Model equivalence of PRISM programs

James Cussens

Dept of Computer Science & York Centre for Complex Systems Analysis
University of York, York YO10 5DD, UK
jc@cs.york.ac.uk

Abstract. The problem of deciding the probability model equivalence of two PRISM programs is addressed. In the finite case this problem can be solved (albeit slowly) using techniques from *algebraic statistics*, specifically the computation of elimination ideals and Gröbner bases. A very brief introduction to algebraic statistics is given. Consideration is given to cases where shortcuts to proving/disproving model equivalence are available.

Keywords. PRISM programs, model equivalence, model inclusion, algebraic statistics, algebraic geometry, ideals, varieties, Gröbner bases, polynomials

1 Introduction

This report presents a solution, in the finite case, to the following *PRISM probability model equivalence problem*:

> Given two PRISM programs both with unspecified parameters decide whether they represent the same probability model (i.e. set of probability distributions) for a given target predicate.

This problem in turn was prompted by the problem of efficient EM maximum likelihood parameter estimation (MLPE) for PRISM programs. The connection is the following: if two PRISM programs represent the same probability model (i.e. set of distributions) then for a given data set the maximum likelihood distribution is the same for both programs. However, it may be that MLPE is faster for one program than another thus it is worth looking into ways of automatically finding whichever program allows the fastest MLPE for a given probability model.

Sometimes it is easy to make MLPE more efficient. For example, in those cases where the structure of a PRISM program directly encodes a decomposable hierarchical model it is straightforward to construct a program with a different structure—encoding a Bayesian net—which nevertheless represents the same probability model. This latter representation allows exact MLPE without recourse to an iterative approach. So program transformation can be used to allow efficient MLPE. The question then arises whether program transformation can

Dagstuhl Seminar Proceedings 07161
Probabilistic, Logical and Relational Learning - A Further Synthesis
http://drops.dagstuhl.de/opus/volltexte/2008/1380

also be used when (as is typical) the original PRISM program does not encode a decomposable hierarchical model.

PRISM programs can represent many probabilistic models, including hierarchical models, Bayesian networks, stochastic grammars (including HMMs and PCFGs) and phylogenetic trees. Given this representational power, it is perhaps not surprising that powerful mathematical tools are required to answer the PRISM model equivalence problem. (In this paper simple models are used as examples, but the techniques developed are applicable to general PRISM programs.)

Here *algebraic statistics* and, in particular, Gröbner bases [1] are used to analyse the problem. It is shown that when the success set of the target predicate is finite, it is possible to decide whether two PRISM programs defining this target predicate are probability model equivalent. In the infinite case an algorithm is given which when provided with two non-equivalent programs will eventually detect that they are non-equivalent, but which will not terminate for equivalent programs. (It seems likely that model equivalence is decidable even in the infinite case, but, at present, no suitable algorithm has been found.) Although algebraic statistics provides the necessary mathematical analysis and algorithms, these algorithms may be excessively inefficient for practical use. So, in addition, consideration is given to cases where 'short cuts' to answering the model inclusion question are possible.

This report assumes knowledge of the essentials of PRISM programs. If this is lacking then [2] can be consulted for the basics and [3] for the extension to 'failure' models. To nudge readers' memories here is a brief description nonetheless. A PRISM program is a logic program together with a probabilistic built-in predicate `msw/2`. A ground fact such as `msw('X1',x)` is actually an abbreviation for a fact `msw('X1',j,x)` which is a statement that it is true that the random variable $X_{1,j}$ is instantiated to have a value x, where j is some natural number. The *parameters* of a PRISM program define the (discrete) probability distribution for the $X_{1,j}$. For any $j, j' \in \mathcal{N}$, $X_{1,j}$ and $X_{1,j'}$ must be independent and identically distributed (which motivates the abbreviation just mentioned). Such random variables are called *switches*. A ground fact of the form `msw('X2',x)`—an abbreviation for `msw('X2',j,x)` for some `j`—declares that $X_{2,j} = x$. $X_{1,j}$ and $X_{2,j'}$ are always independent for all j, j' (but not necessarily identically distributed). A possible world is determined by which ground `msw/3` facts are true ('how the switches are set'). Leaving aside complications which arise if negation is permitted (such complications are evaded throughout this paper), any collection of `msw/3` facts together with the rest of the clauses of the PRISM program defines a unique least Herbrand model. Thus the parameters of the PRISM program define a product distribution over a set of least Herbrand models—possible worlds. This distribution induces a distribution over the success sets of predicates defined in terms of the `msw/3` predicate. For any such predicate p/n', the probability of the atomic formula $p(a_1, a_2, \ldots a_{n'})$ is the sum of the probabilities of the worlds which satisfy it (in the case of failure models this is only up to

normalisation). Usually a particular *target predicate* is distinguished to represent 'the' distribution defined by a PRISM program.

2 Some example PRISM programs

Before diving into the mathematical analysis two example PRISM programs are now given: a hierarchical model and a Bayesian network. These will serve as running examples and provide 'instantiations' of the later more abstract material. In the interests of simplicity and compactness the PRISM code used to present each example is the simplest possible, where the target predicate is defined directly in terms of `msw/2` switches. Note that generally, PRISM programs are considerably more complex using many 'intermediary' predicates.

2.1 Hierarchical models

Let $Y_1, Y_2, \ldots, Y_m$ be a finite set of finite discrete variables. Let $\mathcal{H}$ be a *hypergraph* with the Y_i as vertices. $\mathcal{H}$ is thus just a set of subsets of the Y_i. Each subset of variables $h \in \mathcal{H}$ is called a *hyperedge*. $\mathcal{H}$ defines a *hierarchical model* which is simply the set of all joint distributions over the Y_i of the form:

$$P(Y_1, Y_2, \ldots, Y_m) = Z^{-1} \prod_{h \in \mathcal{H}} \psi_h(Y_1, Y_2, \ldots, Y_m)$$

where the value of ψ_h depends only on the $Y_i \in h$ and otherwise are arbitrary functions to the non-negative reals. Since everything is finite the functions ψ_h can be represented in tabular form usually called factors. If the hyperedges are the cliques of some undirected graph then the hypergraph is *graphical* and the hierarchical model is a *graphical model* [4]. An example of a non-graphical hypergraph is $\{\{Y_1, Y_2\}, \{Y_1, Y_3\}, \{Y_2, Y_3\}\}$

Consider the hierarchical model for binary variables A, B, C, D defined by the hypergraph $\{\{A, B\}, \{B, C\}, \{C, D\}, \{A, D\}\}$. Note that this is a graphical model, whose corresponding graph is given in Fig 1.

Table 1 shows a particular distribution in this model. Let this particular distribution be called P_{hm}. The value of Z which follows from these four factors is 0.2165. The value of $P_{hm}(A = 0, B = 0, C = 0, D = 0)$ for example is $(0.1 \times 0.4 \times 0.5 \times 0.9)/0.2165 \approx 0.08$.

The key to the PRISM representation of a hierarchical model is the simple observation that a factored distribution like P_{hm} remains unchanged if each factor is normalised to become a 'local' probability distribution. Applying this to the representation given in Table 1 gives (to 2 significant figures) the four factors in Table 2. For this representation the value of Z is approximately 0.07913. The value of $P_{hm}(A = 0, B = 0, C = 0, D = 0)$ is $(0.13 \times 0.26 \times 0.42 \times 0.47)/0.07913 \approx 0.08$ as before.

The PRISM program encoding P_{hm} is given in Fig 2. We can use PRISM to compute $P(A = 0, B = 0, C = 0, D = 0)$ as follows:

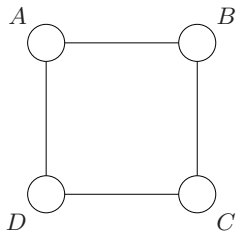

Fig. 1. Graphical model used as a running example

Table 1. Factors defining a distribution in the hierarchical model defined by hypergraph $\{\{A, B\}, \{B, C\}, \{C, D\}, \{A, D\}\}$.

A	B	
0	0	0.1
0	1	0.2
1	0	0.3
1	1	0.2

B	C	
0	0	0.4
0	1	0.7
1	0	0.3
1	1	0.1

C	D	
0	0	0.5
0	1	0.2
1	0	0.4
1	1	0.1

A	D	
0	0	0.9
0	1	0.2
1	0	0.7
1	1	0.1

Table 2. Factors which are local probability distributions defining the same distribution as that given by Table 1. (The numerical values are not exact.)

