MATHEMATICA APPLICANDA
Vol. 40(1) 2012, p. 53–103

Timo J. T. Koski (Stockholm)
John M. Noble (Warsaw)

A Review of Bayesian Networks and Structure Learning

Abstract This article reviews the topic of Bayesian networks. A Bayesian network is a factorisation of a probability distribution along a directed acyclic graph. The relation between graphical *d*-separation and independence is described. A short article from 1853 by Arthur Cayley [8] is discussed, which contains several ideas later used in Bayesian networks: factorisation, the noisy 'or' gate, applications of algebraic geometry to Bayesian networks. The ideas behind Pearl's intervention calculus when the DAG represents a *causal* dependence structure and the relation between the work of Cayley and Pearl is commented on.

Most of the discussion is about structure learning, outlining the two main approaches, search and score versus constraint based. Constraint based algorithms often rely on the assumption of *faithfulness*, that the data to which the algorithm is applied is generated from distributions satisfying a faithfulness assumption where graphical *d*-separation and independence are equivalent. The article presents some considerations for constraint based algorithms based on recent data analysis, indicating a variety of situations where the faithfulness assumption does not hold. There is a short discussion about the causal discovery controversy, the idea that *causal* relations may be learned from data.

2010 Mathematics Subject Classification: 62H05; 68T37; 65S05.

Key words and phrases: Bayesian networks, directed acyclic graph, Arthur Cayley, intervention calculus, graphical Markov model, Markov equivalence, structure learning.

1. Introduction.

1.1. Background and motivation. The models that were later to be called Bayesian networks were introduced in 1982 into artificial intelligence by J. Pearl [63], a seminal article in the literature of that field. A Bayesian network provides a compact representation of a probability distribution that is too large to handle using traditional specifications and provides a systematic and localised method for incorporating probabilistic information about a situation. The description 'Bayesian networks' covers a large field of problems and techniques of data analysis and probabilistic reasoning, where data is collected on a large number of variables and the aim is to factorise the distribution, represent it graphically and exploit the graphical representation. Perhaps the earliest work that explicitly uses directed graphs to represent possible dependencies among random variables is that from 1921 by S. Wright [93], developed by the same author in 1934 (see [94]).

Bayesian networks only represent a small part of the wider field of graphical models; a Bayesian network is a probability distribution factorised along a directed acyclic graph (henceforth, written DAG). In many examples this is not the most efficient model for representing the independence structure; the wider field of graphical models and learning theory is not dealt with here and the reader is referred to Studenỳ [88]. The book by Koller and Friedman [46] is an extensive text on graphical models.

Situations where Bayesian networks provide the natural tools for analysis are, for example: computing the overall reliability of a system given the reliability of the individual components and how they interact (for example, Langseth and Portinale [49]), system security (for example, Zhang and Song [98], where they use Bayesian networks as a tool for assessing intrusion evidence and whether a network is under attack) and forensic analysis (for example, Dawid et. al. [21]). Further applications are, for example: finding the most likely message that was sent across a noisy channel, restoring a noisy image, mapping genes onto a chromosome. One of the leading applications of techniques from the area is to establishing genome pathways. Given DNA hybridization arrays, which simultaneously measure the expression levels for thousands of genes, a major challenge in computational biology is to uncover, from such measurements, gene/protein interactions and key biological features of cellular systems. This is discussed, for example, by Nir Friedman et. al. in [30] and references to applications of Bayesian networks in reconstructing cellular co-expression networks in biology are found in Markowetz and Spang [57]. DAGs have also proved useful in a large number of situations where the graph is constructed along causal principles; parent variables are considered to be direct causes. One field where causal networks have proved particularly effective has been epidemiological research, where DAGs have provided a framework for the problem of multiple confounding factors in genetic epidemiology, as discussed by Greenland, Pearl and Robins [34]. Bayesian networks offer an alternative to 'naïve Bayes' models of supervised classification in machine learning, which exploits more of the structure (see Ekdahl and Koski [25]). Additional applications to medical diagnosis, clinical decision support, crime factor analysis, sensor validation, information retrieval, credit-rating, risk management and robotics are found in Pourret, Nam, Naïm and Marcot [75].

1.2. Organisation of the article. The article is organised as follows: the section 1.1 is introductory, pointing out areas where Bayesian networks and graphical models have had fruitful applications. The section 1.2 describes the organisation of the article. The section 1.3 gives the basic definition of a Bayesian network and outlines the main problems of learning. This leads to the section 1.4, describing the problem of *product approximations of high dimensional probability distributions* and their relation to Bayesian networks. The section 1.5 outlines the very important area of *causal* networks, situations where the directions of the arrows of a DAG have a cause to effect interpretation. The section 1.6 outlines the topic of *dynamic* Bayesian networks, where the variables can be grouped into time slices and some of the arrows indicate the progression between the time slices.

The section 2 describes the basic separation properties in DAGs, known as d-separation, and how they relate to the conditional independence structure of a Bayesian network. The section 2.1 gives some basic definitions, illustrated when the DAG has a causal interpretation. The section 2.2 describes the Aalborg algorithm for updating a probability distribution when hard evidence is received, while the section 2.3 discusses two refinements for a Bayesian network, *context specific independence* and *minimal sufficient causation* which are able to incorporate more of the independence structure.

The section 3 discusses the pioneering work from 1853 by Arthur Cayley [8], who presents a *causal* probability network and a *noisy 'or' gate*. The section 4 discusses the relation between the work of Arthur Cayley and Judea Pearl's intervention calculus, which is described. The section 5 describes the use of algebraic geometry as a tool in Bayesian networks, a subject with which Arthur Cayley was familiar, in the context of [8].

The section 6 discusses the problem of learning. The problem of learning the parameters for a given network is touched on briefly in the section 6.1. The section 6.2 describes the problem of *structure learning*, learning the independence structure from data. There are three basic methods; *search-and-score*, *constraint based* and *hybrid*. The section 6.3 gives an account of the *search and score* algorithms, describing methods of scoring, the *Cooper Herzkovitz likelihood*, Bayesian approaches and Bayesian Information Criterion, along with others, and a brief outline of the approaches to searching the space; Markov chain Monte Carlo, Sparse Candidate, Optimal Reinsertion, Greedy Equivalence Search. The *greedy equivalence search* algorithm, while more economical than other search and score algorithms, requires a *composition* assumption to return the correct graph, which is not required of the other search and score algorithms discussed here.

In the section 6.4, the discussion then moves onto *constraint based* algorithms, which are in general more economical, but which usually *require* a faithfulness assumption, that there exists a DAG whose d-separation statements are equivalent to the conditional independence statements of the distribution. An example is given to show where *faithfulness* and *composition* fail and where algorithms on these principles will return the wrong structure. Various examples of constraint based algorithms are discussed, the *Chow-Liu tree*, *three phase dependency analysis*, the *PC and MPCC* algorithms, the FAST algorithm and RAI algorithm. The Chow-Liu tree locates a maximal dependency tree, while the others work on the principle of adding an edge $X \sim Y$ if and only if there does not exist a set S such that $X \perp Y|S$ (X conditionally independent of Y given a set of variables S). Under a faithfulness assumption, together with a perfect oracle, these produce a correct graph. The Xie - Geng algorithm, which firstly locates the *independence* graph and then pulls it apart to locate the immoralities, is also outlined; this algorithm also requires faithfulness.

The section 6.5 moves on to *hybrid* algorithms, which contain features from both constraint based and search and score approaches. The increasingly important technique of L^1 regularisation is mentioned, the addition of an L^1 component to the score function helps to reduce the overall number of parameters in the output model.

The largest problem with many of the approaches used is the assumption of faithfulness and the section 6.6 discusses the pitfalls of this assumption. Algorithms that work well when applied to simulated data, simulated from a distribution which has a faithful graphical model, do not necessarily function well on 'real world' data sets. One of the problems is the presence of *hidden variables* - even if there were a faithful distribution in principle, if common causes are hidden, the distribution over the observable variables may not be faithful. Various situations where the faithfulness assumption may fail, and the resulting contradictions produced by constraint based methods are indicated.

The section 7 discusses the 'causal discovery' controversy, the idea that a directed arrow that has a valid interpretation of direct cause to effect can be learned from data and also how, if it is possible to carry out controlled experiments, the contradictions can be resolved by the data produced by controlled experiments along with Judea Pearl's intervention calculus. The important work of Freedman and Humphreys [28] is discussed. The section 8 gives a conclusion, which is followed by an extensive bibliography.

1.3. Description of a Bayesian network. A Bayesian network consists of a *factorisation* of a probability distribution and a DAG corresponding to the factorisation. Those conditional independence statements that may be inferred directly from the factorisation correspond to d-separation statements in the DAG and therefore some key conditional independence statements in the probability distribution can be located using graphical techniques for locating d-separation.

Consider a set of random variables $V = \{X_1, \ldots, X_d\}$. It may be ordered in $d!$ ways and each ordering gives rise to a different *factorisation* of the joint probability distribution. Throughout, p_X will be used to denote the probability distribution over a variable, or set of variables X, while $p_{X|Z}$ denotes the conditional probability function of X given Z. The factorisation according to the ordering σ is the representation

$$p_{X_1,\ldots,X_d} = \prod_{j=1}^{d} p_{X_{\sigma(j)}|\Pi_j^\sigma}, \tag{1}$$

where, for each j, $\Pi_j^\sigma \subset \{X_{\sigma(1)}, \ldots, X_{\sigma(j-1)}\}$ and is the smallest such subset for which the formula (1) holds. In the DAG corresponding to this factorisation, the parent set for variable $X_{\sigma(j)}$ is Π_j^σ.

The notation $X \perp Y|Z$ will denote X conditionally independent of Y given Z. The notation $X \not\perp Y|Z$ denotes the negation, that the variables are not conditionally independent.

Bayesian networks deal principally with discrete variables where the distribution, conditioned on the parent set, is multinomial, or with multivariate normal distributions, where the analysis basically extends to correlation. Conditional Gaussian distributions are also considered, where the distribution, conditioned on the discrete variables, is multivariate normal. Some attempts have also been made to deal with other distributions; for example, *skew-normal* by Capitanio, Azzalini and Stanghellini [6].

In this article, attention is predominantly given to the setting where the variables are multinomial.

Broadly speaking, three categories of inference problems are of interest:

1. For a given probability distribution with a given factorisation, compute the conditional probability distribution over the remaining variables when some variables are observed, or *instantiated*.

2. For a given factorisation, representing an independence structure, estimate $p_{X_{\sigma(j)}|\Pi_j^\sigma}$; that is, estimate the parameters that describe the conditional probability distribution of the variable $X_{\sigma(j)}$, conditioned on its parent set Π_j^σ, from the available data, for $j = 1, \dots, d$.

3. Learn the independence structure and, from this, find a suitable factorisation. Having located the independence structure, the key problem is to find an efficient ordering of the variables so that the conditioning sets are kept as small as possible.

The name 'Bayesian networks' derives from the first of the inference problems listed. Although Bayesian statistical techniques are frequently used in the analysis for the second, the term *Bayesian* in the context of 'Bayesian networks' originates, rather, from the non-controversial probabilistic use of Bayes rule and not from the statistical meaning where a probability distribution is placed over the parameter space. The term was first used by Judea Pearl [64] to emphasise the reliance on Bayes conditioning when carrying out inference using a Bayesian network. When a causal network is in view, the leaf variables are observable and Bayes rule is used to update the probability distribution over the hidden variables, given information on the leaf variables.

1.4. Bayesian Network as Product Approximation of High Dimensional Discrete Probability Distributions. The topic of storing a high dimensional discrete probability distribution (in a digital medium) was probably introduced in 1959 into the journal literature by P.M. Lewis II [52]. If it should not be possible to store the whole distribution, the idea suggested and analysed by Lewis was a product approximation of the discrete probability distribution, see e.g. [35]. The problem of storing probability distributions is another expression of the 'curse of dimensionality'. For an intuitive statement of the issues involved we quote P.M. Lewis II in [52, p.220]:

> A product approximation is defined to be an approximation to a higher order distribution made up of a product of several of its lower order component distributions such that the product is an extension of the lower order distributions.

