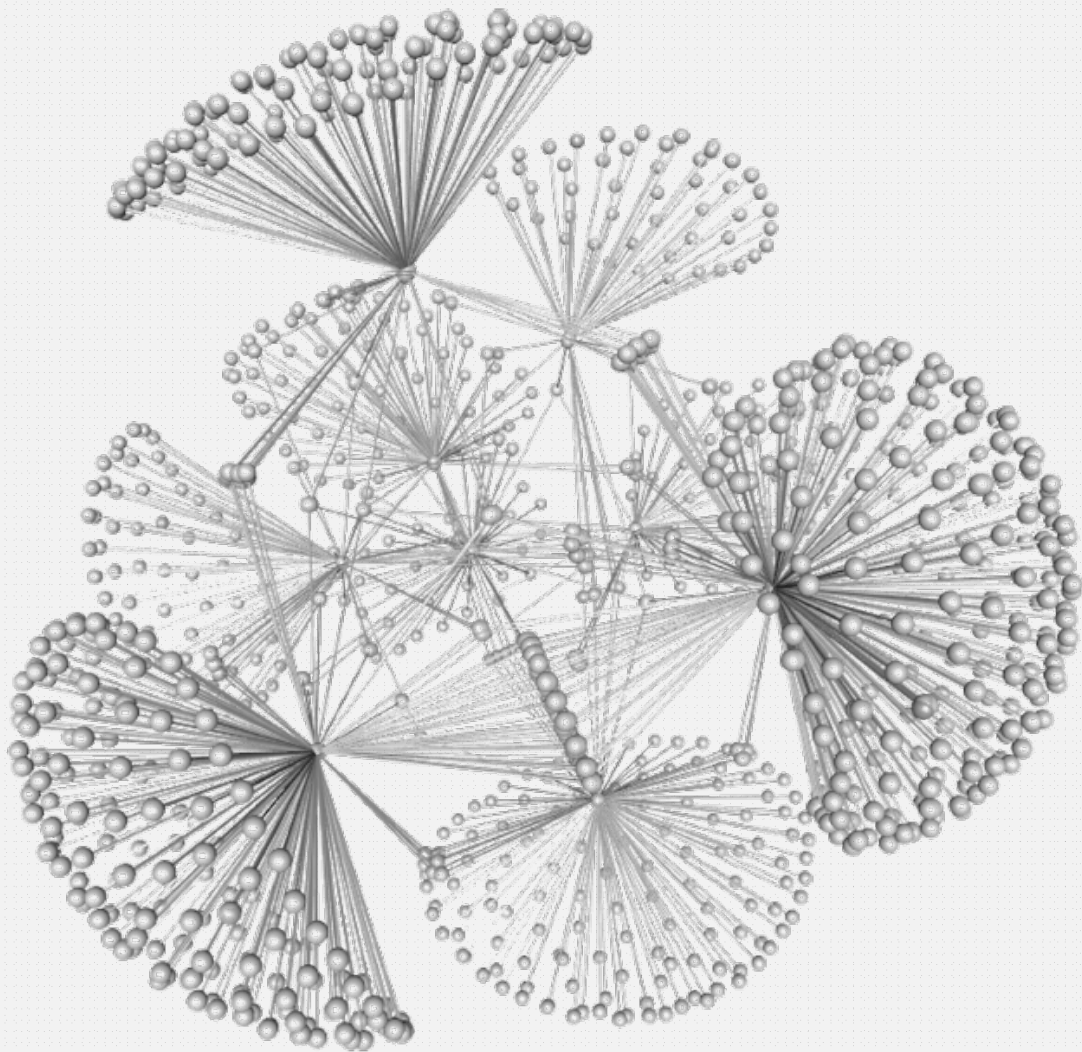


Probabilistic Programming

Michiel Janssen & Bruno Vandekerkhove



ACADEMISCH JAAR 2020

H05N0A: CAPITA SELECTA : ARTIFICIAL
INTELLIGENCE

Contents

| | |
|--|-----------|
| Probabilistic Inference Using Weighted Model Counting | 1 |
| 1.1 SRL to CNF | 1 |
| 1.2 SRL to PGM | 2 |
| 1.3 PGM to CNF | 4 |
| 1.4 Weighted Model Counting | 4 |
| Lifted Inference | 5 |
| 2.1 Calculating Probability with Probabilistic Databases | 5 |
| 2.2 Skolemization & Noisy OR | 6 |
| Parameter Learning | 6 |
| Appendix | 7 |
| A.1 Bayesian Network Encodings | 7 |
| A.1.1 Indicator Clauses | 7 |
| A.1.2 ENC1 (Parameter Clauses) | 7 |
| A.1.3 ENC2 (Parameter Clauses) | 9 |
| References | 11 |

Below's our solution for the given challenges. The questions in each section of the original assignment are answered in a section having the same title.

