \\ \text{carrier}(x) &\mid \ \textbf{mother}(m, x), \textbf{father}(f, x), \text{carrier}(m), \text{carrier}(f) \\ \text{suffers}(x) &\mid \ \text{carrier}(x) \end{aligned}$$

Relational Bayesian Networks (RBNs)

$$\begin{aligned} \text{burglary}(v) &= 0.005; \\ \text{alarm}(v) &= (\text{burglary}(v) : 0.95, 0.01); \\ \text{calls}(v, w) &= (\text{neighbor}(v, w) : (\text{alarm}(w) : 0.9, 0.05), 0); \\ \text{alarmed}(v) &= \textit{NoisyOr}\{\text{calls}(w, v) | w : \text{neighbor}(w, v)\} \end{aligned}$$

Stan

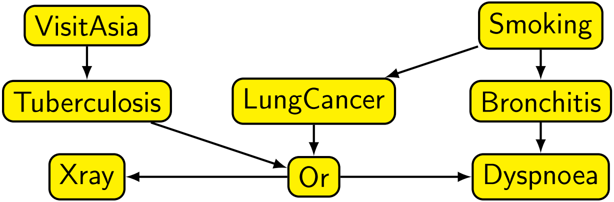
$$\text{model} \quad \{y \ \sim \ \text{std\_normal}();\}$$

The goal here:

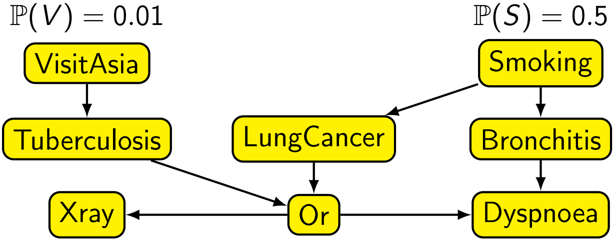
- To study the connection between the language used to specify Bayesian networks and the expressivity of these Bayesian networks.

- To do so by resorting to descriptive complexity, a concept from finite model theory.

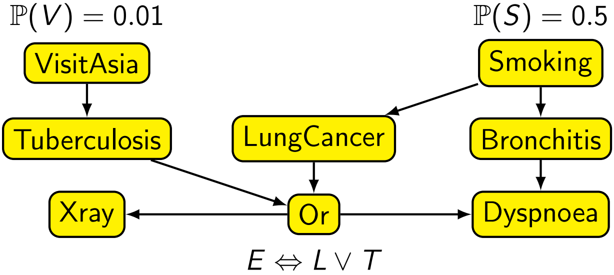
A framework, for propositional languages (Poole 1993)



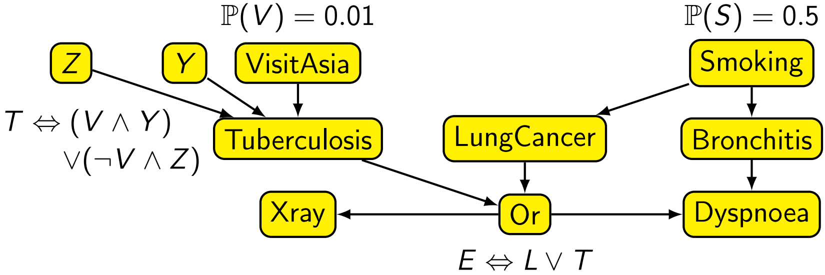
A framework, for propositional languages (Poole 1993)



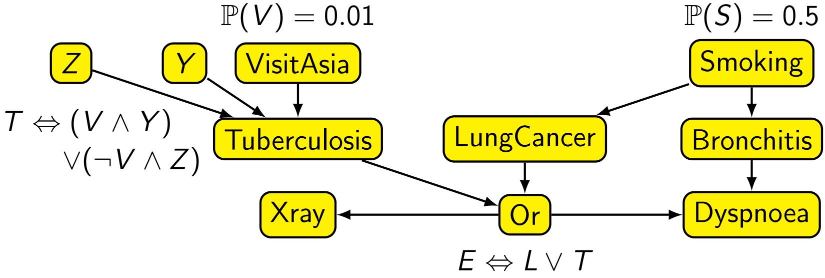
A framework, for propositional languages (Poole 1993)



A framework, for propositional languages (Poole 1993)



A framework, for propositional languages (Poole 1993)



Non-root: $X \Leftrightarrow \phi$ where ϕ in a language $\mathcal{L}$

Now, consider relational Bayesian network specifications

A "fitness"-based model of asymmetric "likeship":

$$
\begin{aligned}
\mathbb{P}(\mathsf{fan}(x)) &= 0.2, \\
\mathbb{P}\big(\mathsf{likes}(x, y)\big) &\Leftrightarrow (x = y) \vee \\
&\quad (\mathsf{fan}(x) \wedge \mathsf{fan}(y)) \vee \\
&\quad \mathsf{other}(x, y), \\
\mathbb{P}\big(\mathsf{other}(x, y)\big) &= 0.1.
\end{aligned}
$$