By 'extension', Lewis means that the lower order component distributions can be obtained by marginalisation from the product and that the product is a probability distribution. A product of an arbitrary set of lower dimensional probability distributions will not satisfy these requirements. The task is a special case of the *classical marginal problem.* Let $\underline{X} = (X^1, \dots, X^d)$ denote the random vector under consideration and $V = \{X^1, \dots, X^d\}$ the set of variables. Let $\mathcal{X}_j$ denote the state space of X^j and let $\mathcal{X} = \prod_{j=1}^d \mathcal{X}_j$ denote the state space of $\underline{X}$. Let $\mathcal{X}_W$ denote the state space of a subset $W \subset V$ of the variables. Let us consider a family of non-empty subsets $\{W_l\}_{l=1}^s$ of V that satisfy $V = \bigcup_{l=1}^s W_l$; $\mathcal{X} = \times_{l=1}^s \mathcal{X}_{W_l}$.

For each $l \in \{1, \dots, s\}$ let $\mathbb{P}_{W_l}$ denote a measure over $\mathcal{X}_{W_l}$. The *classical marginal problem* concerns the existence of a Borel measure $\mathbb{P}^*$ on $\mathcal{X}$ such that for all W_l

$$\mathbb{P}_{W_l} = (\mathbb{P}^*)^{\downarrow W_l}, \tag{2}$$

where $(\mathbb{P}^*)^{\downarrow W_l}$ denotes the marginalization of $\mathbb{P}$ down to W_l. Some fundamental contributions to this problem are due to H.G. Kellerer, E. Marczewski, V. Strassen et. al. (see H.G. Kellerer [41,42]). We are only interested in the modest special case of finite discrete spaces. In fact, the classical marginal problem has either no solution, one solution or an infinite number of solutions.

The probabilities $\mathbb{P}_{W_l}$ should also satisfy an additional consistency condition known as *pairwise compatibility* (see Malvestuto [56]). By pairwise compatibility one means that $C_{i,j} = W_i \bigcap W_j$, $i \neq j$ implies that the margin at $C_{i,j}$ satisfies

$$\mathbb{P}_{C_{i,j}} = \mathbb{P}_{W_i}^{\downarrow C_{i,j}} = \mathbb{P}_{W_j}^{\downarrow C_{i,j}}. \tag{3}$$

The collection of sets $\{W_l\}_{l=1}^{s}$ such that $\cup W_l = V$ is said to satisfy a *dependence structure* if for each l $W_l = A_l \cup B_l$ for two sets A_1, B_l such that $B_1 = \phi$,

$$B_j = W_j \bigcap \left(\bigcup_{k=1}^{j-1} W_k \right), \qquad j = 2, \ldots, s$$

and

$$A_j = W_j \backslash B_j \qquad j = 1, \ldots, s$$

Let $\mathcal{S} = (A_j, B_j)_{j=1}^{l}$. The probability distribution corresponding to the dependence structure $\tilde{\mathbb{P}}$, defined by

$$\tilde{\mathbb{P}}_{\mathcal{S}}(\underline{x}) = \mathbb{P}_{A_1}(\underline{x}_{A_1}) \prod_{j=2}^{s} \mathbb{P}_{A_j|B_j}(\underline{x}_{A_j}|\underline{x}_{B_j})$$

is called the *product approximation* of the distribution $\mathbb{P}$ with respect to $\mathcal{S}$.

It is shown by Malvestuto [56] that if the sets $\{W_l\}_{l=1}^{s}$ constitute a *dependence structure*, then pairwise compatibility implies *collective* compatibility (equation (2)) and there is a unique extension of the given $\mathbb{P}_{W_l}$ to a product approximation of $\mathbb{P}$. It can be seen that the well ordered nodes and their sets of parent nodes in a DAG form a dependence stucture. Hence the product of conditional probabilities $\prod_{j=1}^{d} \mathbb{P}_{X_{\sigma j}|X_{\Pi_j^{\sigma}}}$ is a unique and globally consistent probability distribution extending the given set of lower order probabilities. The first application of this way of looking at Bayesian networks is due to Chow and Liu [16], who approximated high dimensional distributions by products of second order probabilities on a binary tree.

1.5. Bayesian networks and causality. In several applications, the ordering of the variables is already given through principles from cause to effect; the variables arranged such that ancestor variables have causal influences on their descendant variables. In such problems, the factorisation along the DAG is a natural way of expressing the problem; the conditional probabilities of the factorisation are well defined natural building blocks. The leaf variables are the observable variables, the effects, while the variables with lower orders, the causes, are hidden. It is the conditional probability distribution of the hidden variables, given instantiations of the observed variables that are of interest, which are computed using Bayes rule. An early article that considers the notion of a factorisation of a probability distribution along a DAG representing causal dependencies is that by H. Kiiveri, T.P. Speed and J.B. Carlin [40], where a Markov property for Bayesian networks is defined. This is developed in 1990 by J. Pearl [66].

For example, hidden Markov models (HMMs) are used to model dynamic systems whose states are not observable, yet their outputs are. HMMs are widely used in applications requiring temporal pattern recognition including speech, handwriting and gesture recognition and various problems in bioinformatics, discussed by Durbin, Eddy and Mitchison [24].

1.6. Dynamic Bayesian Networks. This review largely confines itself to standard Bayesian networks, although *dynamic Bayesian networks* are an important tool that have proved useful for a large class of problems. The thesis of Kevin Murphy [60] provides a comprehensive introduction to the topic.

The first mention of dynamic Bayesian networks (DBNs) seems to be in 1989 by Dean and Kanazawa [22]. The DBN framework provides a way to extend Bayesian network machinery to model probability distributions over collections of random variables $(\underline{Z}_t)_{t\geqslant 0}$. The parameter $t \in \{0, 1, 2, \ldots\}$ represents time. Typically, the variables at a time slice t are partitioned into $\underline{Z}_t = (\underline{U}_t, \underline{X}_t, \underline{Y}_t)$ representing the input, hidden and output variables of the model. The term 'dynamic' refers to the fact that the system is dynamic; the basic structure remains the same over time.

Definition 1 A *k - slice Dynamic Bayesian network* is a DAG corresponding to a factorisation of the probability distribution over the variables $\{\underline{Z}_0, \underline{Z}_1, \ldots\}$ such that for $t \geqslant k$,

$$p_{Z_0,\ldots,Z_t} = p_{Z_0} \prod_{s=1}^{k-1} p_{Z_s|Z_0,\ldots,Z_{s-1}} \prod_{s=k}^{t} p_{Z_s|Z_{s-k},\ldots,Z_{s-1}}$$

where, for $t \geqslant k$,

$$P_{Z_t|Z_{t-k-1},\ldots,Z_{t-1}} = \prod_j p_{Z_t^j|\Pi(Z_t^j)},$$

Z_t^j is the jth node at time t, which could be a component of either X_t, Y_t or U_t and the set $\Pi(Z_t^j)$ of parents of Z_t^j belongs to the collection $\underline{Z}_{t-k}, \ldots, \underline{Z}_{t-1}$. There may also be associations within a time slice. These are indicated by undirected arrows, which do not represent causality. The resulting graph is a *chain graph.*

The requirement is that the subgraph restricted to $\{\underline{Z}_t, \ldots, \underline{Z}_{t+k-1}\}$ is the same for each $t \geqslant 0$ and the conditional probabilities $p_{Z_t^j|\Pi(Z_t^j)}$ are the same for each $t \geqslant k$. Furthermore, for $1 \leqslant i \leqslant j \leqslant k$, and each $s \geqslant j$, the subgraph restricted to $\{\underline{Z}_{s+i}, \ldots, \underline{Z}_{s+j}\}$ is a subgraph of the subgraph restricted to $\{\underline{Z}_{s+i-1}, \ldots, \underline{Z}_{s+j}\}$. □

The arcs between slices are from left to right and reflecting the causal flow of time. If there is an arc from Z_{t-1}^j to Z_t^j, the node Z^j is said to be *persistent*. The arcs *within* a slice may have arbitrary direction, so long as the overall DBN is a DAG. The arcs within a time slice may be undirected, since they model correlation or constraints rather than causation. The resulting model is then a (dynamic) chain graph.

The parameters of the conditional probabilities $p_{Z_t^j|\Pi(Z_t^j)}$ are time-invariant for $t \geqslant k$, i.e., the model is time-homogeneous. If parameters can change, they may be added to the state-space and treated as random variables or alternatively a hidden variable may be added that selects which set of parameters to use.

Within the engineering community, DBNs have become a popular tool, because they can express a large number of models and are often computationally tractable.

DBNs have been successfully applied to the reconstruction of genetic networks, where genes do not remain static, but rather their expression levels fluctuate constantly. Increased expression level of a gene will result in increased levels of mRNA

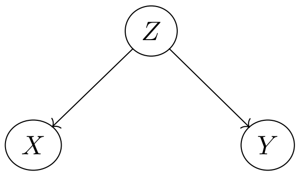

Figure 1: A fork connection

from that gene which will in turn influence the expression levels of other genes. DBNs have proved to be a successful way of analysing genetic expression data.

2. Directed Acyclic Graphs and Causal Networks.

2.1. Independence and *d*-separation. A *causal* network is one where the DAG along which the probability distribution factorises is considered to have a causal interpretation; the parents of a variable are those that have a direct causal effect on a variable. In such situations, it is natural for the conditional probabilities, where the conditioning variables are the direct causes, to be the basic building blocks and to use these to construct probability distributions over larger systems. In a DAG, there are three ways in which two variables with no direct connection between them can be connected via a third; the *fork, chain* and *collider* respectively, which have clear interpretations when the graph has been derived from causal principles.

1. The *fork* connection is illustrated in figure 1. The fork variable Z is a common cause.

   A probability distribution over the variables X, Y, Z that factorises according to the DAG in figure 1 may be expressed as

   $$p_{X,Y,Z} = p_Z p_{X|Z} p_{Y|Z}.$$

   Note that $X \perp Y|Z$; X and Y are conditionally independent given Z, but $X \not\perp Y$, or at least not necessarily. If a causal interpretation is valid, then the fork variable Z is a common cause. It is illustrated by the following example by Albert Engström (1869 - 1940), a Swedish cartoonist. 'During a convivial gathering there is talk of the unhygienic aspect of using galoshes. One of those present chimes in: "Yes, I've also noticed this. Every time I've woken up with my galoshes on, I've had a headache." ' There is an association between head and feet; they are not independent of each other. But the association may be explained fully through the state of the hidden variable Z, which denotes the activities of the previous evening, the common cause of both headache and forgetting to remove galoshes before sleeping.

2. The *chain* connection is illustrated in figure 2. This describes a situation where the association between X and Y is only through Z; X has a causal influence on Z, which in turn has a causal influence on Y.

   A probability distribution over (X, Y, Z) that factorises along the graph in figure 2 may be written as:

   $$p_{X,Y,Z} = p_X p_{Z|X} p_{Y|Z}.$$

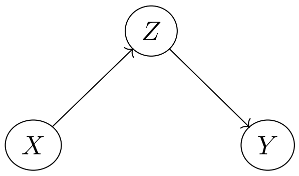

Figure 2: A chain connection

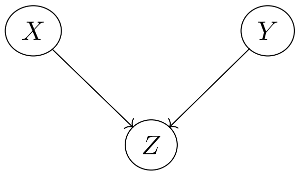

Figure 3: A collider connection

As with the fork connection, $X \perp Y|Z$, but $X \not\perp Y$, or at least not necessarily.

3. The *collider* connection is illustrated in figure 3. Here X and Y both have causal influence on Z. This corresponds to a factorisation

$$p_{X,Y,Z} = p_X p_Y p_{Z|X,Y}.$$

For variables that factorise according to a collider, the properties are the opposite; $X \perp Y$, but $X \not\perp Y|Z$; at least they are not necessarily conditionally independent. If there is information on the state of Z, then there will be a flow of information between X and Y. The classic example here is that of $X =$ 'burglary', $Y =$ 'earth tremor' and $Z =$ 'alarm'. An earth tremor and a real burglary can both set off the burglar alarm. If the burglar alarm is ringing, the information that there has been an earth tremor in the area will reduce one's fear that there may have been a real burglary. Information passes between X and Y only if there is information on Z.

All connections between two variables via a third in a DAG are of these three types: fork, chain or collider.

In a DAG, two nodes X and Y are said to be *d-separated* by a set S if all trails between X and Y, have either a fork or chain connection in S, or a collider connection not in S, with none if its descendants in S. The notation for this will be $X \perp Y\|_{\mathcal{G}}S$, where $\mathcal{G}$ denotes the graph in which the d-separation statement holds.

The main result used in Bayesian networks is that if X and Y are d-separated by a set S in the DAG, written $X \perp Y\|_{\mathcal{G}}S$, then the probabilistic statement $X \perp Y|S$ (X independent of Y given S) holds. There is a direct proof of this in Koski and Noble [48]. This does not necessarily work the other way round; if X and Y are d-connected by S (written $X \not\perp Y\|_{\mathcal{G}}S$), it does not necessarily hold that $X \not\perp Y|S$.