Probabilistic Inference Using Weighted Model Counting

SRL to CNF

First the program is grounded. This is a matter of collecting all atoms involved in all proofs of the query.

```
1  0.2::stress(a).
2  0.2::stress(b).
3  0.2::stress(c).
4
5  0.1::friends(a,a).
6  0.1::friends(a,b).
7  0.1::friends(a,c).
8
9  0.1::friends(b,a).
10 0.1::friends(b,b).
11 0.1::friends(b,c).
12
13 0.1::friends(c,a).
14 0.1::friends(c,b).
15 0.1::friends(c,c).
16
17 0.3::smokes(a) :- stress(a).
18 0.3::smokes(b) :- stress(b).
19 0.3::smokes(c) :- stress(c).
20 0.4::smokes(a) :- friends(a,a), smokes(a).
21 0.4::smokes(a) :- friends(a,b), smokes(b).
22 0.4::smokes(a) :- friends(a,c), smokes(c).
23 0.4::smokes(b) :- friends(b,a), smokes(a).
```

```
24 0.4::smokes(b) :- friends(b,b), smokes(b).
25 0.4::smokes(b) :- friends(b,c), smokes(c).
26
27 0.4::smokes(c) :- friends(c,a), smokes(a).
28 0.4::smokes(c) :- friends(c,b), smokes(b).
29 0.4::smokes(c) :- friends(c,c), smokes(c).
```

Code snippet 1: Relevant ground program.

The proofs of the query make for a trie as shown in figure 1, where colourings indicate the presence of cycles. Any proof involving an atom `friends(X,X)` or `friends(Y,a)` (with $Y \in \{b,c\}$) is non-minimal and doesn't affect the final probability. These atoms are disregarded. For the remaining cycles (involving `friends(b,c)` and `friends(c,b)`) auxiliary variables can be used to obtain a cycle-free program without intensional probabilistic facts :

```
1  0.2::stress(a).
2  0.2::stress(b).
3  0.2::stress(c).
4
5  0.1::friends(a,b).
6  0.1::friends(a,c).
7  0.1::friends(b,c).
8  0.1::friends(c,b).
9
10 0.3::p(a).
11 0.3::p(b).
12 0.3::p(c).
```

```

13
14 0.4::p(a,b).
15 0.4::p(a,c).
16 0.4::p(b,c).
17 0.4::p(c,b).
18
19 smokes(a) :- stress(a), p(a).
20 smokes(b) :- stress(b), p(b).
21 smokes(c) :- stress(c), p(c).
22
23 smokes(a) :-
24     friends(a,b), smokes(b), p(a,b).
25 smokes(a) :-
26     friends(a,c), smokes(c), p(a,c).
27 smokes(b) :-
28     friends(b,c), stress(c), p(c), p(b,c).
29 smokes(c) :-
30     friends(c,b), stress(b), p(b), p(c,b).
31
32 query(smokes(a)).

```

Code snippet 2: Relevant ground program without cycles.

The above logic program is equivalent to the following propositional formula :

$$\begin{aligned}
& (smokes(a) \leftrightarrow (stress(a) \wedge p(a)) \\
& \quad \vee (friends(a,b) \wedge smokes(b) \wedge p(a,b)) \\
& \quad \vee (friends(a,c) \wedge smokes(c) \wedge p(a,c))) \\
& \quad \wedge \\
& (smokes(b) \leftrightarrow (stress(b) \wedge p(b)) \\
& \quad \vee (friends(b,c) \wedge stress(c) \wedge p(c) \wedge p(b,c))) \\
& \quad \wedge \\
& (smokes(c) \leftrightarrow (stress(c) \wedge p(c)) \\
& \quad \vee (friends(c,b) \wedge stress(b) \wedge p(b) \wedge p(c,b)))
\end{aligned}$$

Which corresponds to the following CNF :

$$\begin{aligned}
& (\neg smokes(a) \vee stress(a) \vee friends(a,b) \vee friends(a,c)) \\
& \wedge (\neg smokes(a) \vee stress(a) \vee friends(a,b) \vee smokes(c)) \\
& \wedge (\neg smokes(a) \vee stress(a) \vee friends(a,b) \vee p(a,c)) \\
& \wedge (\neg smokes(a) \vee stress(a) \vee smokes(b) \vee friends(a,c)) \\
& \wedge (\neg smokes(a) \vee stress(a) \vee smokes(b) \vee smokes(c)) \\
& \wedge (\neg smokes(a) \vee stress(a) \vee smokes(b) \vee p(a,c)) \\
& \wedge (\neg smokes(a) \vee stress(a) \vee p(a,b) \vee friends(a,c)) \\
& \wedge (\neg smokes(a) \vee stress(a) \vee p(a,b) \vee smokes(c)) \\
& \wedge (\neg smokes(a) \vee stress(a) \vee p(a,b) \vee p(a,c)) \\
& \wedge (\neg smokes(a) \vee p(a) \vee friends(a,b) \vee friends(a,c)) \\
& \wedge (\neg smokes(a) \vee p(a) \vee friends(a,b) \vee smokes(c)) \\
& \wedge (\neg smokes(a) \vee p(a) \vee friends(a,b) \vee p(a,c)) \\
& \wedge (\neg smokes(a) \vee p(a) \vee smokes(b) \vee friends(a,c)) \\
& \wedge (\neg smokes(a) \vee p(a) \vee smokes(b) \vee smokes(c)) \\
& \wedge (\neg smokes(a) \vee p(a) \vee smokes(b) \vee p(a,c)) \\
& \wedge (\neg smokes(a) \vee p(a) \vee p(a,b) \vee friends(a,c)) \\
& \wedge (\neg smokes(a) \vee p(a) \vee p(a,b) \vee smokes(c)) \\
& \wedge (\neg smokes(a) \vee p(a) \vee p(a,b) \vee p(a,c)) \\
& \wedge (\neg stress(a) \vee \neg p(a) \vee smokes(a)) \\
& \wedge (\neg friends(a,b) \vee \neg smokes(b) \vee \neg p(a,b) \vee smokes(a)) \\
& \wedge (\neg friends(a,c) \vee \neg smokes(c) \vee \neg p(a,c) \vee smokes(a)) \\
& \wedge (\neg smokes(b) \vee stress(b) \vee friends(b,c)) \\
& \wedge (\neg smokes(b) \vee stress(b) \vee stress(c))
\end{aligned}$$

