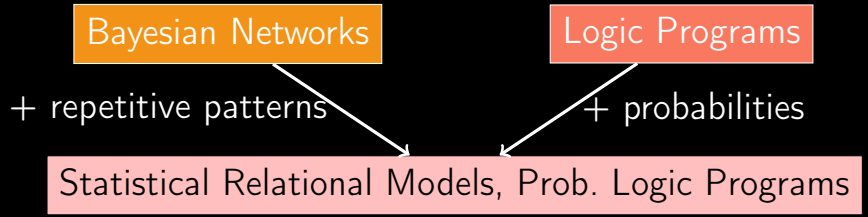
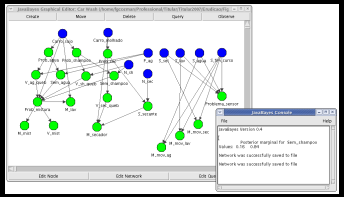
Lecture 2
Section 2: Probabilistic Logic Programming

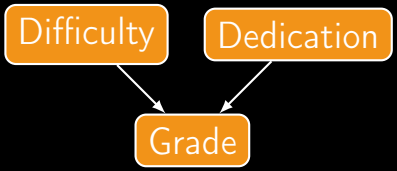
Fabio G. Cozman
Universidade de São Paulo - Brazil

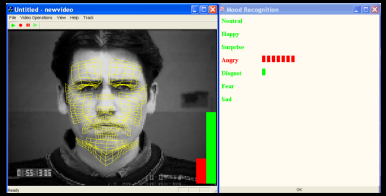
Mixing things. . .

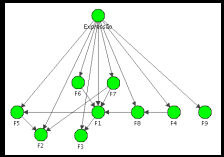


The propositional language of Bayesian networks

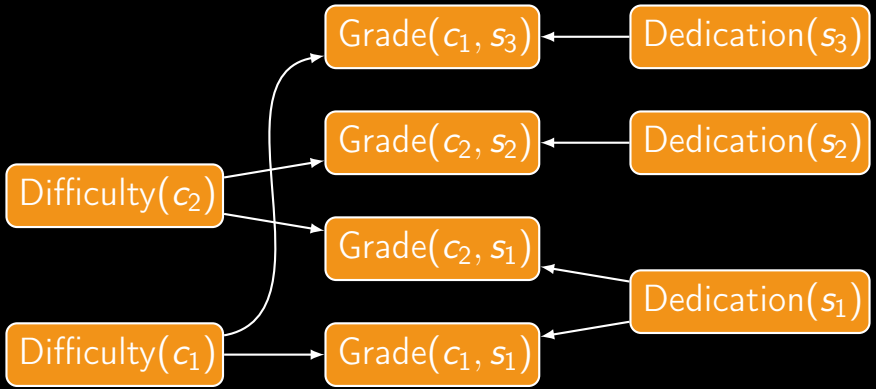




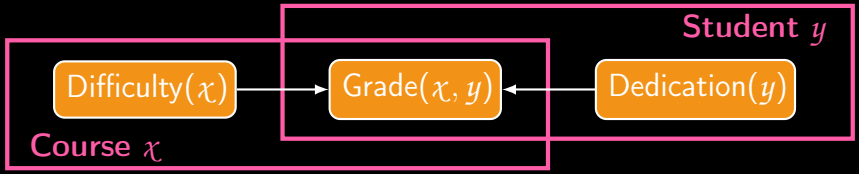


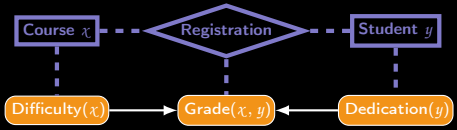


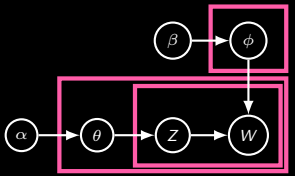
Bayesian networks: repetitive patterns

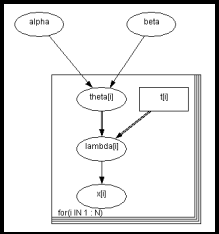


Plates and Probabilistic Relational Models









Logic programming

- Rules:

  $\text{pass}(X, Y) :- \text{student}(X), \text{dedicated}(X), \text{course}(Y), \textbf{not } \text{easy}(Y).$

- Facts:

  $\text{edge}(a, b).$

Logic programming under uncertainty has been around for a while...