Definition 2 (Faithful) When d-separation statements for the DAG and independence statements for the probability distribution are equivalent, the DAG is said to be *faithful*. □

An ordering of the variables gives a factorisation and a corresponding DAG. An efficient factorisation is one where the d-separation statements of the DAG represents as much of the independence structure as possible. If there is a causal structure, then this provides a natural ordering of the variables, a factorisation and an efficient DAG with a causal interpretation.

For a node in a DAG, an important feature is its *Markov blanket*.

Definition 3 (Markov blanket) The Markov blanket of a node X in a DAG, denoted $MB(X)$, is the set of parents and children and parents of children of the variable. □

The important feature of the Markov blanket is that X is d-separated from the remaining variables in the network by its Markov blanket. This may be seen by considering the various connections to X; for each variable in $V\backslash(\{X\} \cup MB(X))$, any trail to X contains at least one fork or chain node in $MB(X)$. This is the smallest set that has this property.

2.2. Computing conditional probabilities. Once the network has been established, consisting of a DAG and specifications of the conditional probabilities corresponding to the factorisation, the network may be used for computation of arbitrary conditional probabilities. This proceeds according to, for example, the *Aalborg algorithm*, by Lauritzen and Spiegelhalter [50]. The Aalborg algorithm proceeds along the following lines:

1. The DAG is *moralised*. This means that, for each variable, undirected edges are added between each pair of common parents and then all the edges are undirected. The moral graph is the undirected graph where each variable / parent set in the original DAG is a clique.
2. The moral graph is then *triangulated* efficiently. Various triangulation algorithms are available; the choice depends on various factors.
3. An undirected graph is *decomposable* if and only if it is triangulated; the cliques of an undirected graph may be organised to form a *junction tree* if and only if the graph is decomposable. The cliques are then organised into a junction tree and the probability distribution factorised over the junction tree.

Definition 4 (Junction Tree) Let $\mathcal{C}$ denote the collection of cliques. A *junction tree* is a tree with node set $\mathcal{C}$ which satisfies the following property: if $X \in C_1$ and $X \in C_2$ for $C_1, C_2 \in \mathcal{C}$, then X is in every clique on the unique path between C_1 and C_2 in the tree. □

Let $\mathcal{C}$ denote the collection of cliques and $\mathcal{S}$ the collection of separators. That is, for two adjacent cliques C_i and C_j, their separator is $S_{ij} = C_i \cap C_j$, the variables present in both cliques. The factorisation of a probability distribution p over the junction tree is

$$p(\underline{x}) = \frac{\prod_{C \in \mathcal{C}} p_C(\underline{x}_C)}{\prod_{S \in \mathcal{S}} p_S(\underline{x}_S)},$$

where p_C denotes the marginal distribution over the clique C and p_S denotes the marginal over the clique S, $\underline{x}_S$ and $\underline{x}_S$ denoting the instantiations of C and S respectively.

The *evidence*, information that certain variables take specific values, is then inserted. Suppose the information is that variables in the set A are instantiated as $\underline{y}_A$. This is done by setting

$$\phi_C(\underline{x}_C) = \begin{cases} p_C(\underline{x}_C) & \underline{x}_{C\cap A} = \underline{y}_{C\cap A} \\ 0 & \text{otherwise.} \end{cases}$$

and the same for $\phi_S(\underline{x}_S)$. The information is then propagated by the following message passing algorithm: messages are passed from the leaf nodes to a designated root and then back out to the leaf nodes, updating the probability distribution over each clique.

The message passed from a clique C_1 to an adjacent clique C_2 with separator $S = C_1 \cap C_2$ is:

1. compute the *update ratio* $\lambda_S(\underline{x}_S) = \frac{\sum_{C_1\setminus S}\phi_{C_1}(\underline{x}_{C_1})}{\phi_S(\underline{x}_S)}$ (where $\sum_{C_1\setminus S}\phi_{C_1}(\underline{x}_{C_1})$ denotes: marginalise ϕ_{C_1} over those variables in C_1 that are not in S)

2. update $\phi_S(\underline{x}_S)$ to $\phi^*_S(\underline{x}_S) = \sum_{C_1\setminus S}\phi_{C_1}\phi_{C_1}(\underline{x}_{C_1})$.

3. update $\phi_{C_2}(\underline{x}_{C_2})$ to $\phi^*_{C_2}(\underline{x}_{C_2}) = \lambda_S(\underline{x}_S)\phi_{C_2}(\underline{x}_{C_2})$.

The message passing has to be carried out according to a *fully active schedule.* That is, messages are only passed from cliques that have received messages from all neighbours except possibly the one to which they are passing, the root has received messages from all its neighbours and then the schedule has been reversed so that all the leaves have received messages.

After a fully active schedule, Lauritzen and Spiegelhalter [50] proved that the marginalisation over each clique and separator gives the same value. Dividing through by this value gives the conditional probability distribution, conditioned on the evidence, hence giving the updated probabilities over each clique and separator. There are modifications of this algorithm to deal with *soft* or *virtual* evidence, evidence that alters the probability distribution over a variable, rather than instantiating it with a particular value. One such algorithm is the *big clique* algorithm, which involves putting all the variables on which soft evidence is received into one big clique, which is the root and using iterative proportional fitting to find a distribution that satisfies the soft evidence requirement closest to the original distribution in terms of Kullback Leibler divergence.

The Aalborg method is useful for large numbers of variables, where each clique on the junction tree is relatively small. It can deal with discrete variables, while Lauritzen [51] extends its use to conditional Gaussian distributions.

Other algorithms are available for updating. For example, the *sum product* algorithm, following Wiberg [92] and Markov chain Monte Carlo, following Pearl [65] are implemented in Kevin Murphy's MATLAB toolbox for Bayesian networks. Shafer [83] contains a concise summary of message passing and introduces the arrow notations for marginals. Hajek et. al. [36] contains much information about various schemes of

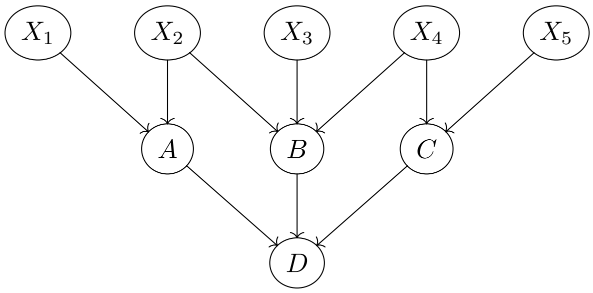

Figure 4: Epidemiology: A, B, C are minimal sufficient causes

message passing and describes the Czech contributions in the 1970's to probabilistic modelling in expert systems.

2.3. Context Specific Independence and Minimal Sufficient Causation. There are modifications available to incorporate more of the independence structure; if X, Y, Z, C are pairwise disjoint sets of variables, then X and Y are said to be *contextually independent* given Z and the context c, a particular instantiation of C, if

$$\mathbb{P}(X = x|Y = y, Z = z, C = c) = \mathbb{P}(X = x|Z = z, C = c)$$
$$\forall x, y, z \quad \text{such that} \quad \mathbb{P}(Y = y, Z = z, C = c) > 0.$$

Clearly, if the aim is to compute conditional probabilities of events given the instantiation $C = c$, the contextual independence can be incorporated into the network to reduce the number of edges that have to be considered in the graph and hence reduce the sizes of the cliques. This modification can be incorporated into the Aalborg algorithm quite easily, increasing its efficiency. Context-specific independence is discussed by Boutilier et. al. [3]. Recently, in Corander, Koski, Nyman, Pensar [20], a structure learning algorithm is given using an extension of context-specific independence.

Context specific independence arises in networks used for epidemiological modelling. Suppose that $X_1, \ldots, X_n$ denote various factors that can cause a condition, $X_i = 1$ if the cause is present and 0 if the cause is absent. Let $\overline{X_i} = 1 - X_i$. Suppose that certain combinations of the presence / absence of various factors cause the condition. One example from Vanderweele and Robins [91] is where there are 5 factors and the condition is caused if $X_1X_2 = 1$ or $\overline{X_2}X_3X_4 = 1$ or $X_4\overline{X}_5 = 1$. Let D denote the condition; $D = 1$ if present, $D = 0$ if absent. By adding in the variables $A = X_1X_2$ and $B = \overline{X_2}X_3X_4$ and $C = X_4\overline{X_5}$, the situation may be represented by the 'or' gate in figure 4. If $X_1, \ldots, X_5$ are the only causing factors and if they only act through either A, B or C, then A, B, C are *minimal sufficient causes.*

If $D = 0$, then $A = 0$ and $B = 0$ and $C = 0$; $A \perp B|\{D = 0\}$, $A \perp C|\{D = 0\}$ and $B \perp C|\{D = 0\}$, while (for example) $A \not\perp B|\{D = 1\}$.

3. The pioneering work of Arthur Cayley [8]. Arthur Cayley F.R.S. (16 August 1821 - 26 January 1895) was a British mathematician, known for his work in pure mathematics. His contributions include the so-called Cayley-Hamilton theorem, that every square matrix satisfies its own characteristic polynomial, which he verified in 1858 for matrices of order 2 and 3 (see [10]). He was the first to define the concept of a group in the modern way, as a set with a binary operation satisfying certain laws. From group theory, he is known for *Cayley's theorem* published in 1854, which states that every group G is isomorphic to a subgroup of the symmetric group acting on G (see [9]).

We do not discuss these aspects of his work; our attention is drawn to a short article by Arthur Cayley from 1853, where in an example that takes less than one page, he seems to develop several principles that later formed the basis of the subject of Bayesian networks. Perhaps it would not be going too far to state that each piece of work listed in the bibliography of over ninety pieces at the end of this article (excluding the works of Cayley) represents a corollary of the principles laid down by Cayley. Yet at the time of writing, this article has, to our knowledge, been quoted four times since its publication; this occasion may be the fifth.

We reproduce the article in its entirety.

XXXVII. *Note on a Question in the Theory of Probabilities.*

By A. Cayley.*

The following question was suggested to me, either by some of Prof. Boole's memoirs on the subject of probabilities, or in conversation with him, I forget which; it seems to me a good instance of the class of questions to which it belongs.

Given the probability α that a cause A will act, and the probability p that A acting the effect will happen; also the probability β that a cause B will act, and the probability q that B acting the effect will happen; required the total probability of the effect.

As an instance of the precise case contemplated, take the following: say a day is called *windy* if there is at least w of wind, and a day is called *rainy* if there is at least r of rain, and a day is called *stormy* if there is at least W of wind, *or* if there is at least R of rain. The day may therefore be stormy because of there being at least W of wind, or because of there being at least R of rain, or on both accounts; but if there is less than W of wind *and* less than R of rain, the day will not be stormy. Then α is the probability that a day chosen at random will be windy, p the probability that a windy day chosen at random will be stormy, β the probability that a day chosen at random will be rainy, q the probability that a rainy day chosen at random will be stormy. The quantities λ, μ introduced in the solution of the question mean in this particular instance, λ the probability that a windy day chosen at random will be stormy by reason of the quantity of wind, or in other words, that there will be at least W of wind, μ the probability that a rainy day chosen at random will be stormy by reason of the quantity of rain, or in other words, that there will be at least R of rain.

The sense of the terms being clearly understood, the problem presents of course no difficulty. Let λ be the probability that the cause A acting will act efficaciously; μ the probability that the cause B acting will act efficaciously; then

$$p = \lambda + (1-\lambda)\mu\beta$$
$$q = \mu + (1-\mu)\alpha\lambda,$$

which determine λ, μ; and the total probability ρ of the effect is given by

$$\rho = \lambda\alpha + \mu\beta - \lambda\mu\alpha\beta,$$

suppose, for instance, $\alpha = 1$, then

$$p = \lambda + (1-\lambda)\mu\beta, \quad q = \mu + \lambda - \lambda\mu, \quad \rho = \lambda + \mu\beta - \lambda\mu\beta,$$

that is, $\rho = p$, for p is in this case the probability that (acting as a cause which is certain to act) the effect will happen, or what is the same thing, p is the probability that the effect will happen.
Machynlleth, August 16, 1853.
*Communicated by the Author.

In this short note, Cayley gives a prototype example of a causal network; rain and wind both have causal effects on the state of the day (stormy or not), which may be *inhibited*. He demonstrates the key principle of *modularity*, taking a problem with several variables and splitting it into its simpler component conditional probabilities, by considering the direct causal influences for each variable and considering the natural factorisation of the probability distribution in this problem into these conditional probabilities.