$$\begin{aligned}
& \wedge (\neg smokes(b) \vee stress(b) \vee p(c)) \\
& \wedge (\neg smokes(b) \vee stress(b) \vee p(b,c)) \\
& \wedge (\neg smokes(b) \vee p(b) \vee friends(b,c)) \\
& \wedge (\neg smokes(b) \vee p(b) \vee stress(c)) \\
& \wedge (\neg smokes(b) \vee p(b) \vee p(c)) \\
& \wedge (\neg smokes(b) \vee p(b) \vee p(b,c)) \\
& \wedge (\neg stress(b) \vee \neg p(b) \vee smokes(b)) \\
& \wedge (\neg friends(b,c) \vee \neg stress(c) \vee \neg p(c) \vee \neg p(b,c) \vee \\
& \quad smokes(b)) \\
& \wedge (\neg smokes(c) \vee stress(c) \vee friends(c,b)) \\
& \wedge (\neg smokes(c) \vee stress(c) \vee stress(b)) \\
& \wedge (\neg smokes(c) \vee stress(c) \vee p(b)) \\
& \wedge (\neg smokes(c) \vee stress(c) \vee p(c,b)) \\
& \wedge (\neg smokes(c) \vee p(c) \vee friends(c,b)) \\
& \wedge (\neg smokes(c) \vee p(c) \vee stress(b)) \\
& \wedge (\neg smokes(c) \vee p(c) \vee p(b)) \\
& \wedge (\neg smokes(c) \vee p(c) \vee p(c,b)) \\
& \wedge (\neg stress(c) \vee \neg p(c) \vee smokes(c)) \\
& \wedge (\neg friends(c,b) \vee \neg stress(b) \vee \neg p(b) \vee \neg p(c,b) \vee \\
& \quad smokes(c))
\end{aligned}$$

The probabilistic literals in the CNF are assigned weights (derived literals get a weight of 1) :

| Literal | Weight |
|---------------------|--------|
| stress(a) | 0.2 |
| \neg stress(a) | 0.8 |
| stress(b) | 0.2 |
| \neg stress(b) | 0.8 |
| stress(c) | 0.2 |
| \neg stress(c) | 0.8 |
| friends(a,b) | 0.1 |
| \neg friends(a,b) | 0.9 |
| friends(a,c) | 0.1 |
| \neg friends(a,c) | 0.9 |
| friends(b,c) | 0.1 |
| \neg friends(b,c) | 0.9 |
| friends(c,b) | 0.1 |
| \neg friends(c,b) | 0.9 |
| p(a) | 0.3 |
| \neg p(a) | 0.7 |
| p(b) | 0.3 |
| \neg p(b) | 0.7 |
| p(c) | 0.3 |
| \neg p(c) | 0.7 |
| p(a,b) | 0.4 |
| \neg p(a,b) | 0.6 |
| p(a,c) | 0.4 |
| \neg p(a,c) | 0.6 |
| p(b,c) | 0.4 |
| \neg p(b,c) | 0.6 |
| p(c,b) | 0.4 |
| \neg p(c,b) | 0.6 |

SRL to PGM

A Bayesian network is shown in figure 2. The conditional probability tables (CPTs) for the nodes are given below. Note that every table represents multiple identical tables in the network (like those of the *stress(a)*, *stress(b)* and *stress(c)* nodes for example).