A	B	
0	0	0.13
0	1	0.25
1	0	0.37
1	1	0.25

B	C	
0	0	0.26
0	1	0.47
1	0	0.20
1	1	0.07

C	D	
0	0	0.42
0	1	0.17
1	0	0.33
1	1	0.08

A	D	
0	0	0.47
0	1	0.11
1	0	0.37
1	1	0.05

```
| ?- prob(hm(0,0,0,0),Psi), prob(hm(_,_,_,_),Z), P is Psi/Z.
prob(hm(0,0,0,0),Psi), prob(hm(_,_,_,_),Z), P is Psi/Z.

Z = 0.07908944
Psi = 0.00667212
P = 0.084361704925462 ?
yes
```

```
target(hm,4).

values(factor(_),[(0,0),(0,1),(1,0),(1,1)]).
:- set_sw(factor(ab),0.13+0.25+0.37+0.25).
:- set_sw(factor(bc),0.26+0.47+0.20+0.07).
:- set_sw(factor(cd),0.42+0.17+0.33+0.08).
:- set_sw(factor(ad),0.47+0.11+0.37+0.05).

hm(A,B,C,D) :-
   msw(factor(ab),(A,B)),
   msw(factor(bc),(B,C)),
   msw(factor(cd),(C,D)),
   msw(factor(ad),(A,D)).
```

Fig. 2. PRISM representation of a (fitted) hierarchical model P_{hm}

2.2 Bayesian networks

Bayesian networks are easily encoded as PRISM programs using the approach given by Poole [5]. Pearl [6] has recently advocated this representation for analysing causality.

Consider the BN with four binary variables A, B, C, D with this recursive decomposition: $P(A, B, C, D) = P(A)P(B|A)P(C|A, B)P(D|A, C)$. The relevant graph is given in Fig 3. This representation involves 1+2+4+4=11 conditional distributions (there is one on each row of each CPT). A PRISM representation has 11 switches, and represents the network structure by the rules of the program. Fig 4 shows the PRISM program with switch probabilities set so as to give the same distribution as that given by Tables 1 and 2.

3 Polynomial equations generated by PRISM programs

If we temporarily leave aside the normalisation required by programs with failure, the probability of any target atom is just the sum of the probabilities of the possible worlds in which it is true. Since the distribution over possible worlds

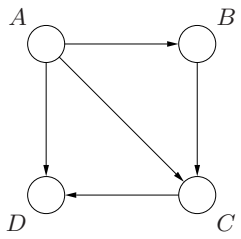

Fig. 3. Bayesian network running example

```
target(bn,4).

values(_,[0,1]).
:- set_sw(cpt(a),0.38+0.62).
:- set_sw(cpt(b,0),0.56+0.44).
:- set_sw(cpt(b,1),0.79+0.21).
:- set_sw(cpt(c,0,0),0.42+0.58).
:- set_sw(cpt(c,0,1),0.79+0.21).
:- set_sw(cpt(c,1,0),0.42+0.58).
:- set_sw(cpt(c,1,1),0.79+0.21).
:- set_sw(cpt(d,0,0),0.92+0.08).
:- set_sw(cpt(d,0,1),0.95+0.05).
:- set_sw(cpt(d,1,0),0.95+0.05).
:- set_sw(cpt(d,1,1),0.97+0.03).


bn(A,B,C,D) :-
  msw(cpt(a),A),
  msw(cpt(b,A),B),
  msw(cpt(c,A,B),C),
  msw(cpt(d,A,C),D).
```

Fig. 4. PRISM program representing a Bayesian network

is determined by the switch probabilities it follows that the probability of any target atom is a function of the switch probabilities. Here the **Finite Support Condition** is invoked which guarantees that this function is a *polynomial*. The Finite Support Condition is generally assumed form PRISM programs, for more about it see [2]).

This can be illustrated using the hierarchical model example. To aid comprehension the relevant random variables will be described with the symbols used in the actual PRISM program given in Fig 2. There are $n = 4$ families of random variables: $\{X_{\mathtt{ab},j}\}_j, \{X_{\mathtt{bc},j}\}_j, \{X_{\mathtt{cd},j}\}_j$ and $\{X_{\mathtt{ad},j}\}_j$. The four probabilities making up the distribution for $\{X_{\mathtt{ab},j}\}_j$ are $p_{AB_{00}}, p_{AB_{01}}, p_{AB_{10}}$ and $p_{AB_{11}}$, a similar naming convention will be used for the other three random variables. The 16 target predicate atoms are $hm(0,0,0,0), hm(0,0,0,1), \ldots hm(1,1,1,1)$. Let their probabilities be $p_{0,0,0,0}, p_{0,0,0,1}, \ldots p_{1,1,1,1}$. Let FAIL denote the set of possible worlds in which no target predicate atom is true. Then:

$$
\begin{aligned}
p_{0,0,0,0} &= p_{AB_{00}} p_{BC_{00}} p_{CD_{00}} p_{AD_{00}} \\
p_{0,0,0,1} &= p_{AB_{00}} p_{BC_{00}} p_{CD_{01}} p_{AD_{01}} \\
p_{0,0,1,0} &= p_{AB_{00}} p_{BC_{01}} p_{CD_{10}} p_{AD_{00}} \\
p_{0,0,1,1} &= p_{AB_{00}} p_{BC_{01}} p_{CD_{11}} p_{AD_{01}} \\
p_{0,1,0,0} &= p_{AB_{01}} p_{BC_{10}} p_{CD_{00}} p_{AD_{00}} \\
p_{0,1,0,1} &= p_{AB_{01}} p_{BC_{10}} p_{CD_{01}} p_{AD_{01}} \\
p_{0,1,1,0} &= p_{AB_{01}} p_{BC_{11}} p_{CD_{10}} p_{AD_{00}} \\
p_{0,1,1,1} &= p_{AB_{01}} p_{BC_{11}} p_{CD_{11}} p_{AD_{01}} \\
p_{1,0,0,0} &= p_{AB_{10}} p_{BC_{00}} p_{CD_{00}} p_{AD_{00}} \\
p_{1,0,0,1} &= p_{AB_{10}} p_{BC_{00}} p_{CD_{01}} p_{AD_{11}} \\
p_{1,0,1,0} &= p_{AB_{10}} p_{BC_{01}} p_{CD_{10}} p_{AD_{10}} \\
p_{1,0,1,1} &= p_{AB_{10}} p_{BC_{01}} p_{CD_{11}} p_{AD_{11}} \\
p_{1,1,0,0} &= p_{AB_{11}} p_{BC_{10}} p_{CD_{00}} p_{AD_{10}} \\
p_{1,1,0,1} &= p_{AB_{11}} p_{BC_{10}} p_{CD_{01}} p_{AD_{11}} \\
p_{1,1,1,0} &= p_{AB_{11}} p_{BC_{11}} p_{CD_{10}} p_{AD_{10}} \\
p_{1,1,1,1} &= p_{AB_{11}} p_{BC_{11}} p_{CD_{11}} p_{AD_{11}} \\
P(\text{FAIL}) &= p_{AB_{00}} p_{BC_{10}} + p_{AB_{00}} p_{BC_{11}} + p_{AB_{01}} p_{BC_{00}} + p_{AB_{01}} p_{BC_{01}} + \ldots \\
&\quad p_{AB_{11}} p_{BC_{11}} p_{CD_{11}} p_{AD_{01}} + p_{AB_{11}} p_{BC_{11}} p_{CD_{11}} p_{AD_{10}}
\end{aligned}
$$