It should also be pointed out that Cayley was no stranger to graph theory; in 1889 he proved *Cayley's tree formula*, that there are n^{n-2} distinct labelled trees of order n (see [13]) and established links between graph theory and group theory, representing groups by graphs. The *Cayley graph* is named after him.

The variables here may be taken as

$$C = \begin{cases} 1 & \text{wind} \\ 0 & \text{no wind} \end{cases} \qquad D = \begin{cases} 1 & \text{rain} \\ 0 & \text{no rain} \end{cases}$$

with

$$\alpha = p_C(1) \qquad \beta = p_D(1).$$

Let Y be the variable denoting whether there is a storm;

$$Y = \begin{cases} 1 & \text{storm} \\ 0 & \text{no storm} \end{cases}$$

Then, in Cayley's notation, if there is rain, it causes a storm with probability μ; if there is wind, it causes a storm with probability λ. The corresponding 'network', on three variables, is seen in figure 5. The subscripts μ and λ on the arrows indicate the probability that the cause, if active, will trigger the effect.

This is a *noisy 'or' gate*, which can be expressed as a logical 'or' gate by the addition of two variables, R and W. The variable R denotes *severe rain*, that is that the 'rain' variable reaches the threshold to trigger a storm. This happens if the quantity of rain is above a threshold. The W variable denotes *severe wind*; that is, that the 'wind' variable reaches the threshold to trigger a storm. This happens if the

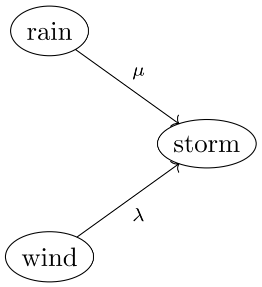

Figure 5: Rain and wind causing a storm

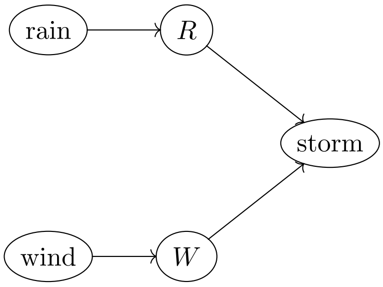

Figure 6: Rain and wind: logical 'or' gate

strength of wind is above a threshold. The variables, to form the logical or gate have conditional probability values given below; $p_{W|C}$ denotes the conditional probability function for the variable W given C and $p_{R|D}$ denotes the conditional probability function for the variable R given D.

$$p_{W|C} = \begin{array}{c|cc} C\backslash W & 1 & 0 \\ \hline 1 & \lambda & 1-\lambda \\ 0 & 0 & 1 \end{array} \qquad p_{R|D} = \begin{array}{c|cc} D\backslash R & 1 & 0 \\ \hline 1 & \mu & 1-\mu \\ 0 & 0 & 1 \end{array}$$

The network may now be expressed graphically according to figure 6. This DAG is a representation of the factorisation that Cayley is using;

$$p_{C,D,R,W,Y} = p_C p_D p_{R|C} p_{W|D} p_{Y|W,R}$$

where $p_{Y|W,R}$ denotes the conditional probability function for the variable Y, given W and R. For $Y = 1$, these values are given in the following table:

$$p_{Y|W,R}(1|.,.) = \begin{array}{c|cc} W\backslash R & 1 & 0 \\ \hline 1 & 1 & 1 \\ 0 & 1 & 0 \end{array}.$$

From the factorisation,

$$p_W(1) = \sum_x p_{W|C}(1|x)p_C(x) = \lambda\alpha, \quad p_R(1) = \mu\beta,$$

From Cayley, p is the probability that a windy day, chosen at random, will be stormy; $p = p_{Y|D}(1|1)$.

$$\begin{aligned} p &= p_{Y|D}(1|1) = \sum_{x_1} p_A(x_1) \sum_{x_2} p_{R|C}(x_2|x_1) \sum_{x_3} p_{Y|R,W}(1|x_2,x_3) p_{W|D}(x_3|1) \\ &= \beta\lambda\mu + \beta\mu(1-\lambda) + \beta(1-\mu)\lambda + (1-\beta)\lambda \\ &= \beta\mu - \beta\lambda\mu + \lambda = \lambda + (1-\lambda)\beta\mu. \end{aligned}$$

Similarly, q, the probability that a rainy day, chosen at random, will be stormy; $q = p_{Y|C}(1|1)$, is given by

$$q = \mu + (1-\mu)\alpha\lambda,$$

as computed by Cayley. Cayley is deriving the expression for the marginal probability of a stormy day, $\rho = p_Y(1)$;

$$\begin{aligned} p_Y(1) &= \sum_{x_1} p_C(x_1) \sum_{x_2} p_D(x_2) \sum_{x_3} p_{R|C}(x_3|x_1) \sum_{x_4} p_{W|D}(x_4|x_2) p_{Y|R,W}(1|x_3,x_4) \\ &= \sum_{x_3} p_R(x_3) \sum_{x_4} p_W(x_4) p_{Y|R,W}(1|x_3,x_4) \\ &= p_R(1)p_W(1) + p_R(1)p_W(0) + p_R(0)p_W(1) \\ &= \alpha\lambda + \beta\mu - \alpha\beta\lambda\mu. \end{aligned}$$

This simple construction from 1853 represents, to our knowledge, the first example of a causal network and the first construction of a noisy-or gate, with the concept of an inhibitor.

4. Arthur Cayley and Judea Pearl's intervention calculus. The rule of Bayes, for computing a conditional probability is

$$\mathbb{P}(A|B) = \frac{\mathbb{P}(A)\mathbb{P}(B|A)}{\mathbb{P}(B)}. \tag{4}$$

Starting with a well defined probability space $(\Omega, \mathcal{A}, \mathbb{P})$, the probability of an event $A \in \mathcal{A}$ has value $\mathbb{P}(A)$. An event $B \in \mathcal{A}$ is then *observed* and, using Bayes rule, one may compute the probability of the event A given the information that event B has been observed. The starting point is a probability space $(\Omega, \mathcal{A}, \mathbb{P})$, where $\mathcal{A}$, an algebra or σ-algebra, is the space of events over which $\mathbb{P}$ has been defined.

Bayes rule, equation (4) is no longer applicable if $B \notin \mathcal{A}$. The classic example is 'sprinkler and wet grass'. If one *observes* that the sprinkler is on, one can infer that the season has been dry. If one knows that the state 'sprinkler on' has been forced, for reasons independent of the season or the amount of use (for example, if it is 'on' as a result of a regular maintenance procedure), then clearly no such inferences can be made; the additional information that regular maintenance work is taking place represents information above and beyond the simple observation that the sprinkler is on. The conditioning is no longer simply on event B.

Judea Pearl [73] proposed the following framework for computing conditional probabilities, when the conditioning is forced independently of other considerations in the network rather than observed. Pearl's framework requires two main assumptions:

1. There is a *causal* structure between the variables, which may be represented by a DAG, where the parent set represents the direct causes.

2. The intervention is made irrespective of the state of the system.

Pearl's framework is of particular value for describing a *controlled experiment* (as pointed out by D. Lindley [53]). Intervening to force the state of a variable, independently of other factors, amounts to a controlled experiment; individuals are randomly assigned to control group and treatment groups, irrespective of other considerations. Links to other factors that may have been common causes for both the 'treatment' variable and chances of recovery have been removed.

Pearl's method can be summarised as follows: if an intervention is carried out whereby a variable X is forced to take a value x (written $X \leftarrow x$), irrespective of the state of the other variables, then the variable X is removed from the graph. That is, all edges to and from X are removed and the variable X is removed. For a variable Y, if $X \in \Pi_Y$ (the parent set of Y) in the original graph, the conditional probability $p_{Y|\Pi_Y}$ with the instantiation $X = x$ in the parent set Π_Y is used.

In terms of the factorisation, suppose that

$$p_{X_1,\dots,X_d} = \prod_{j=1}^{d} p_{X_j|\Pi_j}$$

is obtained from causal principles, where variables of lower order have causal effect on variables of higher order. Then the probability after an intervention $X_i \leftarrow x_i$ is defined as

$$p_{X_1,\dots X_d \| X_i} = \prod_{j \neq i} p_{X_j|\Pi_j}$$

where the instantiation x_i is used for X_i when X_i appears in Π_j. This is known in the literature as 'do'-conditioning. There is the cryptic remark towards the end of Arthur Cayley's paper, which indicates that he may already had this framework in mind when considering causal probabilistic models. The phrase ' .... acting a cause which is certain to act' may be a clumsy way of expressing a brilliant insight into the intervention calculus, if by 'acting' he means intervening to force the state of the variable.

This reading may be somewhat strained; in Arthur Cayley's example, no human intervention is possible to force the states of the wind or rain variables. Since 'wind' and 'rain' are both ancestor variables, no links are removed from the DAG and in Pearl's framework, intervention conditioning is the same as the standard conditioning on an observation. The wording suggests, though, that he understood, from causal principles, that the two equations relating λ and μ to p and q remain valid if the conditioning on an ancestor variable is forced by intervention, rather than simply observed, one of the features of Pearl's intervention calculus.

5. Arthur Cayley: the connection between algebraic geometry and Bayesian networks. The emerging field of algebraic statistics (Pistone et al. [74], Drton et. al. [23]) advocates polynomial algebra as a tool in the statistical analysis of experiments and discrete data; the connection between algebraic geometry and Bayesian networks is discussed by Garcia et. al. [33].

For a probability distribution over a set of variables, the conditional independence statements for subsets X, Y, Z, W satisfy the following logical relations, discussed by Pearl [72]:

1. **symmetry** $X \perp Y|Z \Leftrightarrow Y \perp X|Z$
2. **decomposition** If $X \perp Y \cup W|Z$ then $X \perp Y|Z$ and $X \perp W|Z$.
3. **contraction** If $X \perp Y|Z$ and $X \perp W|Y \cup Z$ then $X \perp W \cup Y|Z$.
4. **weak union** If $X \perp Y \cup Z|W$ then $X \perp Y|Z \cup W$.

A collection of sets $\mathcal{V}$ such that these statements are satisfied for any $X, YW, Z \in \mathcal{V}$ has come to be known as a *semi-graphoid.* Any independence model is a semi-graphoid. A *graphoid* is a semi-graphoid that also satisfies the converse of weak union, known as the *intersection* property.

5 **intersection** If $X \perp Y|W \cup Z$ and $X \perp W|Y \cup Z$ then $X \perp W \cup Y|Z$.

Furthermore, a graphoid or semi-graphoid for which the reverse implication of the decomposition property holds is said to be *compositional*, that is

6 **composition**

$$\text{If} \quad X \perp Y|Z \quad \text{and} \quad X \perp W|Z \quad \text{then} \quad X \perp Y \cup W|Z. \tag{5}$$

The axioms for a compositional graphoid are discussed recently in Sadeghi and Lauritzen [80].

d-separation statements for sets of variables in a DAG also satisfy these relations (replace $|$, denoting conditioning, with $\|_{\mathcal{G}}$, denoting d-separation in the DAG $\mathcal{G}$). The d-separation statements of a DAG also satisfy other relations. Let $\mathcal{G} = (V, D)$ denote a DAG, $X \subseteq V$, $Y \subseteq V$ and $Z \subseteq V$ sets of nodes, $\alpha, \beta, \gamma, \delta \in V \backslash X \cup Y \cup Z$ denote individual nodes. Then the following additional d-separation properties hold:

1. if $X \perp Y\|_{\mathcal{G}}Z$ and $X \perp Y\|_{\mathcal{G}}Z \cup \{\gamma\}$ then either $X \perp \{\gamma\}\|_{\mathcal{G}}Z$ or $Y \perp \{\gamma\}\|_{\mathcal{G}}Z$
2. if $\alpha \perp \beta\|_{\mathcal{G}}\{\gamma, \delta\}$ and $\gamma \perp \delta\|_{\mathcal{G}}\{\alpha, \beta\}$ then either $\alpha \perp \beta\|_{\mathcal{G}}\{\gamma\}$ or $\alpha \perp \beta\|_{\mathcal{G}}\{\delta\}$.

These last two implications may be found in Bromberg and Margaritis in [4]; they hold as d-separation statements in a DAG, but do not necessarily hold for conditional independence. In fact, it is not possible to axiomatise conditional independence, as Studenỳ proved. The proof may be found in [88].