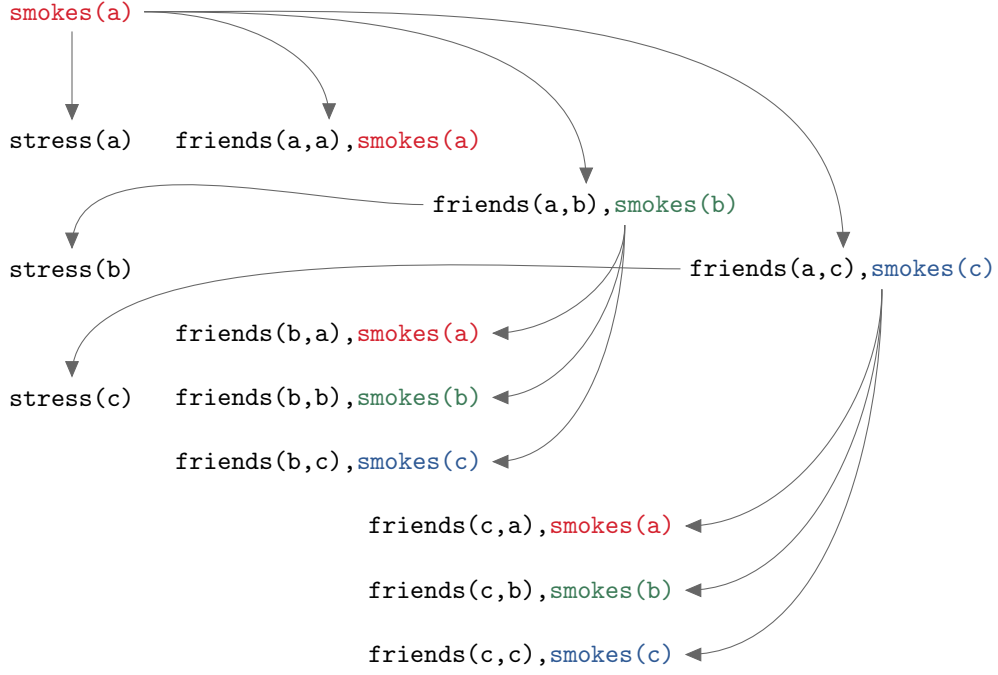


Figure 1: Trie representing proofs of the query. Coloured atoms indicate the presence of cycles.

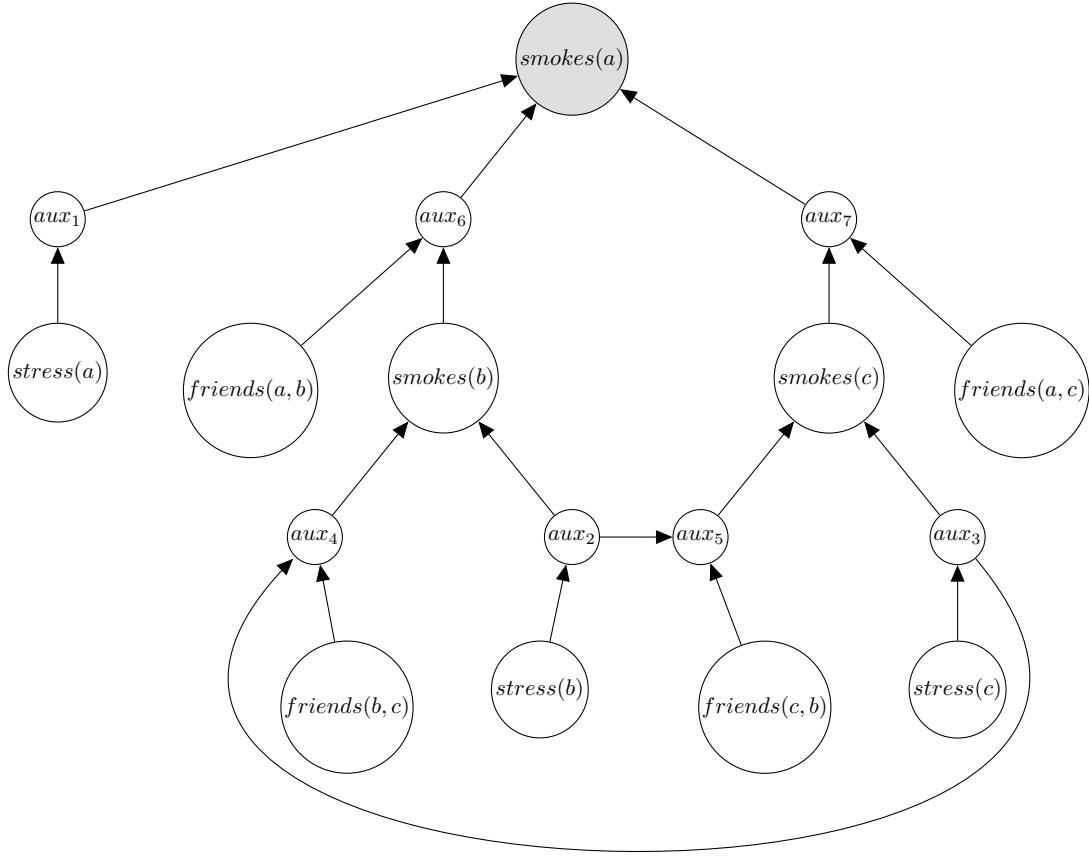


Figure 2: Bayesian network corresponding to the ground acyclic program.

| $stress(\{a, b, c\})$ | | $friends(\{a, b, c\}, \{b, c\})$ | |
|-----------------------|-----|----------------------------------|-----|
| \top | 0.2 | \top | 0.1 |
| \perp | 0.8 | \perp | 0.9 |

| $stress(\{a, b, c\})$ | $aux_{\{1,2,3\}}$ | |
|-----------------------|-------------------|---------|
| | \top | \perp |
| \top | 0.3 | 0.7 |
| \perp | 0.0 | 1.0 |

| $friends(\{b, c\}, \{c, b\})$ | $aux_{2,3}$ | $aux_{4,5}$ | |
|-------------------------------|-------------|-------------|---------|
| | | \top | \perp |
| \top | \top | 0.4 | 0.6 |
| \top | \perp | 0.0 | 1.0 |
| \perp | \top | 0.0 | 1.0 |
| \perp | \perp | 0.0 | 1.0 |