In PRISM, as in the closely related formalism of stochastic logic programs [7,8], by definition the probability distribution given by specifying a target predicate gives probability one to the set of all ground atoms of that predicate. If failure is a possibility, then this is not automatically the case—some probability mass is 'missing', having been assigned to FAIL. This can be fixed by simple conditioning. Doing this using the current hierarchical model case produces the

following set of polynomial equations:

$$
\begin{aligned}
zp_{0,0,0,0} &= p_{AB_{00}} p_{BC_{00}} p_{CD_{00}} p_{AD_{00}} \\
zp_{0,0,0,1} &= p_{AB_{00}} p_{BC_{00}} p_{CD_{01}} p_{AD_{01}} \\
zp_{0,0,1,0} &= p_{AB_{00}} p_{BC_{01}} p_{CD_{10}} p_{AD_{00}} \\
zp_{0,0,1,1} &= p_{AB_{00}} p_{BC_{01}} p_{CD_{11}} p_{AD_{01}} \\
zp_{0,1,0,0} &= p_{AB_{01}} p_{BC_{10}} p_{CD_{00}} p_{AD_{00}} \\
zp_{0,1,0,1} &= p_{AB_{01}} p_{BC_{10}} p_{CD_{01}} p_{AD_{01}} \\
zp_{0,1,1,0} &= p_{AB_{01}} p_{BC_{11}} p_{CD_{10}} p_{AD_{00}} \\
zp_{0,1,1,1} &= p_{AB_{01}} p_{BC_{11}} p_{CD_{11}} p_{AD_{01}} \\
zp_{1,0,0,0} &= p_{AB_{10}} p_{BC_{00}} p_{CD_{00}} p_{AD_{00}} \\
zp_{1,0,0,1} &= p_{AB_{10}} p_{BC_{00}} p_{CD_{01}} p_{AD_{11}} \\
zp_{1,0,1,0} &= p_{AB_{10}} p_{BC_{01}} p_{CD_{10}} p_{AD_{10}} \\
zp_{1,0,1,1} &= p_{AB_{10}} p_{BC_{01}} p_{CD_{11}} p_{AD_{11}} \\
zp_{1,1,0,0} &= p_{AB_{11}} p_{BC_{10}} p_{CD_{00}} p_{AD_{10}} \\
zp_{1,1,0,1} &= p_{AB_{11}} p_{BC_{10}} p_{CD_{01}} p_{AD_{11}} \\
zp_{1,1,1,0} &= p_{AB_{11}} p_{BC_{11}} p_{CD_{10}} p_{AD_{10}} \\
zp_{1,1,1,1} &= p_{AB_{11}} p_{BC_{11}} p_{CD_{11}} p_{AD_{11}} \\
z &= p_{AB_{00}} p_{BC_{00}} p_{CD_{00}} p_{AD_{00}} + p_{AB_{00}} p_{BC_{00}} p_{CD_{01}} p_{AD_{01}} + \cdots + p_{AB_{11}} p_{BC_{11}} p_{CD_{11}} p_{AD_{11}}
\end{aligned}
\tag{1}
$$

Finally, since we have chosen a parameterisation where factors have to be probability distributions we have the additional constraints:

$$
\begin{aligned}
p_{AB_{00}} + p_{AB_{01}} + p_{AB_{10}} + p_{AB_{11}} &= 1 \\
p_{BC_{00}} + p_{BC_{01}} + p_{BC_{10}} + p_{BC_{11}} &= 1 \\
p_{CD_{00}} + p_{CD_{01}} + p_{CD_{10}} + p_{CD_{11}} &= 1 \\
p_{AD_{00}} + p_{AD_{01}} + p_{AD_{10}} + p_{AD_{11}} &= 1
\end{aligned}
\tag{2}
$$

A similar set of polynomial equations is easily produced for the Bayesian network example from Section 2.2. Recall that in this example we have the decomposition $P(A, B, C, D) = P(A)P(B|A)P(C|A, B)P(D|A, C)$. Abbreviating $P(A = 0)$ to p_{A_0}, $P(B = 1|A = 0)$ to $p_{BA_{10}}$ and $P(D = 1|A = 0, C = 1)$ to $p_{DAC_{101}}$ and similarly for all other parameters, the values for all probabilities in the joint distribution $P(A, B, C, D)$ can be written as:

$$
\begin{aligned}
p_{0,0,0,0} &= p_{A_0} p_{BA_{00}} p_{CB_{00}} p_{DAC_{000}} \\
p_{0,0,0,1} &= p_{A_0} p_{BA_{00}} p_{CB_{00}} p_{DAC_{100}} \\
p_{0,0,1,0} &= p_{A_0} p_{BA_{00}} p_{CB_{10}} p_{DAC_{001}} \\
p_{0,0,1,1} &= p_{A_0} p_{BA_{00}} p_{CB_{10}} p_{DAC_{101}} \\
p_{0,1,0,0} &= p_{A_0} p_{BA_{10}} p_{CB_{01}} p_{DAC_{000}} \\
p_{0,1,0,1} &= p_{A_0} p_{BA_{10}} p_{CB_{01}} p_{DAC_{100}} \\
p_{0,1,1,0} &= p_{A_0} p_{BA_{10}} p_{CB_{11}} p_{DAC_{001}} \\
p_{0,1,1,1} &= p_{A_0} p_{BA_{10}} p_{CB_{11}} p_{DAC_{101}} \\
p_{1,0,0,0} &= p_{A_1} p_{BA_{01}} p_{CB_{00}} p_{DAC_{010}} \\
p_{1,0,0,1} &= p_{A_1} p_{BA_{01}} p_{CB_{00}} p_{DAC_{110}} \\
p_{1,0,1,0} &= p_{A_1} p_{BA_{01}} p_{CB_{10}} p_{DAC_{011}} \\
p_{1,0,1,1} &= p_{A_1} p_{BA_{01}} p_{CB_{10}} p_{DAC_{111}} \\
p_{1,1,0,0} &= p_{A_1} p_{BA_{11}} p_{CB_{01}} p_{DAC_{010}} \\
p_{1,1,0,1} &= p_{A_1} p_{BA_{11}} p_{CB_{01}} p_{DAC_{110}} \\
p_{1,1,1,0} &= p_{A_1} p_{BA_{11}} p_{CB_{11}} p_{DAC_{011}} \\
p_{1,1,1,1} &= p_{A_1} p_{BA_{11}} p_{CB_{11}} p_{DAC_{111}}
\end{aligned}
\tag{3}
$$

There are also the following constraints on the parameters:

$$
\begin{aligned}
p_{A_0} + p_{A_1} &= 1 \\
p_{BA_{00}} + p_{BA_{10}} &= 1 \\
p_{BA_{01}} + p_{BA_{11}} &= 1 \\
p_{CB_{00}} + p_{CB_{10}} &= 1 \\
p_{CB_{01}} + p_{CB_{11}} &= 1 \\
p_{DAC_{000}} + p_{DAC_{100}} &= 1 \\
p_{DAC_{001}} + p_{DAC_{101}} &= 1 \\
p_{DAC_{010}} + p_{DAC_{110}} &= 1 \\
p_{DAC_{011}} + p_{DAC_{111}} &= 1
\end{aligned}
\tag{4}
$$

It is clear that in both the hierarchical model case and the Bayesian net case the parameters determine the probabilities $p_{0,0,0,0}, p_{0,0,0,1}, \ldots p_{1,1,1,1}$ (and also z in the case of the hierarchical model). From now on the probabilities $p_{0,0,0,0}, p_{0,0,0,1}, \ldots p_{1,1,1,1}$ will be referred to as the *target distribution probabilities*. Of more interest is the *set* of all target probability distributions which are definable by (legal) values of the parameters. In particular the questions arises as to whether these two sets are equal in the two cases.

It is clear that the above equations effect constraints between parameters and target distribution probabilities and also on target distribution probabilities alone. To analyse such constraints effectively and to answer the model equivalence problem for the general case it is necessary to turn to the techniques of algebraic statistics.

4 Algebraic statistics of PRISM programs

In the simultaneous polynomial equations (1) and (2) for the hierarchical model there are in total $(4 + 4 + 4 + 4) + 16 + 1 = 33$ variables involved. From now on, to avoid confusion with random variables and also to follow the conventions in algebraic statistics these variables will be called *indeterminates* rather than variables. Any set of 33 numbers satisfying all these equations can be seen as a point in $\mathcal{R}^{33}$. The set of all such points thus define a subset of $\mathcal{R}^{33}$. (Clearly, this subset has lower dimension than $\mathcal{R}^{33}$.) The set of target probability distributions defined by considering all possible values of the parameters can be viewed as the projection of this set down onto the 16 dimensions corresponding to the target distribution probabilities $p_{0,0,0,0}, \ldots p_{1,1,1,1}$. This geometric view is the key to understanding model equivalence, but to actually implement the necessary algorithms an algebraic approach is necessary. Fortunately, algebraic statistics provides the necessary bridge between algebra and geometry.