A *factorisation* is equivalent to a set of conditional independence statements;

$$\{X_{\sigma(j)} \perp \Xi_j^{\sigma}|\Pi_j^{\sigma} \quad j = 1, \ldots, d\},$$

where $\Pi_j^{\sigma} \subset \{X_{\sigma(1)}, \ldots, X_{\sigma(j-1)}\}$ is the parent set of variable $X_{\sigma(j)}$ when ordering σ is employed and $\Xi_j^{\sigma} = \{X_{\sigma(1)}, \ldots, X_{\sigma(j-1)}\} \backslash \Pi_j^{\sigma}$.

Let $V = \{X_1, \ldots, X_d\}$ denote the variable set, let $\underline{X} = (X_1, \ldots, X_d)$ the random vector, let the state space for variable X_j be $\mathcal{X}_j = \{x_j^{(1)}, \ldots, x_j^{(k_j)}\}$ and the state space for $\underline{X}$ be $\mathcal{X} = \times_{j=1}^{d}\mathcal{X}_j$. Let $\mathcal{Y} = \times_{j=1}^{d}(1, \ldots, k_j)$ and $\mathbb{R}(\mathcal{Y})$ the ring of polynomial functions on $\mathbb{R}^{\mathcal{Y}}$.

A conditional independence statement $A \perp B|C$, where A, B and C are disjoint subsets of V, translates using proposition 8.1 from Sturmfels [89], into a set of homogeneous quadratic polynomials on $\mathbb{R}(\mathcal{Y})$, and these polynomials generate an *ideal*. Let $\mathcal{I}_{A\perp B|C}$ denote the ideal generated by the statement $A \perp B|C$. The ideal for a collection of independence statements, for example those corresponding to a factorisation, is defined as the sum of the ideals; let $\mathcal{M} = \{A_i \perp B_i|C_i \quad i = 1, \ldots, m\}$, then

$$\mathcal{I}_{\mathcal{M}} = \mathcal{I}_{A_1\perp B_1|C_1} + \ldots + \mathcal{I}_{A_m\perp B_m|C_m}.$$

Cayley is using the expression of the conditional independence statements that define the factorisation in terms of polynomials to obtain the two polynomial equations

$$\begin{cases} p = \lambda + (1-\lambda)\mu\beta \\ q = \mu + (1-\mu)\lambda\alpha \end{cases} \tag{6}$$

and writes, '.... which determine λ and μ'. This amounts to finding roots of the two polynomials in λ, μ

$$\begin{cases} f_1(\lambda,\mu) = \lambda + (1-\lambda)\mu\beta - p \\ f_2(\lambda,\mu) = \mu + (1-\mu)\lambda\alpha - q \end{cases}$$

In terms of algebraic geometry, equation (6) defines the *affine variety*

$$V(f_1, f_2) = \left\{(\lambda,\mu) \in \mathbb{R}^2 | f_1(\lambda,\mu) = f_2(\lambda,\mu) = 0\right\}.$$

In his brief note, Cayley has pointed out the connections between Bayesian networks and algebraic geometry, a subject that he knew well. Cayley did much to clarify a large number of interrelated theorems in algebraic geometry and is known for the Cayley surface introduced in 1869 (see [11]).

6. Learning.

Notations. $V = \{X_1, \ldots, X_d\}$ denotes the variable set, which may be organised as a random vector $\underline{X} = (X_1, \ldots, X_d)$, taken as a row vector. The state space of variable X_j is $\mathcal{X}_j = (x_j^{(1)}, \ldots, x_j^{(k_j)})$ and the state space of $\underline{X}$ is $\mathcal{X} = \times_{j=1}^{d}\mathcal{X}_j$. In a DAG, Π_j denotes the parent set of variable X_j, with state space $\mathcal{X}_{\Pi_j} = (\pi_j^{(1)}, \ldots, \pi_j^{(q_j)})$. The conditional probability values are denoted by

$$\theta_{jil} = \mathbb{P}\left(X_j = x_j^{(i)} | \Pi_j = \pi_j^{(l)}\right)$$

for an $n \times d$ data matrix $\mathbf{x}$, which contains n instantiations of $\underline{X}$, $n(x_j^{(i)}, \pi_j^{(l)})$ denotes the number of times that the variable / parent configuration $(x_j^{(i)}, \pi_j^{(l)})$ appears, $n(\pi_j^{(l)})$ denotes the number of times the $\pi_j^{(l)}$ instantiation of the parent set Π_j appears. The entire collection of parameters is denoted $\underline{\theta}$.

6.1. Learning the parameters, graph structure given. We only mention in passing the second of the tasks listed, learning the parameters for a given structure. If the data matrix $\mathbf{x}$ contains n *complete* instantiations, estimates $\widehat{\theta}_{jil}$ of θ_{jil} may be obtained in the obvious way; $\widehat{\theta}_{jil} = \frac{n(x_j^{(i)}, \pi_j^{(l)})}{n(\pi_j^{(l)})}$, the probability being estimated by the observed proportion.

The situation lends itself, though, to Bayesian inference, exploiting the fact that the Dirichlet distribution is a conjugate family. A prior distribution $\Theta = \prod_{j,l} \Theta_{j,l}$ is placed over the parameter space, where $\Theta_{j,l}$ is the distribution over $(\theta_{j1l}, \ldots, \theta_{jk_j,l})$, given by $\text{Dir}(\alpha_{j1l}, \ldots, \alpha_{jk_jl})$. The parameters $\underline{\alpha}$ are chosen to represent prior assessment. After observing $\mathbf{x}$, an $n \times d$ data matrix with n complete instantiations, Θ updates, using Bayes rule, to $\Theta_{\mathbf{x}} = \prod_{j,l} \Theta_{j,l;\mathbf{x}}$, where

$$\Theta_{j,l;\mathbf{x}} = \text{Dir}(\alpha_{j1l} + n(x_j^{(1)}, \pi_j^{(l)}), \ldots, \alpha_{jk_jl} + n(x_j^{(k_j)}, \pi_j^{(l)})).$$

Information from incomplete instantiations is incorporated by *approximate* updating techniques, to retain the convenient product form and to remain within the conjugate family of Dirichlet distributions. Fading may also be accommodated within this framework, to accommodate dynamic situations where the parameter values change and older information has less value. Parameter learning for a given DAG may be found in Heckerman, Geiger and Chickering [37], approximate updating when learning from incomplete data in Ramoni and Sebastiani [76]. Another treatment of learning is found in Neapolitan [61].

6.2. Structure learning. A structure may be determined by causal reasoning, or formally constructed by engineering considerations. This is not considered here; the problem considered is the task of locating a structure purely from data. That is, learning a structure that encodes the key features of the dependence structure between the d variables of the data set, when presented with n complete or incomplete instantiations. This is the task of *structure learning.*

Markov equivalence. The following definitions, which are standard, are found in Koski and Noble [48]. Two DAGs that have the same d-separation properties are said to be *Markov equivalent.* If two DAGs are Markov equivalent, then a probability distribution that factorises along one of the DAGs also factorises along the other.

In a DAG, a *vee structure* is a configuration of three nodes, α, β and γ, where $\alpha \sim \beta$ and $\beta \sim \gamma$, but $\alpha \not\sim \gamma$ (the symbol $\sim$ denotes that there is an edge between the two nodes, but does not indicate its direction). An *immorality* is a vee structure where $\alpha \to \beta \leftarrow \gamma$. This is the 'burglary' example; information can pass from α to γ through β only if information is received on β. The term 'immorality' derives from the fact that two parents of a node are not linked.

An important result is that two DAGs are Markov equivalent if and only if they have the same *skeleton* (that is the undirected versions of the graphs are the same) and the same immoralities.

The *essential graph* of a Markov equivalence class is a graph that has both directed and undirected edges (known as a p-DAG), where the directed edges are those that retain the same direction for all DAGs within the Markov equivalence class, the other edges being undirected.

Methods for structure learning. Based only on data, structure learning techniques will locate a suitable equivalence class, either by finding a DAG within the equivalence class or by finding the essential graph. Structure learning methods tend to fall generally into three categories: search and score techniques, where a score function is used and the algorithm attempts to find the structure that maximises the score function, *constraint based* methods, where conditional independence tests are carried out and independence relations thus established provide constraints that the d-separation statements of the output graph should satisfy and *hybrid* algorithms that use both constraint based and search and score techniques.

6.3. Search and score. Search and score algorithms operate by selecting various structures for examination and scoring them. The structure with the highest score from among those considered is selected. Several score functions have been considered in the literature. They all have the feature of giving higher scores to those where the best fitting distribution, given the graph structure, is closest to the empirical distribution, with a penalty for the number of parameters. The *likelihood function* for a graph structure D, given an $n \times d$ data matrix $\mathbf{x}$, where the rows, $(x_{i1}, \dots, x_{id})$ for $i = 1, \dots, n$ represent n instantiations of the random vector $\underline{X} = (X_1, \dots, X_d)$, is given by the formula

$$L(D; \mathbf{x}) = \prod_{j=1}^{d} \prod_{l=1}^{q_j} \frac{\Gamma(\sum_{i=1}^{k_j} \alpha_{jil})}{\Gamma\left(n(\pi_j^l) + \sum_{i=1}^{k_j} \alpha_{jil}\right)} \prod_{i=1}^{k_j} \frac{\Gamma(n(x_j^i, \pi_j^l) + \alpha_{jil})}{\Gamma(\alpha_{jil})}, \tag{7}$$

where α_{jil} represents *virtual* information, the weight given to the (x_j^i, π_j^l) configuration before the data is received. Formula (7) is known as the *Cooper Herzkovitz likelihood* and was first derived in [17].

Using this as the basis of a score function has some practical difficulties; a large number of hyper-parameters $\underline{\alpha}$ have to be specified. The *log likelihood* is more practical

$$LL(D; \mathbf{x}) = \sum_{j=1}^{d} \sum_{i=1}^{k_j} \sum_{l=1}^{q_j} n(\pi_j^{(l)}, x_j^{(i)}) \ln \frac{n(\pi_j^{(l)}, x_j^{(i)})}{n(\pi_j^{(l)})}. \tag{8}$$

This cannot be used directly as a score function; it will favour graphs with many edges. A penalisation is usually introduced to favour graphs with fewer parameters. The most common score function considered is the *Bayesian Information Criterion*:

$$\mathrm{BIC}(D, \mathbf{x}) = LL(D; \mathbf{x}) - \frac{1}{2}(\ln n)|\underline{\theta}| \tag{9}$$

where D represents the DAG along which the factorisation is made and $|\underline{\theta}| = \sum_{j=1}^{d} q_j(k_j - 1)$ is the number of independent parameters required (recall that $\sum_{i=1}^{k_j} \theta_{jil} = 1$ for each (j, l)). The BIC was developed in 1978 by Gideon E. Schwartz [81], who described Bayesian principles for using it. The negative of this score function is known as the *minimum description length* (MDL); $MDL = -BIC$. The MDL was introduced in 1978 by Jorma Rissanen [77] independently and simultaneously.

Other score functions may be used, such as the *Akaike Information Criterion* (AIC), which is similar to the BIC, except that for the penalisation, $\frac{1}{2} \ln n$ is replaced by n. The AIC was introduced in 1974 by Hirotugu Akaike [1].

The simplicity of the function in equation (8) as the basis of a score function is appealing, but the hyper-parameters of the Cooper Herzkovitz likelihood give the possibility to input prior information. The parameters $\underline{\alpha}$ represent the prior assessment of the probability values for a given network, but prior information can also be placed over the graph structures. If a prior distribution $p_{\mathcal{D}}$ is placed over the space of DAGs with d nodes, denoted $\mathcal{D}$, then the posterior distribution, given data $\mathbf{x}$, may be written as

$$p_{\mathcal{D}|\mathbf{X}}(D|\mathbf{x}) = \frac{L(D;\mathbf{x})p_{\mathcal{D}}(D)}{p_{\mathbf{X}}(\mathbf{x})} \propto L(D;\mathbf{x})p_{\mathcal{D}}(D),$$

where $L(D;\mathbf{x})$ is the Cooper Herskovitz likelihood of equation (7). The subject of constructing appropriate prior distributions is discussed for example by Castelo and Siebes [7] and by Flores, Nicholson, Brunskill, Korb and Mascaro [27]. This may be used to construct a score function based on equation (9), where the log posterior is used in place of $LL(D;\mathbf{x})$.

The approach of testing each possible structure is not computationally feasible. This is because the number of possible DAGs grows super exponentially in the number of nodes. In [79], Robinson gave the following recursive function for computing the number $N(d)$ of DAGs with d nodes:

$$N(d) = \sum_{i=1}^{d}(-1)^{i+1}\begin{pmatrix} d \\ i \end{pmatrix} 2^{i(d-1)}N(d-i). \tag{10}$$

For $d = 5$ it is 29000 and for $d = 10$ it is approximately 4.2×10^{18}. Here $N(d)$ is a very large number, even for small values of d. It is not feasible to test all possible graphs, even for modest values of d.