| $friends(\{a\}, \{b, c\})$ | $smokes(\{b, c\})$ | $aux_{6,7}$ | |
|----------------------------|--------------------|-------------|---------|
| | | \top | \perp |
| \top | \top | 0.4 | 0.6 |
| \top | \perp | 0.0 | 1.0 |
| \perp | \top | 0.0 | 1.0 |
| \perp | \perp | 0.0 | 1.0 |

| $aux_{4,5}$ | $aux_{2,3}$ | $smokes(\{b, c\})$ | |
|-------------|-------------|--------------------|---------|
| | | \top | \perp |
| \top | \top | 1.0 | 0.0 |
| \top | \perp | 0.0 | 1.0 |
| \perp | \top | 0.0 | 1.0 |
| \perp | \perp | 0.0 | 1.0 |

| aux_1 | $smokes(b)$ | $smokes(c)$ | $smokes(a)$ | |
|---------|-------------|-------------|-------------|---------|
| | | | \top | \perp |
| \top | \top | \top | 1.0 | 0.0 |
| \top | \top | \perp | 0.0 | 1.0 |
| \top | \perp | \top | 0.0 | 1.0 |
| \top | \perp | \perp | 0.0 | 1.0 |
| \perp | \top | \top | 0.0 | 1.0 |
| \perp | \top | \perp | 0.0 | 1.0 |
| \perp | \perp | \top | 0.0 | 1.0 |
| \perp | \perp | \perp | 0.0 | 1.0 |

PGM to CNF

The Bayesian network can be encoded as a logical formula. Encodings ENC1 and ENC2 discussed by Chavira [1] both make use of the same indicator variables such as $\lambda_{stress(a)=true}$ and $\lambda_{stress(a)=false}$ (which introduces redundancy considering the fact that all network variables are boolean).

In ENC1 each row in each CPT is encoded by a parameter clause.

In ENC2 an order is assumed over each variable's values. Then, each row of each CPT is encoded by an equivalence.

For both of these a Python script was written to auto-

mate the creation of CNF files, since manual conversion to a CNF turned out to be cumbersome. The script also produces L^AT_EX output which is added to the appendix.

Since the noisy OR relations end up being encoded naively by considering each row of the CPT separately, a more compact encoding is obtained by replacing this part of the encoding with a one-liner. For example :

$$smokes(a) \Leftrightarrow aux_1 \vee aux_6 \vee aux_7$$

This can be converted to a CNF in the usual way :

$$\begin{aligned} & (smokes(a) \vee \neg aux_1 \vee \neg aux_6 \vee \neg aux_7) \\ & \wedge (\neg smokes(a) \vee aux_1) \\ & \wedge (\neg smokes(a) \vee aux_6) \\ & \wedge (\neg smokes(a) \vee aux_7) \end{aligned}$$

Weighted Model Counting

The previous paragraphs led to the construction of 5 different encodings. All of them lie in the `/src/bayesian/` folder, having the following file names :

| Encoding | Filename |
|-------------------------|--------------------------------|
| Conversion from ProbLog | <code>propositional.cnf</code> |
| ENC1 | <code>enc1.cnf</code> |
| ENC1 + noisy OR | <code>enc1_noisy.cnf</code> |
| ENC2 | <code>enc2.cnf</code> |
| ENC2 + noisy OR | <code>enc2_noisy.cnf</code> |

Weighted model counting was performed on all of them using either PySDD or miniC2D. A modified version of Cachet that allows for the specification of negative weights was toyed with too, though not for all encodings as it expects a different format for specifying the weights. These were the results :

| Encoding | PySDD | miniC2D |
|-------------------------|-------|---------|
| Conversion from ProbLog | XXX | XXX |
| ENC1 | XXX | XXX |
| ENC1 + noisy OR | XXX | XXX |
| ENC2 | XXX | XXX |
| ENC2 + noisy OR | XXX | XXX |

The counts can be interpreted as probabilities (keeping in mind that some assumptions result from the semantics of the ProbLog program, such as independence of friendships). In this particular case since no query was specified (yet), the probability equals 1 since the count represents the probability of any possible world.

To answer a more complex query like $P(smokes(a) = \top)$ a line was added to the CNF files : $smokes(a)$. This makes sure that any model is one where person a actually smokes. A more complex query like $P(smokes(a) = \top \mid friends(a, b) = \top, friends(a, c) = \top)$ can be computed in the way Fierens proposed [2], by dividing the joint probability of these three atoms by the probability of the previous query. One could also toy with the weights of the literals.

The smallest circuit sizes were the following :

| Encoding | Hyperparameters | PySDD | miniC2D |
|-------------------------|-----------------|-------|---------|
| Conversion from ProbLog | XXX | XXX | XXX |
| ENC1 | XXX | XXX | XXX |
| ENC1 + noisy OR | XXX | XXX | XXX |
| ENC2 | XXX | XXX | XXX |
| ENC2 + noisy OR | XXX | XXX | XXX |

| Encoding | $P(smokes(a) = \top)$ | $P(smokes(a) = \top \mid friends(a, b) = \top, friends(a, c) = \top)$ |
|-------------------------|-----------------------|---|
| Conversion from ProbLog | XXX | XXX |
| ENC1 | XXX | XXX |
| ENC1 + noisy OR | XXX | XXX |
| ENC2 | XXX | XXX |
| ENC2 + noisy OR | XXX | XXX |