Algebraic statistics is a sub-branch of algebraic geometry devoted to analysing statistical problems. A central concept of algebraic geometry (and thus also of algebraic statistics) is that of an *ideal* of polynomials. As succinctly put in [9] "A polynomial ideal formalises the intuitive idea of the algebraic consequences of a system of polynomial equations". A polynomial ideal is just a set of polynomials which are closed under addition and product by other polynomials. More precisely, let $k[x_1, \ldots x_s]$ be the set of all polynomials in some set of indeterminates $x_1, \ldots x_s$ with coefficients belonging to the field k. (In this paper it will be assumed that $k = \mathcal{R}$.) $I \subset k[x_1, \ldots x_s]$ is an ideal if, for all $f, g \in I$ and $h \in k[x_1, \ldots x_s]$, $f + g \in I$ and $hf \in I$.

Ideals *generated* by finite sets of polynomials are of particular interest. These are defined as follows in [9].

Definition 1. *An ideal I is* finitely generated *if there exists $f_1, \ldots, f_r$ polynomials in $k[x_1, \ldots x_s]$ such that for any $f \in I$ there exist $s_1, s_2, \ldots s_r$ polynomials of $k[x_1, \ldots x_s]$ such that*

$$f = \sum_{i=1}^{r} s_i f_i$$

We write $I = \langle f_1, \ldots, f_r \rangle$ and the set $\{f_1, \ldots, f_r\}$ is called a basis *of I.*

It is not difficult to see that for any set of polynomials $\{f_1, \ldots, f_r\}$, $I = \langle f_1, \ldots, f_r \rangle$ meets the conditions of being an ideal. The Hilbert Basis Theorem states that, in fact, *any* ideal in $k[x_1, \ldots x_s]$ has a finite basis.

Ideals are algebraic objects. To see how they can be used to analyse constraints on probability distributions, it is useful to consider the associated geometric concept, that of a *variety*. Again, the relevant definition is taken from [9].

Definition 2. *The* variety *generated by a polynomial ideal* $I \subset k[x_1, \ldots x_s]$ *is*

$$\text{Variety}(I) = \{(a_1, \ldots a_s) \in k^s : f(a_1, \ldots, a_s) = 0 \text{ for all } f \in I\}$$

So the variety of an ideal is the set of points in k^s which are zero-points for all polynomials in the ideal.

Returning again to the polynomials for the hierarchical model (1) and (2). It is not difficult to see that the set of points in $\mathcal{R}^{33}$ simultaneously satisfying all these equations is a variety. Indeed it is the variety of the ideal generated by these 21 polynomials:

$$
\begin{aligned}
& z p_{0,0,0,0} - p_{AB_{00}} p_{BC_{00}} p_{CD_{00}} p_{AD_{00}} \\
& z p_{0,0,0,1} - p_{AB_{00}} p_{BC_{00}} p_{CD_{01}} p_{AD_{01}} \\
& z p_{0,0,1,0} - p_{AB_{00}} p_{BC_{01}} p_{CD_{10}} p_{AD_{00}} \\
& z p_{0,0,1,1} - p_{AB_{00}} p_{BC_{01}} p_{CD_{11}} p_{AD_{01}} \\
& z p_{0,1,0,0} - p_{AB_{01}} p_{BC_{10}} p_{CD_{00}} p_{AD_{00}} \\
& z p_{0,1,0,1} - p_{AB_{01}} p_{BC_{10}} p_{CD_{01}} p_{AD_{01}} \\
& z p_{0,1,1,0} - p_{AB_{01}} p_{BC_{11}} p_{CD_{10}} p_{AD_{00}} \\
& z p_{0,1,1,1} - p_{AB_{01}} p_{BC_{11}} p_{CD_{11}} p_{AD_{01}} \\
& z p_{1,0,0,0} - p_{AB_{10}} p_{BC_{00}} p_{CD_{00}} p_{AD_{00}} \\
& z p_{1,0,0,1} - p_{AB_{10}} p_{BC_{00}} p_{CD_{01}} p_{AD_{11}} \\
& z p_{1,0,1,0} - p_{AB_{10}} p_{BC_{01}} p_{CD_{10}} p_{AD_{10}} \\
& z p_{1,0,1,1} - p_{AB_{10}} p_{BC_{01}} p_{CD_{11}} p_{AD_{11}} \\
& z p_{1,1,0,0} - p_{AB_{11}} p_{BC_{10}} p_{CD_{00}} p_{AD_{10}} \\
& z p_{1,1,0,1} - p_{AB_{11}} p_{BC_{10}} p_{CD_{01}} p_{AD_{11}} \\
& z p_{1,1,1,0} - p_{AB_{11}} p_{BC_{11}} p_{CD_{10}} p_{AD_{10}} \\
& z p_{1,1,1,1} - p_{AB_{11}} p_{BC_{11}} p_{CD_{11}} p_{AD_{11}} \\
& z - p_{AB_{00}} p_{BC_{00}} p_{CD_{00}} p_{AD_{00}} - p_{AB_{00}} p_{BC_{00}} p_{CD_{01}} p_{AD_{01}} - \cdots - p_{AB_{11}} p_{BC_{11}} p_{CD_{11}} p_{AD_{11}} \\
& p_{AB_{00}} + p_{AB_{01}} + p_{AB_{10}} + p_{AB_{11}} - 1 \\
& p_{BC_{00}} + p_{BC_{01}} + p_{BC_{10}} + p_{BC_{11}} - 1 \\
& p_{CD_{00}} + p_{CD_{01}} + p_{CD_{10}} + p_{CD_{11}} - 1 \\
& p_{AD_{00}} + p_{AD_{01}} + p_{AD_{10}} + p_{AD_{11}} - 1
\end{aligned}
\tag{5}
$$

Call this ideal I_{hm}. Recall that an ideal is a set of polynomials. Of particular interest are the polynomials in I_{hm} which only involve the target distribution

probabilities, i.e. the 16 indeterminates $p_{0,0,0,0}, p_{0,0,0,1}, \ldots p_{1,1,1,1}$. This set, which is $I_{hm} \cap k[p_{0,0,0,0}, p_{0,0,0,1}, \ldots p_{1,1,1,1}]$ is also an ideal and is an example of an *elimination ideal*. This elimination ideal captures the algebraic constraints on the target distribution probabilities $p_{0,0,0,0}, p_{0,0,0,1}, \ldots p_{1,1,1,1}$. Crucially, it is possible (as will be seen in Section 4.1) to *compute* (a basis for) an elimination ideal. This process of extracting polynomial constraints on the target distribution from a set of parametric polynomials is known as *implicitisation*.

The model equivalence problem can now be posed in the language of algebraic statistics. In fact, a more general problem will be considered: that of model inclusion. Given two PRISM programs M_1 and M_2, M_2 includes M_1 ($M_1 \subseteq M_2$) if every distribution expressible by M_1 can also be expressed by M_2. Clearly two PRISM programs are equivalent if each includes the other.

Evidently, for there to be any prospect of model inclusion, the success sets of the target predicates in the two programs must be equal (modulo predicate symbol renaming). The hierarchical model and Bayesian net running examples are thus candidates for model inclusion since `hm/4` in Fig 2 and `bn/4` in Fig 4 have the same success set of 16 ground atoms.