Search and score methods usually deal with this by running a Markov process through the search space. One example is the *Markov chain Monte Carlo model composition* algorithm introduced by Madigan and York [55] in 1995 and developed by Madigan, Andersson, Perlman and Volinsky [54] in 1997. These algorithms choose, as search space, the space of essential graphs.

An efficient Markov chain Monte Carlo algorithm for searching the model space was developed by Corander, Ekdahl and Koski [18] in 2008, for families of models for which the marginal likelihood can be calculated analytically, either exactly or approximately, given any fixed structure. The model space is explored by a finite number of interacting parallel stochastic processes.

The use of 'Markov chain Monte Carlo' with reference to structure learning is misleading. The term McMC has a well defined standard mathematical meaning, which is to build up an empirical distribution that approximates the stationary distribution of the Markov chain. In structure learning for Bayesian networks, the stationary distribution of the Markov chain is irrelevant. The aim of producing a well constructed process is to visit the most promising candidate graphs within the search space. The search space is so large that very few graphs will be visited more than once and the overwhelming majority will not be visited at all. There is insufficient information to build up an empirical distribution over graph space. The decision is made purely by computing the score functions and choosing the graph visited which gives the largest score. The only consideration in constructing a process is to ensure that the process will visit structures that give a good representation of the data; properties of the search process, such as the Markov property, reversibility, convergence of the empirical distribution to a certain stationary distribution are all irrelevant.

Sparse Candidate Algorithm. The sparse candidate algorithm has been developed by Friedman, Nachman and Pe'er [29] in 1999. The main idea of the technique is to identify a relatively small number of *candidate* parents for each variable. This is based on simple local statistics, such as correlation. Attention is then restricted to networks in which the parent set is a subset of the candidate parent set.

The algorithm proceeds as follows: let D_n denote the DAG chosen at iteration n, let $\Pi_i^{(n)}$ denote the parent set for variable X_i in D_n.

- For $i = 1, \ldots, d$, choose the candidate set $C_i^{(n)} = \{Y_1, \ldots, Y_k\}$ of candidate variables for Π_i, the parent set for variable X_i. The set $C_i^{(n)}$ is chosen as $\Pi_i^{(n-1)}$ together with children and parents of children of X_i in D_n, and all those variables $Y \notin MB(X_i)$ such that the score

$$\sum_{(x,y,z)\in\mathcal{X}_{X_i}\times\mathcal{X}_Y\times\mathcal{X}_{\Pi_i^{(n-1)}}} n_{X_i,Y,\Pi_i^{(n-1)}}(x,y,z) \ln \frac{n_{X_i,Y,\Pi_i^{(n-1)}}(x,y,z)n_{\Pi_i^{(n-1)}}(z)}{n_{Y,\Pi_i^{(n-1)}}(y,z)n_{X_i,\Pi_i^{(n-1)}}(x,z)}$$

  is sufficiently high. Here MB denotes *Markov blanket* (definition 3). Also, for a set W, $n_W(w)$ denotes the number of appearances of configuration w in the data matrix $\mathbf{x}$. If the test statistic is low, it supports $X_i \perp Y|\Pi_i^{(n-1)}$ and hence Y is not a candidate parent.

  There are other ways of determining the candidate parents; anything in the current Markov blanket not d-separated from the variable by the Markov blanket should be included as a candidate parent.

- Find a high scoring network D_n where $\Pi_i^{D_n} \subset C_i^{(n)}$ for $i = 1, \ldots, d$.

This is the method successfully used for analysis genetic expression data by Friedman et. al. [30].

Optimal Reinsertion. The *optimal reinsertion* algorithm, introduced by Moore and Wong [59] in 2003, is a search-and-score algorithm that works along the following lines: at each step a *target node* is chosen, all edges entering or leaving the target are deleted, and the optimal combination of in-edges and out-edges is found, the node is re-inserted with these edges. This involves searching through the legal candidate parent sets and, for each candidate parent set, the legal child sets. The optimal reinsertion may be combined with sparse candidate.

Greedy Search and Greedy Equivalence Search. The *Greedy Search*, introduced by Chickering [15] in 2002, works along the following lines to produce a DAG, along which the probability distribution factorises, starting from the graph with no edges:

- *Forward phase* Let E_0 denote the graph with no edges. Let E_n denote the essential graph from stage n of the forward phase. Consider all possible DAGs within the Markov equivalence class, all possible DAGs obtained by adding exactly one edge to a DAG from this equivalence class and consider the set of essential graphs corresponding to this collection of DAGs. Let E_{n+1} denote the essential graph with the highest score if it has a higher score than E_n and continue to forward phase stage $n+1$. Otherwise, terminate the forward phase, with output E_n.

- *Backward phase* Let $\tilde{E}_0$ denote the output graph from the forward phase. Let $\tilde{E}_n$ denote the output graph from stage n of the backward phase. Consider all possible DAGs corresponding to the equivalence class $\tilde{E}_n$, all possible DAGs formed by an edge deletion from these DAGs and consider the set of essential graphs corresponding to this collection of DAGs. Let $\tilde{E}_{n+1}$ denote the essential graph with the highest score if it is higher than that for $\tilde{E}_n$ and continue to backward phase stage $n+1$. Otherwise terminate; $\tilde{E}_n$ is the output of the backward phase and of the greedy equivalence search algorithm.

After the forward and backward phase, this algorithm is guaranteed to return an optimal structure provided there exists a faithful DAG (definition 2). The faithfulness assumption may be relaxed; the algorithm returns a suitable structure provided the weaker *composition* condition holds (axiom 6 of compositional graphoid, equation 5). The compositional axiom is essential for the algorithm to return the correct graph.

6.4. Constraint Based Methods. Constraint based methods carry out tests, for triples (X, Y, S) where X and Y are variables and S is a subset of variables, to decide whether or not $X \perp Y|S$. The results of these tests are the *constraints* and a graph is then chosen that satisfies as many of the constraints, thus established, as possible.

For larger numbers of variables, these are overwhelmingly faster than search and score algorithms; for each structure tested, a search and score algorithm has to compute a score function that involves the entire data set. Testing whether $X \perp Y|S$ only requires the variables X, Y and those in S. In most applications, locating the structure only requires examining conditioning sets S that have a maximum of three or four variables.

The main problem is that these algorithms usually *require* that there exists a graph faithful to the distribution and use the following result:

Theorem 1 *Let $\underline{X} = (X_1, \ldots, X_d)$ be a random vector, with probability distribution p. A faithful DAG D contains an edge between two distinct variables X_i and X_j if and only if $X_i \not\perp X_j|S$ for any $S \subseteq V\backslash\{X_i, X_j\}$, where $V = \{X_1, \ldots, X_d\}$ is the variable set.* □

The following example illustrates a typical situation where constraint based algorithms in general perform rather badly.

Example 1

Let Y_1, Y_2, Y_3 be independent binary variables, each with probability function

$$p_Y(0) = p_Y(1) = \frac{1}{2}.$$

Let

$$X_1 = \begin{cases} 1 & Y_2 = Y_3 \\ 0 & Y_2 \neq Y_3 \end{cases} \qquad X_2 = \begin{cases} 1 & Y_1 = Y_3 \\ 0 & Y_1 \neq Y_3 \end{cases} \qquad X_3 = \begin{cases} 1 & Y_1 = Y_2 \\ 0 & Y_1 \neq Y_2 \end{cases}$$

The events $\{X_1 = 1\}$, $\{X_2 = 1\}$ and $\{X_3 = 1\}$ form the classic illustration in any first course in probability of three events that are pairwise independent, but not mutually independent. The variables (X_1, X_2, X_3) satisfy $X_i \perp X_j$ for $i \neq j$, but $X_1 \not\perp X_2|X_3$. In fact, the joint probability distribution for (X_1, X_2, X_3) is

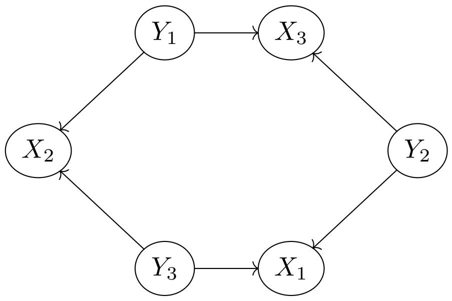

Figure 7: DAG for the natural factorisation; it is not faithful

$$\begin{cases} p_{X_1,X_2,X_3}(1,1,1) = \frac{1}{4}, \\ p_{X_1,X_2,X_3}(1,1,0) = p_{X_1,X_2,X_3}(1,0,1) = p_{X_1,X_2,X_3}(0,1,1) = 0, \\ p_{X_1,X_2,X_3}(1,0,0) = p_{X_1,X_2,X_3}(0,1,0) = p_{X_1,X_2,X_3}(0,0,1) = \frac{1}{4}, \\ p_{X_1,X_2,X_3}(0,0,0) = 0. \end{cases}$$

Note that

$$p_{X_1,X_2}(1,1) = p_{X_1,X_2}(1,0) = p_{X_1,X_2}(0,1) = p_{X_1,X_2}(0,0) = \frac{1}{4}$$

$$p_{X_1}(1) = p_{X_1}(0) = \frac{1}{2}.$$

Many constraint based algorithms assume that there exists a faithful DAG and remove an edge $X \sim Y$ as soon as they encounter a set S such that $X \perp Y|S$. Since $X_1 \perp X_2$, $X_1 \perp X_3$, $X_2 \perp X_3$, the graph for the three variables $\{X_1, X_2, X_3\}$ returned by such a procedure will be the empty graph; in other words, the graph corresponding to three mutually independent random variables. The fitted distribution will be

$$p_{X_1,X_2,X_3}(x_1, x_2, x_3) = \frac{1}{8} \qquad (x_1, x_2, x_3) \in \{0, 1\}^3.$$

This situation can loosely be described as follows; X_1 by itself gives no information about X_3 and X_2 by itself gives no information about X_3, but X_1 and X_2 taken together give everything about X_3. As Edward Nelson put it in 'Radically Elementary Probability Theory' in 1987 (see [62] page 11), 'this is the principle on which a good detective story is based.' Constraint based algorithms that assume faithfulness will be unable to solve the mystery.

If the set of six variables $(Y_1, Y_2, Y_3, X_1, X_2, X_3)$ is considered, then a suitable DAG to describe the distribution is given in figure 7, but since $Y_i \perp X_j$ for all (i, j), an algorithm based on the principle of theorem 1 would again produce an empty graph. Y_1 by itself gives no information about X_3 and Y_2 by itself gives no information about X_3, but Y_1 and Y_2 taken together give full information about X_3.

Note that (X_1, X_2, X_3) do not satisfy *composition*; $X_1 \perp X_2$ and $X_1 \perp X_3$, but $X_1 \not\perp \{X_2, X_3\}$. The search-and-score *greedy equivalence search* algorithm will fail with this distribution.

Some examples of constraint based methods, described in more detail below, are *Chow Liu tree*, *three phase dependency analysis*, the PC and MMPC algorithms, Fast algorithm and RAI algorithm. They all operate according to the principle of removing an edge $X \sim Y$ whenever a set S is located such that $X \perp Y | S$.

Chow Liu Tree. In many problems, the aim is not so much to find a structure that represents every aspect of the independence structure of the distribution; rather, it is to locate a part of the independence structure that can be used in applications. The Chow Liu tree, introduced by Chow and Liu [16] in 1968, finds the best fitting tree, which is used in the problem of classification. The application discussed by Chow and Liu is the automated recognition of hand written characters.

The *mutual information*

$$I(X,Y) = \sum \hat{p}_{X,Y}(x,y) \ln \frac{\hat{p}_{X,Y}(x,y)}{\hat{p}_X(x)\hat{p}_Y(y)}$$

is computed for each pair of variables (X,Y), where $\hat{p}$ denotes the empirical probability. The tree of maximal weight, i.e. the tree with the largest sum of mutual information is then obtained using Kruskal's algorithm, which amounts to first choosing the pair (X_i, X_j) such that $I(X_i, X_j)$ is largest and in subsequent steps choosing the pair with the largest mutual information available provided that addition of the edge does not form a cycle. This is one of the earliest structure learning algorithms and it is well known that Kruskal's algorithm returns the tree of maximal weight.