EXPLAIN DIFFERENCE BETWEEN MODEL COUNTERS HERE

Lifted Inference

Calculating Probability with Probabilistic Databases

Starting from the following probabilistic database tables :

| X | $\frac{stress(X)}{}$ | X | Y | $\frac{friends(X, Y)}{}$ | X | $\frac{p(X)}{}$ | X | Y | $\frac{p(X, Y)}{}$ |
|-----|----------------------|-----|-----|--------------------------|-----|-----------------|-----|-----|--------------------|
| a | 0.2 | a | b | 0.1 | a | 0.3 | a | b | 0.4 |
| b | 0.2 | a | c | 0.1 | b | 0.3 | a | c | 0.4 |
| c | 0.2 | b | c | 0.1 | c | 0.3 | b | c | 0.4 |
| | | c | b | 0.1 | | | c | b | 0.4 |

Table 1: Probabilistic database to be used for querying.

Querying for $smokes(a)$ can be done as follows :

$$(stress(a) \wedge p(a)) \vee (\exists x : p(a, x) \wedge friends(a, x) \wedge ((stress(x) \wedge p(x)) \vee \exists y : p(x, y) \wedge friends(x, y) \wedge stress(y) \wedge p(y)))$$

Rules can be applied to this formula. Applying the inclusive or rule gives :

$$1 - (1 - p_{stress(a)} \wedge p(a)) \times (1 - p_{s_1})$$

Where s_1 refers to the subformula for the first existential operator. The probability p_{s_1} cannot simply be calculated by exponentiation as the various instantiations of s_1 are dependent. However, there are only two such instantiations (for $smokes(a)$ and $smokes(b)$) and since the existential operator generalises the logical or, the formula for exclusive or can directly be used instead :

$$p_{s_1} = p_{friends(a,b)} \times p_{p(a,b)} \times p_{smokes(b)} + p_{friends(a,c)} \times p_{p(a,c)} \times p_{smokes(c)} - p_{both}$$

Where $p_{both} = p_{friends(a,b)} \wedge p_{p(a,b)} \wedge p_{friends(a,c)} \wedge p_{p(a,c)} \wedge p_{smokes(b)} \wedge p_{smokes(c)}$. The probability of either $smokes(b)$ or $smokes(c)$ is the same (as is the probability of $friends(a, b) \wedge p(a, b)$ and $friends(a, c) \wedge p(a, c)$, so this reduces to :

$$p_{s_1} = 2 \times p_{friends(a,b)} \times p_{p(a,b)} \times p_{smokes(b)} - p_{friends(a,b) \wedge friends(a,c) \wedge smokes(b) \wedge smokes(c)}$$

The probability of $smokes(b)$ can be calculated using the same rules (an inclusive or and the and-rule) :

$$\begin{aligned}
& 1 - (1 - p_{stress(b)} \times p_{p(b)}) \times (1 - p_{friends(b,c)} \times p_{p(b,c)} \times p_{stress(c)} \times p_{p(c)}) \\
& = (1 - (1 - 0,2 \times 0,3) \times (1 - 0,1 \times 0,4 \times 0,3 \times 0,2)) \\
& \approx 0.62256
\end{aligned}$$

The probability of $p_{smokes(b) \wedge smokes(c)}$ can also be calculated with these rules, for example :

$$\begin{aligned}
& p_{smokes(b) \wedge smokes(c)} \\
& = p_{stress(b) \wedge p(b) \wedge smokes(c)} + p_{stress(c) \wedge p(c) \wedge smokes(b)} - p_{all} \\
& = 2 \times p_{stress(b)} \times p_{p(b)} \times (p_{stress(c) \wedge p(c)} + p_{friends(b,c) \wedge p(b,c)} - p_{stress(c) \wedge p(c) \wedge friends(b,c) \wedge p(b,c)}) \\
& \quad - p_{stress(b)} \times p_{p(b)} \times p_{stress(c)} \times p_{p(c)}
\end{aligned}$$

The probability of $p_{stress(a)} \wedge p(a)$ equals $p_{p(a)} \times p_{stress(a)}$. After filling in all the missing probabilities of the involved literals (by looking them up in the database) the following total probability is obtained :

$$\begin{aligned}
& 1 - (1 - 0,2 \times 0,3) \times (1 - (2 \times 0,1 \times 0,4 \times (1 - (1 - 0,2 \times 0,3) \times (1 - 0,1 \times 0,4 \times 0,3 \times 0,2)) - \\
& 0,1 \times 0,4 \times 0,1 \times 0,4 \times (0,2 \times 0,3 \times (0,2 \times 0,3 + 0,1 \times 0,4 - 0,2 \times 0,3 \times 0,1 \times 0,4) + 0,2 \times 0,3 \times \\
& (0,2 \times 0,3 + 0,1 \times 0,4 - 0,2 \times 0,3 \times 0,1 \times 0,4) - 0,2 \times 0,3 \times 0,2 \times 0,3))) \\
& \approx 0,06466945075
\end{aligned}$$

This is the same number as the one previously found by the weighted model counters. Some symmetry was taken advantage of.