Now consider a distribution $p_{0,0,0,0}, p_{0,0,0,1}, \ldots, p_{1,1,1,1}$ which is definable by both the hierarchical model and Bayesian net model. Seen as a point in $\mathcal{R}^{16}$ it must be in the appropriate projection of the variety generated by the ideal whose basis is (5) *and* in the appropriate projection of the variety of the ideal whose basis is given by the 25 Bayesian network polynomials (6).

$$
\begin{array}{ll}
p_{0,0,0,0} - p_{A_0} p_{BA_{00}} p_{CB_{00}} p_{DAC_{000}} & p_{A_0} + p_{A_1} - 1 \\
p_{0,0,0,1} - p_{A_0} p_{BA_{00}} p_{CB_{00}} p_{DAC_{100}} & p_{BA_{00}} + p_{BA_{10}} - 1 \\
p_{0,0,1,0} - p_{A_0} p_{BA_{00}} p_{CB_{10}} p_{DAC_{001}} & p_{BA_{01}} + p_{BA_{11}} - 1 \\
p_{0,0,1,1} - p_{A_0} p_{BA_{00}} p_{CB_{10}} p_{DAC_{101}} & p_{CB_{00}} + p_{CB_{10}} - 1 \\
p_{0,1,0,0} - p_{A_0} p_{BA_{10}} p_{CB_{01}} p_{DAC_{000}} & p_{CB_{01}} + p_{CB_{11}} - 1 \\
p_{0,1,0,1} - p_{A_0} p_{BA_{10}} p_{CB_{01}} p_{DAC_{100}} & p_{DAC_{000}} + p_{DAC_{100}} - 1 \\
p_{0,1,1,1} - p_{A_0} p_{BA_{10}} p_{CB_{11}} p_{DAC_{101}} & p_{DAC_{001}} + p_{DAC_{101}} - 1 \\
p_{1,0,0,0} - p_{A_1} p_{BA_{01}} p_{CB_{00}} p_{DAC_{010}} & p_{DAC_{010}} + p_{DAC_{110}} - 1 \\
p_{1,0,0,1} - p_{A_1} p_{BA_{01}} p_{CB_{00}} p_{DAC_{110}} & p_{DAC_{011}} + p_{DAC_{111}} - 1 \\
p_{1,0,1,0} - p_{A_1} p_{BA_{01}} p_{CB_{10}} p_{DAC_{011}} & \\
p_{1,0,1,1} - p_{A_1} p_{BA_{01}} p_{CB_{10}} p_{DAC_{111}} & \\
p_{1,1,0,0} - p_{A_1} p_{BA_{11}} p_{CB_{01}} p_{DAC_{010}} & \\
p_{1,1,0,1} - p_{A_1} p_{BA_{11}} p_{CB_{01}} p_{DAC_{110}} & \\
p_{1,1,1,1} - p_{A_1} p_{BA_{11}} p_{CB_{11}} p_{DAC_{111}} &
\end{array}
\tag{6}
$$

By simply adding the hierarchical model basis to that of the Bayesian network (thus producing a basis of 46 polynomials) we define an ideal whose variety is the set of mutually consistent hierarchical model parameters, Bayesian network

parameters and target distribution probabilities. To check for model inclusion it suffices to project this ideal down onto one set of parameters (say, those of the Bayesian network model) and see if any polynomials are in this elimination ideal which were not in the original ideal for this model. If there are, then this reveals that there is a new constraint on these parameters and thus inclusion fails. Otherwise inclusion is proven.

If model inclusion is our sole concern then ultimately only polynomials involving parameters are of interest. All the constraints between the parameters of the two models under consideration can be effected by equating target distribution probabilities. It follows that the target probability distributions can be immediately eliminated. In our running example this provides the following basis of 29 polynomials in 34 indeterminates.

(7)

$$
\begin{aligned}
&p_{A_0}p_{BA_{00}}p_{CB_{00}}p_{DAC_{000}}(p_{AB_{00}}p_{BC_{00}}p_{CD_{00}}p_{AD_{00}} + \cdots + p_{AB_{11}}p_{BC_{11}}p_{CD_{11}}p_{AD_{11}}) - p_{AB_{00}}p_{BC_{00}}p_{CD_{00}}p_{AD_{00}}\\
&p_{A_0}p_{BA_{00}}p_{CB_{00}}p_{DAC_{100}}(p_{AB_{00}}p_{BC_{00}}p_{CD_{00}}p_{AD_{00}} + \cdots + p_{AB_{11}}p_{BC_{11}}p_{CD_{11}}p_{AD_{11}}) - p_{AB_{00}}p_{BC_{00}}p_{CD_{01}}p_{AD_{01}}\\
&p_{A_0}p_{BA_{00}}p_{CB_{10}}p_{DAC_{001}}(p_{AB_{00}}p_{BC_{00}}p_{CD_{00}}p_{AD_{00}} + \cdots + p_{AB_{11}}p_{BC_{11}}p_{CD_{11}}p_{AD_{11}}) - p_{AB_{00}}p_{BC_{01}}p_{CD_{10}}p_{AD_{00}}\\
&p_{A_0}p_{BA_{00}}p_{CB_{10}}p_{DAC_{101}}(p_{AB_{00}}p_{BC_{00}}p_{CD_{00}}p_{AD_{00}} + \cdots + p_{AB_{11}}p_{BC_{11}}p_{CD_{11}}p_{AD_{11}}) - p_{AB_{00}}p_{BC_{01}}p_{CD_{11}}p_{AD_{01}}\\
&p_{A_0}p_{BA_{10}}p_{CB_{01}}p_{DAC_{000}}(p_{AB_{00}}p_{BC_{00}}p_{CD_{00}}p_{AD_{00}} + \cdots + p_{AB_{11}}p_{BC_{11}}p_{CD_{11}}p_{AD_{11}}) - p_{AB_{01}}p_{BC_{10}}p_{CD_{00}}p_{AD_{00}}\\
&p_{A_0}p_{BA_{10}}p_{CB_{01}}p_{DAC_{100}}(p_{AB_{00}}p_{BC_{00}}p_{CD_{00}}p_{AD_{00}} + \cdots + p_{AB_{11}}p_{BC_{11}}p_{CD_{11}}p_{AD_{11}}) - p_{AB_{01}}p_{BC_{10}}p_{CD_{01}}p_{AD_{01}}\\
&p_{A_0}p_{BA_{10}}p_{CB_{11}}p_{DAC_{001}}(p_{AB_{00}}p_{BC_{00}}p_{CD_{00}}p_{AD_{00}} + \cdots + p_{AB_{11}}p_{BC_{11}}p_{CD_{11}}p_{AD_{11}}) - p_{AB_{01}}p_{BC_{11}}p_{CD_{10}}p_{AD_{00}}\\
&p_{A_0}p_{BA_{10}}p_{CB_{11}}p_{DAC_{101}}(p_{AB_{00}}p_{BC_{00}}p_{CD_{00}}p_{AD_{00}} + \cdots + p_{AB_{11}}p_{BC_{11}}p_{CD_{11}}p_{AD_{11}}) - p_{AB_{01}}p_{BC_{11}}p_{CD_{11}}p_{AD_{01}}\\
&p_{A_1}p_{BA_{01}}p_{CB_{00}}p_{DAC_{010}}(p_{AB_{00}}p_{BC_{00}}p_{CD_{00}}p_{AD_{00}} + \cdots + p_{AB_{11}}p_{BC_{11}}p_{CD_{11}}p_{AD_{11}}) - p_{AB_{10}}p_{BC_{00}}p_{CD_{00}}p_{AD_{10}}\\
&p_{A_1}p_{BA_{01}}p_{CB_{00}}p_{DAC_{110}}(p_{AB_{00}}p_{BC_{00}}p_{CD_{00}}p_{AD_{00}} + \cdots + p_{AB_{11}}p_{BC_{11}}p_{CD_{11}}p_{AD_{11}}) - p_{AB_{10}}p_{BC_{00}}p_{CD_{01}}p_{AD_{11}}\\
&p_{A_1}p_{BA_{01}}p_{CB_{10}}p_{DAC_{011}}(p_{AB_{00}}p_{BC_{00}}p_{CD_{00}}p_{AD_{00}} + \cdots + p_{AB_{11}}p_{BC_{11}}p_{CD_{11}}p_{AD_{11}}) - p_{AB_{10}}p_{BC_{01}}p_{CD_{10}}p_{AD_{10}}\\
&p_{A_1}p_{BA_{01}}p_{CB_{10}}p_{DAC_{111}}(p_{AB_{00}}p_{BC_{00}}p_{CD_{00}}p_{AD_{00}} + \cdots + p_{AB_{11}}p_{BC_{11}}p_{CD_{11}}p_{AD_{11}}) - p_{AB_{10}}p_{BC_{01}}p_{CD_{11}}p_{AD_{11}}\\
&p_{A_1}p_{BA_{11}}p_{CB_{01}}p_{DAC_{010}}(p_{AB_{00}}p_{BC_{00}}p_{CD_{00}}p_{AD_{00}} + \cdots + p_{AB_{11}}p_{BC_{11}}p_{CD_{11}}p_{AD_{11}}) - p_{AB_{11}}p_{BC_{10}}p_{CD_{00}}p_{AD_{10}}\\
&p_{A_1}p_{BA_{11}}p_{CB_{01}}p_{DAC_{110}}(p_{AB_{00}}p_{BC_{00}}p_{CD_{00}}p_{AD_{00}} + \cdots + p_{AB_{11}}p_{BC_{11}}p_{CD_{11}}p_{AD_{11}}) - p_{AB_{11}}p_{BC_{10}}p_{CD_{01}}p_{AD_{11}}\\
&p_{A_1}p_{BA_{11}}p_{CB_{11}}p_{DAC_{011}}(p_{AB_{00}}p_{BC_{00}}p_{CD_{00}}p_{AD_{00}} + \cdots + p_{AB_{11}}p_{BC_{11}}p_{CD_{11}}p_{AD_{11}}) - p_{AB_{11}}p_{BC_{11}}p_{CD_{10}}p_{AD_{10}}\\
&p_{A_1}p_{BA_{11}}p_{CB_{11}}p_{DAC_{111}}(p_{AB_{00}}p_{BC_{00}}p_{CD_{00}}p_{AD_{00}} + \cdots + p_{AB_{11}}p_{BC_{11}}p_{CD_{11}}p_{AD_{11}}) - p_{AB_{11}}p_{BC_{11}}p_{CD_{11}}p_{AD_{11}}\\
&p_{A_0} + p_{A_1} - 1\\
&p_{BA_{00}} + p_{BA_{10}} - 1\\
&p_{BA_{01}} + p_{BA_{11}} - 1\\
&p_{CB_{00}} + p_{CB_{10}} - 1\\
&p_{CB_{01}} + p_{CB_{11}} - 1\\
&p_{DAC_{000}} + p_{DAC_{100}} - 1\\
&p_{DAC_{001}} + p_{DAC_{101}} - 1\\
&p_{DAC_{010}} + p_{DAC_{110}} - 1\\
&p_{DAC_{011}} + p_{DAC_{111}} - 1\\
&p_{AB_{00}} + p_{AB_{01}} + p_{AB_{10}} + p_{AB_{11}} - 1\\
&p_{BC_{00}} + p_{BC_{01}} + p_{BC_{10}} + p_{BC_{11}} - 1\\
&p_{CD_{00}} + p_{CD_{01}} + p_{CD_{10}} + p_{CD_{11}} - 1\\
&p_{AD_{00}} + p_{AD_{01}} + p_{AD_{10}} + p_{AD_{11}} - 1
\end{aligned}
$$