Now, consider relational Bayesian network specifications

A "fitness"-based model of asymmetric "likeship":

$$
\begin{aligned}
\mathbb{P}(\mathsf{fan}(x)) &= 0.2, \\
\mathbb{P}\big(\mathsf{likes}(x,y)\big) &\Leftrightarrow (x = y) \vee \\
&\quad (\mathsf{fan}(x) \wedge \mathsf{fan}(y)) \vee \\
&\quad \mathsf{other}(x,y), \\
\mathbb{P}\big(\mathsf{other}(x,y)\big) &= 0.1.
\end{aligned}
$$

fan(a) fan(b) fan(c)

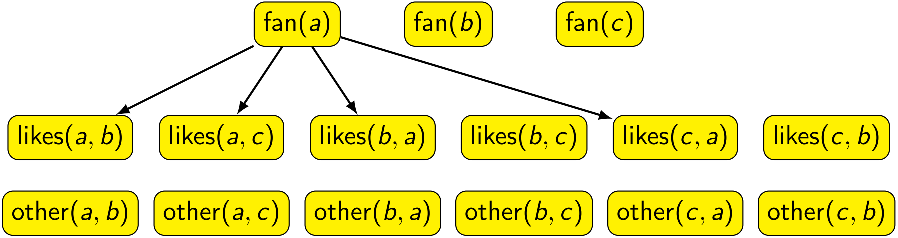
likes(a, b) likes(a, c) likes(b, a) likes(b, c) likes(c, a) likes(c, b)

other(a, b) other(a, c) other(b, a) other(b, c) other(c, a) other(c, b)

Now, consider relational Bayesian network specifications

A "fitness"-based model of asymmetric "likeship":

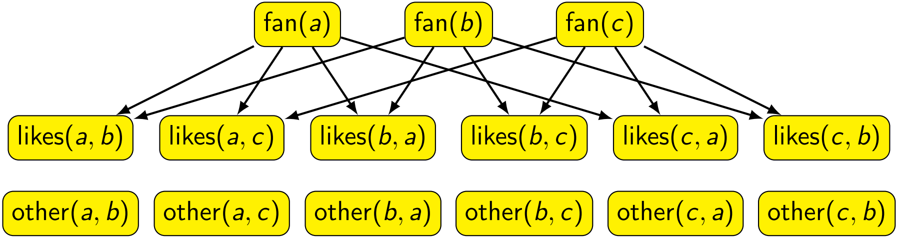
$$
\begin{aligned}
\mathbb{P}(\text{fan}(x)) &= 0.2, \\
\mathbb{P}\big(\text{likes}(x, y)\big) &\Leftrightarrow (x = y) \vee \\
&\quad (\text{fan}(x) \wedge \text{fan}(y)) \vee \\
&\quad \text{other}(x, y), \\
\mathbb{P}\big(\text{other}(x, y)\big) &= 0.1.
\end{aligned}
$$



Now, consider relational Bayesian network specifications

A “fitness”-based model of asymmetric “likeship”:

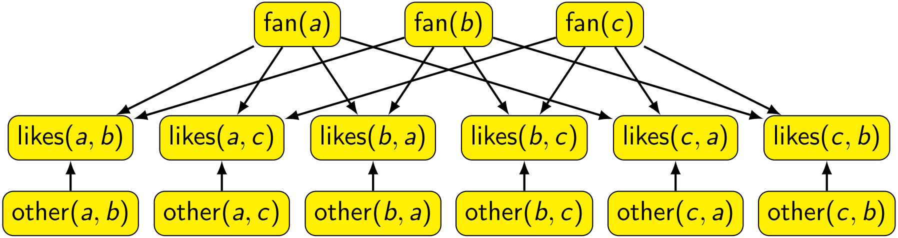
$$
\begin{aligned}
\mathbb{P}(\mathsf{fan}(x)) &= 0.2, \\
\mathbb{P}\big(\mathsf{likes}(x, y)\big) &\Leftrightarrow (x = y) \vee \\
&\quad (\mathsf{fan}(x) \wedge \mathsf{fan}(y)) \vee \\
&\quad \mathsf{other}(x, y), \\
\mathbb{P}\big(\mathsf{other}(x, y)\big) &= 0.1.
\end{aligned}
$$



Now, consider relational Bayesian network specifications

A "fitness"-based model of asymmetric "likeship":

$$\begin{aligned}
\mathbb{P}(\mathsf{fan}(x)) &= 0.2, \\
\mathbb{P}\big(\mathsf{likes}(x, y)\big) &\Leftrightarrow (x = y) \vee \\
& \quad (\mathsf{fan}(x) \wedge \mathsf{fan}(y)) \vee \\
& \quad \mathsf{other}(x, y), \\
\mathbb{P}\big(\mathsf{other}(x, y)\big) &= 0.1.
\end{aligned}$$



Model Theory:

- the study of *models* of formulas in a formal language.