Skolemization & Noisy OR

A new encoding can be made by, instead of converting the formula $X \Leftrightarrow Y_1 \vee Y_2 \vee \dots \vee Y_n$ to CNF in the usual way, applying a Tseitin transformation. This leads to :

$$\begin{aligned}
& X \vee \neg Y_1 \\
& X \vee \neg Y_2 \\
& \dots \\
& X \vee \neg Y_n \\
& X \vee T \\
& T \vee \neg Y_1 \\
& T \vee \neg Y_2 \\
& \dots \\
& T \vee \neg Y_n
\end{aligned}$$

Applying this transformation to the noisy OR relations in the encoding lead to a slightly *larger* circuit though, when tested with a program in which there were more ancestors, the resulting circuits were *smaller*.

Parameter Learning

Appendix

Bayesian Network Encodings

Indicator Clauses

$\lambda_{stress(a)=true} \vee \lambda_{stress(a)=false}$
 $\neg \lambda_{stress(a)=true} \vee \neg \lambda_{stress(a)=false}$
 $\lambda_{stress(b)=true} \vee \lambda_{stress(b)=false}$
 $\neg \lambda_{stress(b)=true} \vee \neg \lambda_{stress(b)=false}$
 $\lambda_{stress(c)=true} \vee \lambda_{stress(c)=false}$
 $\neg \lambda_{stress(c)=true} \vee \neg \lambda_{stress(c)=false}$
 $\lambda_{aux1=true} \vee \lambda_{aux1=false}$
 $\neg \lambda_{aux1=true} \vee \neg \lambda_{aux1=false}$
 $\lambda_{aux2=true} \vee \lambda_{aux2=false}$
 $\neg \lambda_{aux2=true} \vee \neg \lambda_{aux2=false}$
 $\lambda_{aux3=true} \vee \lambda_{aux3=false}$
 $\neg \lambda_{aux3=true} \vee \neg \lambda_{aux3=false}$
 $\lambda_{friends(a,b)=true} \vee \lambda_{friends(a,b)=false}$
 $\neg \lambda_{friends(a,b)=true} \vee \neg \lambda_{friends(a,b)=false}$
 $\lambda_{friends(a,c)=true} \vee \lambda_{friends(a,c)=false}$
 $\neg \lambda_{friends(a,c)=true} \vee \neg \lambda_{friends(a,c)=false}$
 $\lambda_{friends(b,c)=true} \vee \lambda_{friends(b,c)=false}$
 $\neg \lambda_{friends(b,c)=true} \vee \neg \lambda_{friends(b,c)=false}$
 $\lambda_{friends(c,b)=true} \vee \lambda_{friends(c,b)=false}$
 $\neg \lambda_{friends(c,b)=true} \vee \neg \lambda_{friends(c,b)=false}$
 $\lambda_{aux4=true} \vee \lambda_{aux4=false}$
 $\neg \lambda_{aux4=true} \vee \neg \lambda_{aux4=false}$
 $\lambda_{aux5=true} \vee \lambda_{aux5=false}$
 $\neg \lambda_{aux5=true} \vee \neg \lambda_{aux5=false}$
 $\lambda_{smokes(b)=true} \vee \lambda_{smokes(b)=false}$
 $\neg \lambda_{smokes(b)=true} \vee \neg \lambda_{smokes(b)=false}$
 $\lambda_{smokes(c)=true} \vee \lambda_{smokes(c)=false}$
 $\neg \lambda_{smokes(c)=true} \vee \neg \lambda_{smokes(c)=false}$
 $\lambda_{aux6=true} \vee \lambda_{aux6=false}$
 $\neg \lambda_{aux6=true} \vee \neg \lambda_{aux6=false}$
 $\lambda_{aux7=true} \vee \lambda_{aux7=false}$
 $\neg \lambda_{aux7=true} \vee \neg \lambda_{aux7=false}$
 $\lambda_{smokes(a)=true} \vee \lambda_{smokes(a)=false}$
 $\neg \lambda_{smokes(a)=true} \vee \neg \lambda_{smokes(a)=false}$

ENC1 (Parameter Clauses)

$\lambda_{stress(a)=true} \Leftrightarrow \theta_{stress(a)=true}$
 $\lambda_{stress(a)=false} \Leftrightarrow \theta_{stress(a)=false}$
 $\lambda_{stress(b)=true} \Leftrightarrow \theta_{stress(b)=true}$
 $\lambda_{stress(b)=false} \Leftrightarrow \theta_{stress(b)=false}$
 $\lambda_{stress(c)=true} \Leftrightarrow \theta_{stress(c)=true}$
 $\lambda_{stress(c)=false} \Leftrightarrow \theta_{stress(c)=false}$
 $\lambda_{stress(a)=true} \wedge \lambda_{aux1=true} \Leftrightarrow \theta_{aux1=true|stress(a)=true}$
 $\lambda_{stress(a)=true} \wedge \lambda_{aux1=false} \Leftrightarrow \theta_{aux1=false|stress(a)=true}$
 $\lambda_{stress(a)=false} \wedge \lambda_{aux1=true} \Leftrightarrow \theta_{aux1=true|stress(a)=false}$
 $\lambda_{stress(a)=false} \wedge \lambda_{aux1=false} \Leftrightarrow \theta_{aux1=false|stress(a)=false}$
 $\lambda_{stress(b)=true} \wedge \lambda_{aux2=true} \Leftrightarrow \theta_{aux2=true|stress(b)=true}$
 $\lambda_{stress(b)=true} \wedge \lambda_{aux2=false} \Leftrightarrow \theta_{aux2=false|stress(b)=true}$
 $\lambda_{stress(b)=false} \wedge \lambda_{aux2=true} \Leftrightarrow \theta_{aux2=true|stress(b)=false}$
 $\lambda_{stress(b)=false} \wedge \lambda_{aux2=false} \Leftrightarrow \theta_{aux2=false|stress(b)=false}$
 $\lambda_{stress(c)=true} \wedge \lambda_{aux3=true} \Leftrightarrow \theta_{aux3=true|stress(c)=true}$
 $\lambda_{stress(c)=true} \wedge \lambda_{aux3=false} \Leftrightarrow \theta_{aux3=false|stress(c)=true}$
 $\lambda_{stress(c)=false} \wedge \lambda_{aux3=true} \Leftrightarrow \theta_{aux3=true|stress(c)=false}$
 $\lambda_{stress(c)=false} \wedge \lambda_{aux3=false} \Leftrightarrow \theta_{aux3=false|stress(c)=false}$