So the question is: does the basis (7) define an ideal containing 'new' constraints on either set of parameters. To answer this question, consideration of Gröbner bases is required.

4.1 Gröbner bases

To explain Gröbner bases the key concept of *term orderings* must be given. Here the presentation given in [1] is followed. Essentially, a term ordering can be seen as a kind of complexity ordering on the terms of polynomials. Denote an arbitrary term $x_1^{\beta_1}, \ldots, x_n^{\beta_n}$ as $\mathbf{x}^\beta$, then a term ordering is a total order on terms such that:

1. $1 < \mathbf{x}^\beta$ for all terms $\mathbf{x}^\beta \neq 1$;
2. If $\mathbf{x}^\alpha < \mathbf{x}^\beta$, then $\mathbf{x}^\alpha \mathbf{x}^\gamma < \mathbf{x}^\beta \mathbf{x}^\gamma$ for all terms $\mathbf{x}^\gamma$.

For example, if x and y are the only variables, then

$$1 < x < x^2 < x^3 < \cdots < y < xy < \cdots < y^2 < \ldots$$

is a term ordering: a *lexicographical ordering*. Here is another lexicographical ordering with the roles of x and y reversed

$$1 < y < y^2 < y^3 < \cdots < x < xy < \cdots < x^2 < \ldots$$

There are also *degree lexicographical orders* such as:

$$1 < x < y < x^2 < xy < y^2 < x^3 < x^2y < xy^2 < y^3 < \ldots$$

where ordering is by degree and terms of the same degree are ordered lexicographically. These examples are taken from [1] where a full account of term orderings can be found. For current purposes a particularly important class of term orderings are elimination orderings:

Definition 3. *For X_1, X_2 power products in the x indeterminates $\{x_1, \ldots x_n\}$ and Y_1, Y_2 power products in the y indeterminates $\{y_1, \ldots y_m\}$, we define*

$$X_1Y_1 < X_2Y_2 \Leftrightarrow \begin{cases} X_1 <_x X_2 \\ \textit{or} \\ X_1 = X_2 \textit{ and } Y_1 <_y Y_2 \end{cases}$$

where $<_x$ and $<_y$ are term orderings on the x and y variables. This term order is called an elimination order *with the x variables larger than the y variables.*

A proper account of Gröbner bases will not be given here—for such an account see [1]—instead the bare minimum for the analysis of PRISM model inclusion will be given. The definition of a Gröbner bases uses the notion of a term ordering. In particular, note that, given a term ordering, any non-zero polynomial $f \in k[x_1, \ldots x_s]$ can be written as

$$f = a_1\mathbf{x}^{\alpha_1} + a_2\mathbf{x}^{\alpha_2} + \cdots + a_r\mathbf{x}^{\alpha_r}$$

where $\mathbf{x}^{\alpha_1} > \mathbf{x}^{\alpha_2} > \cdots > \mathbf{x}^{\alpha_r}$ according to the given term ordering. $a_1\mathbf{x}^{\alpha_1}$ is known as the *leading term* of f (and denoted $\mathrm{lt}(f)$) and $\mathbf{x}^{\alpha_1}$ is called the *leading power product* of f (and denoted $\mathrm{lp}(f)$).

Definition 4. *A set of non-zero polynomials* $G = \{g_1, \ldots, g_t\}$ *contained in an ideal* I *is called a* Gröbner basis *for* I *if and only if for all* $f \in I$ *such that* $f \neq 0$*, there exists* $i \in \{1, \ldots, t\}$ *such that* $\mathrm{lp}(g_i)$ *divides* $\mathrm{lp}(f)$*. [1]*

An example from [9] illustrates the point that a Gröbner basis is defined in terms of a term ordering

Example 1. The Gröbner basis of $\langle x_1^2 - 2x_1x_3 + 5, x_1x_2^2 + x_2x_3^3, 3x_2^2 - 8x_3^3\rangle \in \mathcal{Q}[x_1, x_2, x_3]$ with respect to a lexicographical ordering with the indeterminates ordered as $x_2 > x_1 > x_3$ is:

$$\begin{aligned}
&3x_2^2 - 8x_3^3,\\
&80x_2x_3^3 - 3x_3^8 + 32x_3^7 - 40x_3^5,\\
&x_1^2 - 2x_1x_3 + 5,\\
&-96x_3^7 + 9x_3^8 + 120x_3^5 + 640x_3^3x_1,\\
&240x_3^6 + 1600x_3^3 - 96x_3^8 + 9x_3^9
\end{aligned}$$

but with a degree reverse lexicographical ordering (again with $x_2 > x_1 > x_3$) the Gröbner basis is

$$\begin{aligned}
&x_1^2 - 2x_1x_3 + 5,\\
&-3x_2^2 + 8x_3^3,\\
&8x_1x_2^2 + 3x_2^2
\end{aligned}$$

To illustrate the notion of one leading power product dividing another as mentioned in the definition of Gröbner basis, note that according to the first (purely lexicographical) ordering in Example 1, the leading power products of $x_1^2 - 2x_1x_3 + 5, x_1x_2^2 + x_2x_3^3$ and $3x_2^2 - 8x_3^3$ are $x_1^2, x_1x_2^2$ and x_2^2, respectively. These are divisible by x_1^2, x_2^2 and x_2^2, respectively, the leading power products of Gröbner basis polynomials $x_1^2 - 2x_1x_3 + 5$, $3x_2^2 - 8x_3^3$ and $3x_2^2 - 8x_3^3$, respectively. In contrast, for the second degree-based term ordering from Example 1, the leading power products of $x_1^2 - 2x_1x_3 + 5, x_1x_2^2 + x_2x_3^3$ and $3x_2^2 - 8x_3^3$ are $x_1x_3, x_2x_3^3$ and x_3^3, respectively. These are divisible by x_1x_3, x_3^3 and x_3^3 respectively, the leading power products of Gröbner basis polynomials $x_1^2 - 2x_1x_3 + 5$, $-3x_2^2 + 8x_3^3$ and $3x_2^2 - 8x_3^3$ respectively.