The algorithm for learning the Chow Liu tree has quadratic time and space complexity in the number of variables. If, to a good approximation, the probability distribution has a factorisation

$$p_{X_1,\dots,X_d} = p_{X_{\sigma(1)}} \prod_{j=2}^{d} p_{X_{\sigma(j)}|X_{\sigma(j-1)}} \tag{11}$$

for a permutation σ of $\{1,\dots,d\}$, then the algorithm by Kłopotek [43] in 2002 locates the Chow-Liu tree and represents a substantial improvement both in storage and run time; the space consumption grows linearly with the number of variables d while the execution time is proportional to $d\ln(d)$, by both measures a substantial improvement over the Chow - Liu algorithm for large numbers of variables. The economy comes from the *tree common sense assumption*, introduced by Kłopotek, which is satisfied if a distribution has a factorisation along a tree. The assumption is that if X is on the path from Z to A and X is on the path from Z to B, then $I(A,Z) > I(B,Z)$ if and only if $I(A,X) > I(B,X)$.

A distribution over four binary variables A, B, X, Z that factorises along the DAG in figure 8, factorisation given by equation (12) produces situations where the 'common sense tree assumption' is not satisfied.

$$p_{A,B,X,Z} = p_A p_B p_{Z|A} p_{X|A,B,Z}. \tag{12}$$

Clearly, there is no factorisation along a tree of the form given in equation (11) that accurately describes this distribution; (Z,X,B) and (B,X,A) are immoralities, which appear in any Markov equivalent factorisation, while a Chow-Liu tree has no immoralities. Here $A \perp B$ and $B \perp Z$, so that $I(A,B) = I(B,Z) = 0$. It is straightforward to construct distributions on $\{0,1\}^4$ (four binary variables) that

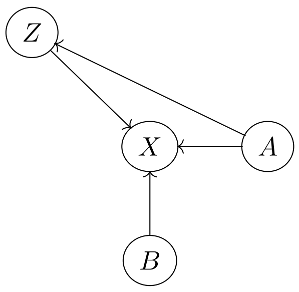

Figure 8: DAG for distribution that does not satisfy 'tree common sense' assumption

satisfy $I(B, X) > I(A, X)$, but $I(A, Z) > I(B, Z)$. It is debatable whether a Chow-Liu tree is appropriate for such distributions. Kłopotek's algorithm will produce the same result as Kruskal's algorithm when there is a Chow-Liu tree that accurately fits the distribution. In other situations, the results may be different.

The reader is referred to Kłopotek [43] for a complete description of the algorithm and proofs of efficiency. Briefly, Kłopotek seeks to construct a *junction tree* where the cliques are the edges of the Chow Liu tree. For a variable set $V = \{X_1, \dots, X_d\}$, the algorithm considers consecutively the sets $V_j = \{X_1, \dots, X_j\}$ for $j = 2, \dots, d$. At stage j, it forms a rooted junction tree for the variables in V_j, each clique of size 2. The position of the root is chosen in order to *balance* the tree (definition of 'balanced' in this context found in [43]). The choice of root, together with appropriate *focal* nodes in the edge tree (rooted junction tree) enable a reduction in the number of mutual informations $I(X, Y)$ that have to be computed, under the common sense tree assumption.

In 'Extended Structure Bayesian Networks' [44] in 2003, Kłopotek develops an efficient algorithm for finding a rooted junction tree for a probability distribution. In the Aalborg algorithm, where the Bayesian network is moralised, triangulated and then a junction tree is constructed from the triangulated graph, the most computationally expensive phase can be the triangulation. Kłopotek [44] presents a different approach, based on mutual information, for constructing a suitable Markov tree.

In 2010 Corander, Gyllenberg and Koski [19] consider a Bayesian approach to learning a Chow Liu tree, based on the principle of minimising stochastic complexity, for learning a genetic population structure where potential patterns of dependence are a priori unknown.

Three phase dependency analysis. The *three phase dependency analysis* algorithm (denoted TPDA) was introduced by Cheng, Greiner, Kelly, Bell and Liu [14] in 2002, who write, 'this TPDA algorithm is correct (i.e., will produce the perfect model of the distribution) given a sufficient quantity of training data whenever the underlying model is monotone DAG faithful.' The algorithm requires the faithfulness assumption to hold and relies on theorem 1. The TPDA algorithm works in three phases; *draughting*, *thickening* and *thinning*, outlined in Algorithm 1.

This outlines the main steps of the algorithm; a precise description of the algorithm and proof that it returns a faithful DAG when it exists, is given in [14].

Algorithm 1 The Three Phase Dependency Analysis Algorithm

```
Stage 1: Draughting Locate the Chow - Liu tree
    This stage is simply Kruskal's algorithm
Stage 2: Thickening Add edges
for $i = 1, \dots, d-1$, $j = i+1, \dots, d$ do
  Let $C_{i,j}$ be the set of neighbours $Z$ of $X_i$ or $X_j$ such that $Z$ lies on a path between
  $X_i$ and $X_j$
  if $X_i \not\perp X_j | C$ for any subset $C \subseteq C_{ij}$ then
    add an edge $X_i \sim X_j$
  else
    do not add an edge $X_i \sim X_j$
    and let $S_{i,j}$ denote the set such that $X_i \perp X_j | S_{ij}$
  end if
end for
Stage 3: Thinning Removing unnecessary edges
for $i = 1, \dots, d-1$, $j = i+1, \dots, d$ do
  Let $C_{ij}$ denote common neighbours of $X_i$ and $X_j$
  if $X_i \sim X_j$ and there is a set $C \subseteq C_{ij}$ such that $X_i \perp X_j | C$ then
    remove the edge between $X_i$ and $X_j$.
  end if
end for
Stage 4: Directing edges For each vee structure $X_i \sim X_k \sim X_j$, $(X_i, X_k, X_j)$ is an
immorality if $X_k \notin S_{ij}$, otherwise it is not. Once the immoralities have been added, the
additional compelled edges are obtained using Meek's rules (described later).
```

PC and MMPC algorithms. The PC algorithm was introduced by Spirtes, Glymour and Scheinesin [87] in 1993 and was modified to produce the MMPC algorithm by Tsamardinos, Brown and Aliferis [90] in 2006. It is algorithm for locating the skeleton of a *faithful* DAG, which may be used to construct the essential graph if the separating sets, sets $S_{X,Y}$ such that $X \perp Y | S_{XY}$ are recorded. It works in three stages. Firstly, a forward stage starts with an empty graph and proceeds according to Algorithm 2.
Note that edge removal is on the principle of theorem 1. Assuming faithfulness and a perfect oracle (that is, tests always give the correct results), there are possibly too many edges after this stage. Secondly, a backward stage removes some of the edges; Algorithm 3.

The algorithm may return false positives. Suppose a probability distribution may be represented by the DAG in figure 9. Working from T, the node C may enter the output, and remain in the output. This is because $C \not\perp T | S$, for any subset S of parents and children of T; namely, ϕ (the empty set) and $\{A\}$. Note that the *collider* connection TAB, is opened when A is instantiated so that, when A is instantiated and B is uninstantiated, T is d-connected with C. The connection TAC is a chain connection. Therefore, when A is uninstantiated, T is d-connected to C.

T and C are d-separated if and only if A and B are simultaneously instantiated; that is, $T \perp C | \{A, B\}$. But B is independent from T given the empty set, so the node B will be removed from the parent / child set of T and therefore the algorithm will not remove C from the set; the link TC will remain.

A third stage is implemented to remove the false positives; Algorithm 4.
This returns the complete parent / child set for X_i. The *sep-sets* for the variables removed in the third stage have already been established. The sep-set for a pair of

Algorithm 2 The MMPC Algorithm, Stage 1

```
for $j = 1, \ldots, d$ do
  Set $\mathcal{Z}_{j,0} = \phi$ (the empty set)
  for $i = 1, \ldots, d$ do
    if $i = j$ then
      Set $\mathcal{Z}_{j,i} = \mathcal{Z}_{j,i-1}$
    else
      if $X_i \perp X_j | \mathcal{Z}_{j,i-1}$ then
        Set $\mathcal{Z}_{j,i} = \mathcal{Z}_{j,i-1}$ and $S_{i,j} = \mathcal{Z}_{j,i-1}$ ($S_{i,j}$ the sep-set)
      else
        Set $\mathcal{Z}_{j,i} = \mathcal{Z}_{j,i-1} \cup \{X_i\}$
      end if
    end if
  end for
  Set $\mathcal{Z}_j = \mathcal{Z}_{j,d}$
end for
```

Algorithm 3 The MMPC Algorithm, Stage 2

```
for $j = 1, \ldots, d$ do
  Set $\mathcal{Y}_{j,0} = \mathcal{Z}_j$.
  for $k = 1, \ldots, d$, do
    if $X_k \in \mathcal{Y}_{j,k-1}$ and $\exists S \subseteq \mathcal{Y}_{j,k-1} \backslash \{X_k\}$ such that $X_j \perp X_k | S$ then
      Set $S_{j,k} = S$ and set $\mathcal{Y}_{j,k} = \mathcal{Z}_{j,k-1} \backslash \{X_k\}$.
    else
      Set $\mathcal{Y}_{j,k} = \mathcal{Y}_{j,k-1}$.
    end if
  end for
  Set $\mathcal{Z}_j = \mathcal{Y}_{j,d}$
end for
```

Algorithm 4 The MMPC Algorithm, Stage 3

```
for $j = 1, \ldots, d$ do
  Set $\mathcal{Y}_{j,0} = \mathcal{Z}_j$.
  for $k = 1, \ldots, d$ do
    if $X_k \in \mathcal{Y}_{j,k-1}$ but $X_j \not\in \mathcal{Z}_k$ then
      Set $\mathcal{Y}_{j,k} = \mathcal{Y}_{j,k-1} \backslash \{X_k\}$
    else
      Set $\mathcal{Y}_{j,k} = \mathcal{Y}_{j,k-1}$
    end if
  end for
  Set $\mathcal{Z}_j = \mathcal{Y}_{j,d}$
end for
```

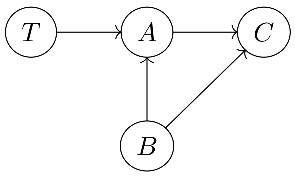

Figure 9: Min Max Parent Child: A False Positive

variables X, Y is the set $S_{X,Y}$ such that $X \perp Y | S_{X,Y}$, which caused the algorithm to remove the edge $X \sim Y$.

Establishing the Essential Graph. Having recorded the sep-sets, it is now straightforward to construct the essential graph. For each vee structure (X, Z, Y) (that is a structure such that $\{X, Y\} \subset \mathcal{Z}^{(Z)}$, but $X \notin \mathcal{Z}^{(Y)}$), check whether or not $Z \in S_{XY}$. If $Z \in S_{XY}$, then (X, Z, Y) is not an immorality; the edges $X - Z - Y$ remain undirected at this stage. If $Z \notin S_{XY}$, then (X, Z, Y) is an immorality.

Finally the additional directed edges in the essential graph are known as *compelled edges* and may be established using a straightforward set of rules known as *Meek's rules* from Meek [58]. These are that once the immoralities have been established, if the edge $\alpha - \beta$ appears in any of the structures in figure 10, it is directed as $\alpha \mapsto \beta$. Immoralities are determined by the results of the conditional independence tests. In the first of these structures, if the edge $\gamma \rightarrow \alpha$ is present and (γ, α, β) is not an immorality, then the direction $\alpha \rightarrow \beta$ is force. In the second, if $\alpha \rightarrow \gamma$ and $\gamma \rightarrow \beta$ are present, then $\alpha \rightarrow \beta$ is forced to prevent a cycle. In the third, if the immorality $(\gamma_1, \beta, \gamma_2)$ is present, but $(\gamma_1, \alpha, \gamma_2)$ is not an immorality, then $\alpha \rightarrow \beta$ is forced to prevent cycles.

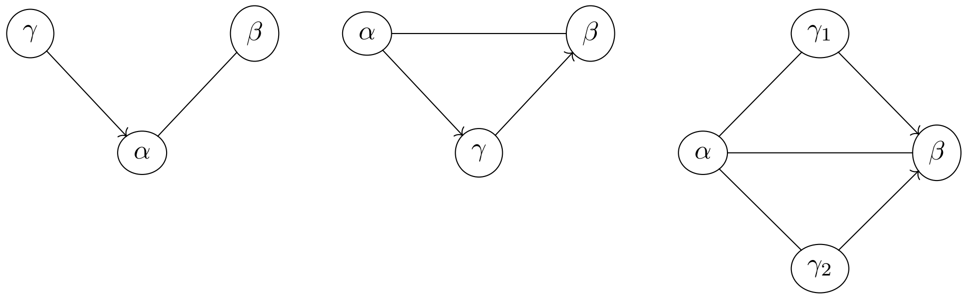

Figure 10: Meeks rules: the undirected edge $\alpha - \beta$ is given the direction $\alpha \mapsto \beta$ if it appears in one of these configurations.