- Example (Ng and Subrahmanian 1992):

  $\text{path}(X, Y)\colon [0.85, 1] \leftarrow \text{connect}(X, Z) \ \wedge \ \text{path}(Z, Y)\colon [0.75, 1].$

The distribution semantics and its variants

- Dantsin (1990), Poole (1993), Fuhr (1995), Sato (1995):
  - *Probabilistic facts*:

$$\mathbb{P}(A) = 0.2, \mathbb{P}(B) = 0.3.$$

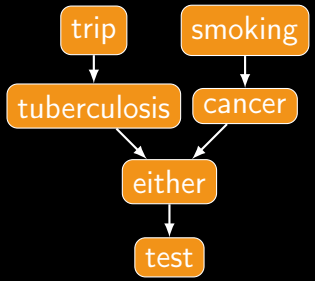
  - A (definite/acyclic/stratified) logic program:

$$C \leftarrow A, B.$$

- A single probability distribution is induced over programs.

Acyclic propositional ProbLog (Bayesian network)

```
0.01 :: trip.
0.5 :: smoking.
tuberculosis :- trip, a1.
tuberculosis :- not trip, a2.
0.05 :: a1.      0.01 :: a2.
cancer :- smoking, a3.
cancer :- not smoking, a4.
0.1 :: a3.       0.01 :: a4.
either :- tuberculosis.
either :- cancer.
test :- either, a5.      0.98 :: a5.
test :- either, a6.      0.05 :: a6.
```



Acyclic relational ProbLog

$0.6 :: \text{dedicated}(X).$
$0.4 :: \text{hard}(Y).$

$\text{fails}(X, Y) \text{ :- } \text{student}(X), \text{dedicated}(X),$
$\text{course}(Y), \text{not hard}(Y), \text{a}(X, Y).$

$0.1 :: \text{a}(X, Y).$

Stratified programs

$\text{edge}(X, Y) :- \text{edge}(Y, X).$
$\text{path}(X, Y) :- \text{edge}(X, Y).$
$\text{path}(X, Y) :- \text{edge}(X, Z), \text{path}(Z, Y).$.

$0.6 :: \text{edge}(1, 2).$
$0.1 :: \text{edge}(1, 3).$
$0.4 :: \text{edge}(2, 5).$
$0.3 :: \text{edge}(2, 6).$
$0.3 :: \text{edge}(3, 4).$
$0.8 :: \text{edge}(4, 5).$
$0.2 :: \text{edge}(5, 6).$

Cyclic programs: many/one/no stable models...

$$\text{engineer}(X) :- \text{person}(X), \text{not laywer}(X).$$

$$\text{laywer}(X) :- \text{person}(X), \text{not engineer}(X).$$

$$0.9 :: \text{person}(\text{dilbert}).$$

- person(dilbert) is false: a unique stable model.
- person(dilbert) is true: there are two stable models (one where dilbert is a lawyer, the other in which he is an engineer).

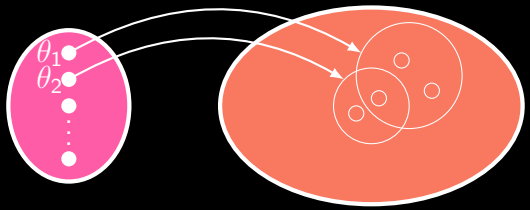
Credal semantics: intuition

- One idea: when there are many possible stable models, impose a probability distribution over them (P-log solution: uniform).
  - For instance: when dilbert is a person, probability 0.5 that it is a lawyer/engineer.

- Another idea: adopt the set of all possible probability distributions!
  - T. Lukasiewicz (2005, 2007).

Credal semantics

- A total choice may induce a program with many stable models.



- Probability of each total choice may be distributed freely over answer sets (we have a random set).
- Semantics is a set of probability distributions that dominates an infinitely-monotone capacity.