Gröbner bases have many useful properties. For example using the polynomial division algorithm a Gröbner basis can be used to check whether any given polynomial is in the ideal generated by the Gröbner basis. For current purposes the utility of Gröbner bases is due to the following theorem:

Theorem 1. *Let* I *be a non-zero ideal of* $k[y_1, \ldots y_m, x_1, \ldots x_n]$*, and let* $<$ *be an elimination order with the* x *indeterminates larger than the* y *indeterminates. Let* $G = \{g_1, \ldots, g_t\}$ *be a Gröbner basis for this ideal. Then* $G \cap k[y_1, \ldots, y_m]$ *is a Gröbner basis for the ideal* $I \cap k[y_1, \ldots, y_m]$*. [1]*

This is the key theorem which provides a solution to the PRISM model inclusion problem in the finite case. Returning to our running example of the hierarchical model and Bayesian network model, consider again the polynomials in (7) and the ideal (call it $I_{\mathrm{bn=hm}}$) they generate. Using an elimination ordering with the hierarchical model parameters larger than those of the Bayesian net parameters a Gröbner basis G for $I_{\mathrm{bn=hm}}$ can then be computed. Theorem 1 states that the polynomials in G which only involve Bayesian network parameters form a Gröbner basis for the ideal encapsulating all the constraints on these Bayesian network parameters. If a polynomial not implied by the original constraints on Bayesian network parameters is found then model inclusion fails, otherwise it succeeds.

The good news is that (1) there is an algorithm—Buchberger's algorithm—for computing Gröbner bases for any given term ordering and (2) this algorithm is available in a number of algebraic software packages, such as Maple. Putting our running example to one side temporarily, this good news will be illustrated with the following very simple example of model inclusion. Consider 2 binary variables A and B and the two models: the *saturated* model M_1 which contains all possible joint distributions over A and B and the *independence* model M_2 which includes only those distributions in which A and B are independent. (There is no need to go to the bother of encoding these models as PRISM programs, although this is easily done.) Clearly $M_2 \subset M_1$ but $M_1 \not\subset M_2$. This can be established using algebraic statistics using e.g. Maple. In the independence model let `a0, a1, b0, b1` denote $P(A = 0), P(A = 1), P(B = 0), P(B = 1)$, respectively; and in the saturated model let `ab00, ab01, ab10, ab11` denote $P(A = 0, B = 0), P(A = 0, B = 1), P(A = 1, B = 0), P(A = 1, B = 1)$, respectively. Then `simple` denotes the 'model equivalence ideal' in Fig 5.

```
with(PolynomialIdeals);
simple := <a0+a1-1, b0+b1-1, ab00+ab01+ab10+ab11-1, a0*b0-ab00,
a0*b1-ab01, a1*b0-ab10, a1*b1-ab11>
EliminationIdeal(simple, {a0, a1, b0, b1})
                   <a0 + a1 - 1, b0 + b1 - 1>
EliminationIdeal(simple, {ab00, ab01, ab10, ab11})
<                                                  2
 ab00 + ab01 + ab10 + ab11 - 1, ab00 ab10 + ab00  + ab01 ab10 + ab00 ab01 - ab00 >
```

Fig. 5. Using Maple to establish model (non)-inclusion

Using the Maple built-in `EliminationIdeal` (which works by computing an appropriate Gröbner basis), it is found that no new constraints are put on `a0, a1, b0, b1` by asserting model equivalence thus proving that $M_2 \subset M_1$ but that a new constraint represented by the polynomial $ab00ab10 + ab00^2 + ab01ab10 + ab00ab01 - ab00$ is effected on the saturated model parameters thus showing that $M_1 \not\subset M_2$.

The bad news is that computation of Gröbner bases can be tremendously computationally expensive. An attempt to compute a Gröbner basis for the polynomials in (7) using an elimination term ordering with Maple was abandoned after a few hours of computation. As noted in [1] "the computational complexity of Buchberger's algorithm often makes it difficult to compute a Gröbner basis for even small problems". Although improving Buchberger's algorithm is an area of active research, there is the basic problem that Gröbner bases can have polynomials with very many terms making it difficult to compute them efficiently. Taking an example from [1] the (unthreatening-looking) ideal $\langle x^7 + xy + y, y^5 + yz + z, z^2 + z + 1\rangle$ has a Gröbner basis (for a particular lexicographical term ordering) of 3 polynomials with 58, 70 and 35 terms, respectively.

Another problem, of course, is actually forming the polynomials such as (7) in the first place. In the finite case this can at least be done: for each model, for each atom in the target predicate's success set one can use PRISM's built-in `probf` predicate to get the needed polynomial for that model and atom. Once this has been done a polynomial can be formed for each success set atom by putting the polynomials from each model on opposite sides of a minus sign—as was done in (7). If one or both models are 'failure' models then normalisation must be taken into account; note that this has been done in (7) by multiplying each Bayesian network polynomial by $(p_{AB_{00}}p_{BC_{00}}p_{CD_{00}}p_{AD_{00}} + \cdots + p_{AB_{11}}p_{BC_{11}}p_{CD_{11}}p_{AD_{11}})$ which is the expression for the Z normalising constant for the hierarchical model. Eventually a finite set of polynomials are formed and an elimination ideal can be found by Gröbner basis computation and checked for new constraints (=polynomials).

This is an unwieldy business, and when the success set is infinite such an approach will not work. In the infinite case it is at least possible to compute elimination ideals for finite subsets of the full infinite set of polynomials. If such an ideal contains a new polynomial then model inclusion is disproven; but if not it may be that a bigger finite subset will generate a new polynomial. In the infinite case, Hilbert's Theorem tell us that there is a *finite* basis for the infinite set of polynomials defined by an infinite success set, but is not clear how this would be found.

5 Shortcuts to deciding model inclusion

In the running example since both models are graphical, knowledge of conditional independence relations in such models can be used to provide a direct answer to the question of model inclusion. The structure of the hierarchical model as given in Fig 1 makes it evident that only distributions where $A \perp C|\{B, D\}$ can be members of this model, whereas inspection of Fig 3 makes it evident that distributions not meeting this constraint *are* permitted by the Bayesian network. It follows that the polynomials in (7) must generated an ideal containing a polynomial only involving BN parameters which was not in the original ideal for the BN.

In fact, it is not too difficult to construct such a polynomial. If $A \perp C|\{B, D\}$ then, for example,

$$P(A = 0|B = 0, D = 0)P(C = 0|B = 0, D = 0) = P(A = 0, C = 0|B = 0, D = 0)$$

from which it follows that

$$P(A = 0, B = 0, D = 0)P(C = 0, B = 0, D = 0) - P(A = 0, C = 0, B = 0, D = 0)P(B = 0, D = 0) = 0$$

or using our abbreviated notation

$$\begin{aligned}&(p_{0,0,0,0} + p_{0,0,1,0})(p_{0,0,0,0} + p_{1,0,0,0}) - p_{0,0,0,0}(p_{0,0,0,0} + p_{0,0,1,0} + p_{1,0,0,0} + p_{1,0,1,0}) = 0\\ \Leftrightarrow\ &p_{0,0,1,0}p_{1,0,0,0} - p_{0,0,0,0}p_{1,0,1,0} = 0\end{aligned}$$