Model Theory:

- the study of *models* of formulas in a formal language.

- Since 1970, finite model theory, motivated by:
  - applications: databases, complexity theory, language design.
  - A central theme:
    The connection between a language and
    what it can/cannot express.

Expressivity: Descriptive complexity

Input (string) → Turing machine → YES/NO

Expressivity: Descriptive complexity

Input (string) → Turing machine → YES/NO

Fagin's theorem (ESO captures NP):

A set of strings is in NP
if and only if
the strings are the finite models of a formula in
existential second-order logic.

Expressivity: Descriptive complexity

Input (string) → Turing machine → YES/NO

Fagin's theorem (ESO captures NP):

A set of strings is in NP
if and only if
the strings are the finite models of a formula in
existential second-order logic.

Example: give relation *edge*, and test model for 3-colorability.

Expressivity: Descriptive complexity

Input (string) → Turing machine → YES/NO

Fagin's theorem (ESO captures NP):

A set of strings is in NP
if and only if
the strings are the finite models of a formula in
existential second-order logic.

Example: give relation *edge*, and test model for 3-colorability.

$$\exists \mathrm{r}, \mathrm{g}, \mathrm{b} : \left(\begin{array}{c} (\forall x : (\text{ either } \mathrm{r}(x) \text{ or } \mathrm{g}(x) \text{ or } \mathrm{b}(x))) \wedge \\ \forall x, y : \mathrm{edge}(x, y) \rightarrow \left(\begin{array}{c} \neg(\mathrm{r}(x) \wedge \mathrm{r}(y)) \wedge \neg(\mathrm{g}(x) \wedge \mathrm{g}(y)) \\ \wedge \neg(\mathrm{b}(x) \wedge \mathrm{b}(y)) \end{array} \right) \end{array} \right)$$

So, let's look at Bayesian network specifications...

- A propositional Bayesian network encodes *a fixed* distribution.
- We can do more with a relational Bayesian network specification. How much more?

The complexity class PP

- Consider a *probabilistic* Turing machine:
  - probabilities are assigned to transitions.
- Suppose that, for any input, we get to know (magically) whether the probability of accepting the input is larger than $1/2$.

The complexity class PP

- Consider a *probabilistic* Turing machine:
  - probabilities are assigned to transitions.
- Suppose that, for any input, we get to know (magically) whether the probability of accepting the input is larger than $1/2$.

- The class of sets of strings thus decided within polynomial time is *exactly* PP.
- So: PP reflects the complexity of computing probabilities for a phenomenon that can be simulated by a polynomial probabilistic Turing machine.

What do relational Bayesian networks specifications capture?

Theorem: Bayesian networks in FFFO "capture" PP

A set of strings is in PP
if and only if
the strings encode the domains/queries with probability $> 1/2$
for a relational Bayesian network specification.

What do relational Bayesian networks specifications capture?

Theorem: Bayesian networks in FFFO "capture" PP

A set of strings is in PP
if and only if
the strings encode the domains/queries with probability $> 1/2$
for a relational Bayesian network specification.

- That is: if you have a phenomenon that can be simulated by a probabilistic Turing machine, it can be modeled by a relational Bayesian network specification.
- More complex phenomena *cannot* be modeled with such a Bayesian network specification (unless complexity classes collapse!).

Going up to second order

- Suppose we allow second-order quantification in the specification. For instance:

$$\text{partitioned} \quad \Leftrightarrow \quad \exists \text{partition} : \forall x : \forall y : \left(\text{edge}(x,y) \Rightarrow \left(\text{partition}(x) \Leftrightarrow \neg\text{partition}(y)\right)\right).$$

Theorem: Bayesian networks in ESO "capture" $\mathsf{PP}^{\mathsf{NP}}$

A set of strings is in $\mathsf{PP}^{\mathsf{NP}}$
if and only if
the strings encode the domains/queries with probability $> 1/2$ for an existential second-order Bayesian network specification.

Conclusion

- A framework in which to study the expressivity of Bayesian network specifications.
  - A theory of descriptive complexity for Bayesian networks.
  - A “feasible” fragment of finite *probabilistic* model theory?

- In short: Relational specifications capture PP; existential second-order specifications capture PP^{NP}.