Consequences

- The credal semantics of a *consistent* PLP is a convex closed set of probability measures (a credal set).
- $\overline{\mathbb{P}}$ is submodular: $\overline{\mathbb{P}}(A \vee B) \leq \overline{\mathbb{P}}(A) + \overline{\mathbb{P}}(B) - \overline{\mathbb{P}}(A \wedge B)$.
- There are easy formulas for expectations and conditioning:

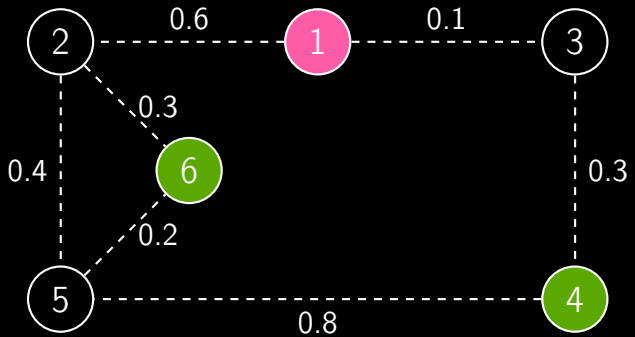
$$\underline{\mathbb{P}}(A|B) = \frac{\underline{\mathbb{P}}(A \wedge B)}{\underline{\mathbb{P}}(A \wedge B) + \overline{\mathbb{P}}(\neg A \wedge B)}.$$

Three-coloring

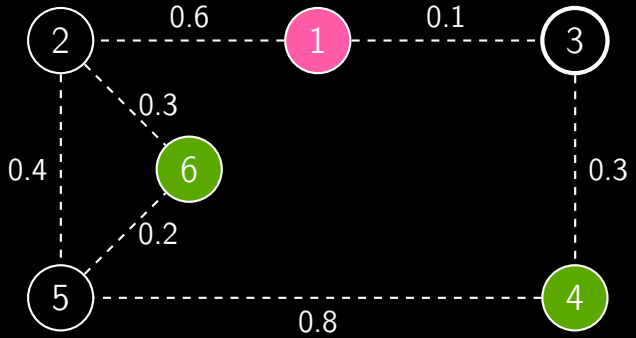
$$\mathrm{red}(X) \vee \mathrm{green}(X) \vee \mathrm{blue}(X) :- \mathrm{node}(X).$$
$$:- \mathrm{edge}(X, Y), \mathrm{red}(X), \mathrm{red}(Y).$$
$$:- \mathrm{edge}(X, Y), \mathrm{green}(X), \mathrm{green}(Y).$$
$$:- \mathrm{edge}(X, Y), \mathrm{blue}(X), \mathrm{blue}(Y).$$

Three-coloring for random graphs

$\mathrm{red}(X) \vee \mathrm{green}(X) \vee \mathrm{blue}(X) :- \mathrm{node}(X).$
$:- \mathrm{edge}(X, Y), \mathrm{red}(X), \mathrm{red}(Y).$
$:- \mathrm{edge}(X, Y), \mathrm{green}(X), \mathrm{green}(Y).$
$:- \mathrm{edge}(X, Y), \mathrm{blue}(X), \mathrm{blue}(Y).$
$0.6 :: \mathrm{edge}(1, 2).$
$0.1 :: \mathrm{edge}(1, 3).$
$0.4 :: \mathrm{edge}(2, 5).$
$0.3 :: \mathrm{edge}(2, 6).$
$0.3 :: \mathrm{edge}(3, 4).$
$0.8 :: \mathrm{edge}(4, 5).$
$0.2 :: \mathrm{edge}(5, 6).$
$\mathrm{red}(1).$
$\mathrm{green}(4).$
$\mathrm{green}(6).$



We have:



- $\overline{\mathbb{P}}(\text{blue}(3)) = 1$, $\underline{\mathbb{P}}(\text{blue}(3)) = 0.03$.
- $\overline{\mathbb{P}}(\text{red}(3)) = 0.9$, $\underline{\mathbb{P}}(\text{red}(3)) = 0$.

Cozman & Mauá, The joy of Probabilistic Answer Set Programming: Semantics, complexity, expressivity, inference, IJAR vol. 125, 2020.

Inference: computing upper probabilities

- Intuition: for each total choice, we must run brave inference.
- Algorithm:
  1. Ground the program.
  2. Turn propositional program into CNF formula in propositional logic.
  3. Run counting algorithm for $\exists$SAT (Aziz et al. 2015).

Closing this section...

- Probabilistic Answer Set Programming gives us a very powerful modeling language.

- The basic (acyclic non-disjunctive) version offers a "relational" version of Bayesian networks;
- the general (cyclic, disjunctive) version asks for a semantics based on credal sets.