The LHS of this second polynomial is (1) contained in the hierarchical model ideal I_{hm}, but (2) not in the Bayesian net ideal I_{bn}. (1) is easy to check: for example, the 4 target distribution probabilities above can be re-expressed in terms of the HM parameters and z. It is then obvious that $p_{0,0,1,0}p_{1,0,0,0} - p_{0,0,0,0}p_{1,0,1,0} = 0$. Any easy way to check (2) is to re-express $p_{0,0,1,0}p_{1,0,0,0} - p_{0,0,0,0}p_{1,0,1,0}$ in terms of the BN parameters:

$$\begin{aligned}&p_{A_0}p_{BA_{00}}p_{CB_{10}}p_{DAC_{001}} \times p_{A_1}p_{BA_{01}}p_{CB_{00}}p_{DAC_{010}}\\ -\ &p_{A_0}p_{BA_{00}}p_{CB_{00}}p_{DAC_{000}} \times p_{A_1}p_{BA_{01}}p_{CB_{10}}p_{DAC_{011}}\end{aligned}$$

or more simply:

$$p_{A_0}p_{A_1}p_{BA_{00}}p_{BA_{01}}(p_{DAC_{001}}p_{DAC_{010}} - p_{DAC_{000}}p_{DAC_{011}}) \tag{8}$$

and then check that this polynomial is not a member of the ideal generated by the original BN parameter constraints. Fig 6 shows this being done using Maple.

```
with(PolynomialIdeals);
> bnsw := <pa0 + pa1 - 1, pba01 + pba11 - 1, pdac001 + pdac101 - 1,
  pcb01 + pcb11 - 1, pcb00 + pcb10 - 1, pdac010 + pdac110 - 1,
  pdac011 + pdac111 - 1, pdac000 + pdac100 - 1, pba00 + pba10 - 1>
> newcons := [pa0*pa1*pba00*pba01*(pdac001*pdac010-pdac000*pdac011)]
> IdealMembership(newcons, bnsw)
                 false
```

Fig. 6. Proving that a polynomial is not a member of an existing ideal using Maple

Turning now to model inclusion note that $M_1 \subset M_2$ if for any parameter setting for M_1 there is a parameter setting for M_2 that gives the same

distribution. It follows that model inclusion can be established if such a map can be found. In our running example, this is easily done. Any parameter setting for the hierarchical model defines a particular joint distribution over the variables A, B, C and D which in turn determines conditional distributions $P(A), P(B|A), P(C|A, B)$ and $P(D|A, C)$ which are precisely the required parameters for the Bayesian network. This is the required parameter mapping. For example consider $P(B = 0|A = 0)$ which is abbreviated to $p_{BA_{00}}$. In terms of target distribution probabilities we have

$$p_{BA_{00}} = \frac{p_{0,0,0,0} + p_{0,0,0,1} + p_{0,0,1,0} + p_{0,0,1,1}}{p_{0,0,0,0} + p_{0,0,0,1} + p_{0,0,1,0} + p_{0,0,1,1} + p_{0,1,0,0} + p_{0,1,0,1} + p_{0,1,1,0} + p_{0,1,1,1}}$$

and in terms of the hierarchical model parameters $p_{BA_{00}} = (1 + T)^{-1}$ where T is:

$$\frac{p_{AB_{01}}p_{BC_{10}}p_{CD_{00}}p_{AD_{00}} + p_{AB_{01}}p_{BC_{10}}p_{CD_{01}}p_{AD_{01}} + p_{AB_{01}}p_{BC_{11}}p_{CD_{10}}p_{AD_{00}} + p_{AB_{01}}p_{BC_{11}}p_{CD_{11}}p_{AD_{01}}}{p_{AB_{00}}p_{BC_{00}}p_{CD_{00}}p_{AD_{00}} + p_{AB_{00}}p_{BC_{00}}p_{CD_{01}}p_{AD_{01}} + p_{AB_{00}}p_{BC_{01}}p_{CD_{10}}p_{AD_{00}} + p_{AB_{00}}p_{BC_{01}}p_{CD_{11}}p_{AD_{01}}}$$

It is clear that each Bayesian network parameter can be expressed as a function of the hierarchical model parameters in this way. To check that the same distribution is defined by both sets of parameters it suffices for each of the 16 target probabilities:

1. To find its expression in terms of BN parameters (these are all monomials of 4 terms, see (6)).
2. and then to replace each BN parameter in this expression by the corresponding expression in terms of hierarchical model parameters (e.g. each occurrence of $p_{BA_{00}}$ is replaced by the expression described immediately above).
3. and then prove that this expression is equivalent to the one used to define the target probability in the hierarchical model.

If a rational function is used to map between parameter spaces this check can be done using ideals. From such a map a collection of polynomials can be generated. For example, from above we have that:

$$(X + Y)p_{BA_{00}} - Y = 0$$

where

$$X = p_{AB_{01}}p_{BC_{10}}p_{CD_{00}}p_{AD_{00}} + p_{AB_{01}}p_{BC_{10}}p_{CD_{01}}p_{AD_{01}} + p_{AB_{01}}p_{BC_{11}}p_{CD_{10}}p_{AD_{00}} + p_{AB_{01}}p_{BC_{11}}p_{CD_{11}}p_{AD_{01}}$$

and

$$Y = p_{AB_{00}}p_{BC_{00}}p_{CD_{00}}p_{AD_{00}} + p_{AB_{00}}p_{BC_{00}}p_{CD_{01}}p_{AD_{01}} + p_{AB_{00}}p_{BC_{01}}p_{CD_{10}}p_{AD_{00}} + p_{AB_{00}}p_{BC_{01}}p_{CD_{11}}p_{AD_{01}}$$

So a polynomial can be generated for each BN parameter. Recall that each such polynomial can be interpreted as a constraint between the indeterminates involved. To do the check above we add each of these polynomials to the hierarchical model ideal I_{hm} and then see if the resulting ideal contains the BN ideal whose basis is given in (6). In essence, we just check that the constraints on BN parameters are sufficiently strong so that they define exactly the same distribution as the hierarchical model parameters.

6 Future work

Hopefully, the current report provides sufficient evidence that algebraic statistics is at least worth exploring as a mathematical tool for the analysis of PRISM programs and closely related formalisms such as SLPs and ICL. Due to the generality of the approach it is likely that it will be useful in the analysis of other SRL formalisms too.

However, there are serious problems to address. Firstly, the approach taken here threw away the compactness which a first-order representation affords. By effectively 'grounding out', large numbers of structurally similar polynomials (with many indeterminates) were produced. In the infinite case no satisfactory solution was found. Since there exists work connecting algebraic geometry (and Gröbner base computation in particular) with first-order (and propositional) theorem proving [10,11,12] there is hope that a 'more first-order' approach can be found. On a related point, the second problem is that structure of the ideals produced was not really exploited. Our two running examples were both PRISM encodings of graphical models and as such their ideals are *toric ideals* [13] which have special properties. Although PRISM programs are such a general class it will be possible to construct models with little exploitable structure, when structure is there it should be exploited. A third issue is that no proper connection between the structure of the logic program and the nature of the ideals produced has been given.

References

1. Adams, W.W., Loustauna, P.: An Introduction to Gröbner Bases. Volume 3 of Graduate Studies in Mathematics. American Mathematical Society, Providence, RI, USA (1994)
2. Sato, T., Kameya, Y.: Parameter learning of logic programs for symbolic-statistical modeling. Journal of Artificial Intelligence Research **15** (2001) 391–454
3. Sato, T., Kameya, Y., Zhou, N.F.: Generative modeling with failure in PRISM. In: Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI-05), Edinburgh (2005)
4. Lauritzen, S.L.: Graphical Models. Oxford University Press, Oxford (1996)
5. Poole, D.: Probabilistic Horn abduction and Bayesian networks. Artificial Intelligence **64** (1993) 81–129
6. Pearl, J.: Causality: Models, Reasoning and Inference. Cambridge University Press (2000)
7. Muggleton, S.: Stochastic logic programs. In De Raedt, L., ed.: Advances in Inductive Logic Programming. Volume 32 of Frontiers in Artificial Intelligence and Applications. IOS Press, Amsterdam (1996) 254–264
8. Cussens, J.: Parameter estimation in stochastic logic programs. Machine Learning **44** (2001) 245–271
9. Pistone, G., Riccomagno, E., Wynn, H.P.: Algebraic statistics: computational commutative algebra in statistics. Chapman & Hall, New York (2001)
10. Wu, J.: First-order polynomial based theorem proving. In Gao, X.S., Wang, D., eds.: Mathematics Mechanizations and Applications. Academic Press, San Diego, USA (2000) 273–294

11. Kapur, D., Narendran, P.: An equational approach to theorem proving in first-order predicate calculus. SIGSOFT Softw. Eng. Notes **10** (1985) 63–66
12. Clegg, M., Edmonds, J., Impagliazzo, R.: Using the groebner basis algorithm to find proofs of unsatisfiability. In: STOC '96: Proceedings of the twenty-eighth annual ACM symposium on Theory of computing, New York, NY, USA, ACM Press (1996) 174–183
13. Geiger, D., Meek, C., Sturmfels, B.: On the toric algebra of graphical models. The Annals of Statistics **34** (2006) 1463–1492