[illegible]

$$\begin{aligned}
\lambda_{aux1}=true \wedge \lambda_{aux6}=false \wedge \lambda_{aux7}=true \wedge \lambda_{smokes}(a)=true &\Leftrightarrow \theta_{smokes(a)=true|aux1=true\wedge aux6=false\wedge aux7=true} \\
\lambda_{aux1}=true \wedge \lambda_{aux6}=false \wedge \lambda_{aux7}=true \wedge \lambda_{smokes}(a)=false &\Leftrightarrow \theta_{smokes(a)=false|aux1=true\wedge aux6=false\wedge aux7=true} \\
\lambda_{aux1}=true \wedge \lambda_{aux6}=false \wedge \lambda_{aux7}=false \wedge \lambda_{smokes}(a)=true &\Leftrightarrow \theta_{smokes(a)=true|aux1=true\wedge aux6=false\wedge aux7=false} \\
\lambda_{aux1}=true \wedge \lambda_{aux6}=false \wedge \lambda_{aux7}=false \wedge \lambda_{smokes}(a)=false &\Leftrightarrow \theta_{smokes(a)=false|aux1=true\wedge aux6=false\wedge aux7=false} \\
\lambda_{aux1}=false \wedge \lambda_{aux6}=true \wedge \lambda_{aux7}=true \wedge \lambda_{smokes}(a)=true &\Leftrightarrow \theta_{smokes(a)=true|aux1=false\wedge aux6=true\wedge aux7=true} \\
\lambda_{aux1}=false \wedge \lambda_{aux6}=true \wedge \lambda_{aux7}=true \wedge \lambda_{smokes}(a)=false &\Leftrightarrow \theta_{smokes(a)=false|aux1=false\wedge aux6=true\wedge aux7=true} \\
\lambda_{aux1}=false \wedge \lambda_{aux6}=true \wedge \lambda_{aux7}=false \wedge \lambda_{smokes}(a)=true &\Leftrightarrow \theta_{smokes(a)=true|aux1=false\wedge aux6=true\wedge aux7=false} \\
\lambda_{aux1}=false \wedge \lambda_{aux6}=true \wedge \lambda_{aux7}=false \wedge \lambda_{smokes}(a)=false &\Leftrightarrow \theta_{smokes(a)=false|aux1=false\wedge aux6=true\wedge aux7=false} \\
\lambda_{aux1}=false \wedge \lambda_{aux6}=false \wedge \lambda_{aux7}=true \wedge \lambda_{smokes}(a)=true &\Leftrightarrow \theta_{smokes(a)=true|aux1=false\wedge aux6=false\wedge aux7=true} \\
\lambda_{aux1}=false \wedge \lambda_{aux6}=false \wedge \lambda_{aux7}=true \wedge \lambda_{smokes}(a)=false &\Leftrightarrow \theta_{smokes(a)=false|aux1=false\wedge aux6=false\wedge aux7=true} \\
\lambda_{aux1}=false \wedge \lambda_{aux6}=false \wedge \lambda_{aux7}=false \wedge \lambda_{smokes}(a)=true &\Leftrightarrow \theta_{smokes(a)=true|aux1=false\wedge aux6=false\wedge aux7=false} \\
\lambda_{aux1}=false \wedge \lambda_{aux6}=false \wedge \lambda_{aux7}=false \wedge \lambda_{smokes}(a)=false &\Leftrightarrow \theta_{smokes(a)=false|aux1=false\wedge aux6=false\wedge aux7=false}
\end{aligned}$$

ENC2 (Parameter Clauses)

[illegible]

[illegible]

References

- [1] Mark Chavira and Adnan Darwiche. On probabilistic inference by weighted model counting. *Artificial Intelligence*, 172(6):772 – 799, 2008.
- [2] Daan Fierens, Guy Van Den Broeck, Joris Renkens, Dimitar Shterionov, Bernd Gutmann, Ingo Thon, Gerda Janssens, and Luc De Raedt. Inference and learning in probabilistic logic programs using weighted boolean formulas. *Theory and Practice of Logic Programming*, 15(3):358–401, April 2014.