Constraint based algorithms and hypothesis testing. One serious problem with constraint based algorithms is that even if the data is generated by a probability distribution with a faithful graphical representation, the power of the χ^2 tests used to establish independence decreases rapidly as the size of the conditioning sets increases, according to the inverse of the size of the conditioning set. With a test $H_0 : X \perp Y|S$ versus $H_1 : X \not\perp Y|S$, independence is *accepted* and hence the edge $X - Y$ *removed* from the skeleton whenever the null hypothesis of independence is not rejected. This, of course, violates an elementary statistical principle that is taught whenever the subject of hypothesis testing is introduced, even at the most basic level; a null hypothesis is never accepted. This can have rather serious consequences; an edge $X - Y$ can be removed simply because the test is unsatisfactory due to a large conditioning set, even if this results in a graph where X and Y are d-separated and it has been established through previous tests, where the null hypothesis has been rejected and the alternative hypothesis accepted, that $X \not\perp Y$. Other algorithms have been developed with this in mind, to keep the size of the conditioning sets as small as possible.

The FAST algorithm. The FAST algorithm, due to Andrew Fast [26], takes this into account. It starts with the complete undirected graph. Suppose the variable set is $X_1, \ldots, X_d$, then the algorithm is described in Algorithm 5.

Algorithm 5 The FAST algorithm

```
output the essential graph
initialisation complete undirected graph with $d$ nodes
for $k \geqslant 0$ do
  for $1 \leqslant i \leqslant d-1$, $i+1 \leqslant j \leqslant d$ do
    if there is an edge $X_i \sim X_j$ then
      test for $X_i \perp X_j|C$ for all sets $C$ such that $|C| = k$. If there is a set $C$
      such that the conditional independence statement holds, remove the edge
      $X_i \sim X_j$ and set $S_{ij} = C$
    end if
  end for
end for
Termination: the algorithm terminates either when all pairs $X_i, X_j$ have fewer
than $k$ common neighbours, or else a pre-assigned value of $k$ is reached.
Edge direction For vee structures $X_i - Z - X_j$, if $Z \notin S_{ij}$, direct the edges
$X_i \rightarrow Z \leftarrow X_j$. If $Z \in S_{ij}$, then leave the edges unaltered
Compelled edges After locating the immoralities, apply Meek's rules to find all
the directed edges of the essential graph
```

The algorithm is run either until all variable pairs have fewer than k common neighbours, or until some pre-determined level of k is reached (usually 3 or 4), beyond which the conditional independence tests are considered unreliable.

As with the MMPC algorithm a vee-structure $X - Z - Y$ in the resulting graph is declared an immorality if $Z \notin S_{X,Y}$ (the sep-set for X, Y) and is not an immorality otherwise. Meek's rules are then used to locate the other compelled edges.

Recursive Autonomy Identification. The *Recursive Autonomy Identification algorithm*, introduced by Yehezkel and Lerner in [96], proceeds under the assumption that there is a faithful graphical model and removes an edge $X - Y$ whenever there is a test result $X \perp Y|S$ for some S. Like the Fast algorithm, it starts with the smallest conditioning sets first and works upwards. It employs more of the structure than Fast's algorithm, which reduces the number of tests that have to be performed.

The additional structure used is that of the *essential graph*. At each stage of the algorithm, the immoralities produced by edge deletions in the current round are inserted. The essential graph is a *chain graph*. This means that the variable set V may be split into p disjoint subsets; $V = V_1 \cup \ldots \cup V_p$, where the graph, restricted to V_j, is undirected for each j and, for $i \neq j$, $\alpha \in V_i$ and $\beta \in V_j$, there is no cycle containing both α and β. (With a cycle, the direction of the edges is observed. $(\alpha_0, \ldots, \alpha_m)$ is a cycle if $\alpha_0 = \alpha_m$, $\alpha_i \neq \alpha_j$ for all other pairs (i, j) and for each $i, i+1$ there is either an undirected edge $\alpha_i - \alpha_{i+1}$ or a directed edge $\alpha_i \to \alpha_{i+1}$). The essential graph also satisfies the condition that the chain components are triangulated.

The first step of the RAI algorithm is given by Algorithm 6.

Algorithm 6 The RAI algorithm, step 0

```
**Initialization** Starting with a variable set $V = \{X_1, \ldots, X_d\}$, the initial graph is
the complete graph, with undirected edges between each pair of variables
**for** $1 \leqslant i < j \leqslant d$ **do**
    Test whether or not $X_i \perp X_j$. If true, the edge $X_i \sim X_j$ is removed.
    Record $S_{i,j} = \phi$, the empty set ($S_{i,j}$ is the *separator* or *sep-set*).
    **for** each vee structure $X_i - Z - X_j$ where there is no edge $X_i - X_j$ **do**
        direct edges so that the triple $(X_i, Z, X_j)$ is an immorality $X \to Z \leftarrow Y$.
    **end for**
**end for**
Apply Meek's rules, to direct the additional compelled edges
```

This produces a graph that is an essential graph. After this initialisation (stage 0), the algorithm proceeds recursively. Algorithm 7 gives the nth stage of the RAI algorithm. For a graph $\mathcal{G} = (V, E)$, the notation $\mathcal{G}_D$ denotes the sub-graph $(D, E \cap D \times D)$.

The RAI proceeds either until the size of the largest neighbour set in the undirected graph is equal to n (the size of the conditioning set in the current round) or until the algorithm has performed a pre-assigned number of rounds, governed by the reliability of the conditional independence tests. The output is the resulting essential graph.

This additional use of the structure by the RAI algorithm can reduce the number of tests required in comparison with the Fast algorithm.

Resolving contradictions. The constraint based algorithms discussed so far, the TPDA, MMPC, Fast and RAI, all rely on the results of conditional independence tests. There are two difficulties that can arise: firstly, the independence structure may not satisfy a faithfulness condition. Secondly, even if it does, with a finite data set, there is not a perfect oracle; the results of the CI (conditional independence) tests are not entirely reliable. A. Fast discusses the issues raised by the second of these difficulties at length in [26]. For example, the MMPC, Fast and RAI algorithms may produce contradictory immoralities; $X \to Y \leftarrow Z$ and $Y \to Z \leftarrow W$.

Tsamardinos, Brown and Aliferis [90] in 2006 propose the following solution: after running the MMPC algorithm, they do not try to direct the edges and simply take the skeleton. To produce a DAG, they take the skeleton from MMPC as the candidate edge set and run the MMHC (Maximum Minimum Hill Climbing) algorithm, a search

Algorithm 7 The RAI algorithm, step n for $n \geqslant 1$

```
Starting with a chain component that has no descendants and proceeding backwards, consider in turn each chain component $\mathcal{G}_C$ and the sub-graph $\mathcal{G}_D$ formed by taking the chain component $\mathcal{G}_C = (C, U_C)$ together with the chain components that have parent variables of $\mathcal{G}_C$ and all all the directed edges connecting these chain components.
for each $Y \in C$ and each neighbour $X$ of $Y$ (consider first the parents in different connected components, and then the undirected neighbours in the component $\mathcal{G}_C$) do
    check whether there is a set $S_{XY} \subset D$ of size $n$ such that $X \perp Y | S$.
    if true then
        remove the edge between $X$ and $Y$ and record $S_{XY}$.
        For each new vee structure $(X, Z, Y)$, declare it to be an immorality if and only if $Z \notin S_{XY}$. Find the additional compelled edges generated by the immorality.
    end if
    Remove the chain component $\mathcal{G}_C$
end for
proceed recursively until the whole graph has been considered.
```

and score algorithm. Starting with an empty graph, at each stage the best add / delete / reverse edge is chosen, where only edges from the skeleton obtained by the MMPC algorithm are considered, according to which operation gives the highest scoring graph. The results in that article indicate that, for the largest test network attempted, the search and score edge orientation phase takes 10 times as long as the constraint based MMPC skeleton finding phase.

Fast takes a different approach, largely on philosophical grounds. Search-and-score presents one criterion for deciding on the best graph, the one that produces the highest score. Constraint based presents an entirely different criterion. The result of each independence test is a 'yes' or a 'no' and the best fitting graph is the graph that satisfies as many of these constraints as possible. To deal with contradictory immoralities, Fast develops the 'EDGE-OPT' algorithm, which finds the graph that satisfies the largest number of constraints.

The results of conditional independence tests should be consistent, in the sense that they should satisfy *decomposition*, *contraction*, *weak union* and *intersection*. If different CI tests produce results that contradict these relations results, then Bromberg and Margaritis [4] in 2009 propose a system of *argumentation*, based on the power of the tests for deciding which results to accept into the set of constraints. These systems of argumentation seem to be in their initial stages; an assumption in Bromberg and Margaritis is that the random variables corresponding to different test statistics are independent.

The Xie - Geng Algorithm. The Xie - Geng algorithm, by Xie and Geng [95] is a constraint based algorithm that takes a different approach. It is not suitable for sets of *discrete* variables, because the conditioning sets for the CI tests are too large, but works well with multivariate Gaussian variables that satisfy a faithfulness assumption.

The algorithm first constructs the *independence graph*, an undirected graph, which contains an undirected edge $X - Y$ for each pair of variables such that $X \not\perp Y | V \backslash \{X, Y\}$ (X not independent of Y conditioned on all the other variables in the network). In the case of Gaussian variables, this simply reduces to testing whether or not the conditional covariance between the two variables is significant. The authors suggest considering the inverse of the statistical covariance matrix and adding an edge corresponding to each entry that is significantly different from zero. This can be unstable if the number of data is small compared with the number of variables or if the statistical covariance matrix is singular (or close to singular) for some other reason. A generalised inverse is unreliable for this procedure. The conditional covariance for the two variables conditioned on the other variables can always be computed, even when the sample covariance matrix is close to singular.

The *independence graph*, containing an edge if and only if $X \not\perp Y | V \backslash \{X, Y\}$, has the property that graphical separation implies independence. (Note, this is separation in the usual sense for undirected graphs and not d-separation).

Recall that in a Bayesian network, a variable X is d-separated by its Markov blanket $MB(X)$ (definition 3) from all the other variables in the network and hence independent of the rest of the network, conditionally on $MB(X)$. Therefore, in the moral graph of a Bayesian network (the undirected graph obtained by linking all the parents of a variable for each variable with undirected edges and then un-directing all the edges), a variable is graphically separated from all the other variables in the network by its Markov blanket.

The edge set of the independence graph is therefore a subset of the moral graph of any DAG corresponding to a factorisation of the distribution; the independence graph and moral graph are equal if the DAG is faithful. The Xie - Geng algorithm takes the view that the independence graph is the moral graph of a faithful DAG. It starts by recursively decomposing the graph, until it has decomposed it into cliques. It then uses the 'faithfulness' principle when dealing with the cliques to locate immoralities, delete unnecessary edges in the cliques and direct the vee-structures corresponding to immoralities. For a clique C and set $S \subseteq C \backslash \{X, Y\}$, if $X \perp Y | S$, then the edge $X - Y$ is removed and for each $W \in C \backslash (S \cup \{X, Y\})$, the directions $X \to W \leftarrow Y$ are given to remaining vee structures $X - W - Y$.

It then re-assembles the pieces according to the following principle: an edge $X - Y$ belonging to a structure is removed at re-assembly if it has already been removed from one of the two structures being merged; for any variable W that is not in the sep-set $S_{X,Y}$ (the set that caused $X - Y$ to be removed) where edges $X - W$ and $Y - W$ are present, these edges are directed to form an immorality $X \to W \leftarrow Y$.

6.5. Hybrid Algorithms. Hybrid algorithms combine constraints with search and score. One example is the MMHC algorithm of Tsamardinos, Brown and Aliferis [90], which starts with the constraint based MMPC stage to locate the skeleton and then carries out a search and score based MMHC stage, using the skeleton obtained from MMPC as the candidate edge set.

L1-Regularisation. One method, introduced by Schmidt, Niculescu-Mizil and Murphy [82] in 2007, places constraints on the model and then uses an L^1 score function, described below, as the basis of a search and score within the constrained space.

The method can be employed with Gaussian or binary variables. The binary case is outlined here.

In this algorithm, there is no restriction on the number of parents that a variable may have, but there is a constraint on the way in which the parents influence the variable. The state space of variable X_j is $\{-1, 1\}$ for each j and the conditional probabilities are modelled so that the *logit* function is linear:

$$\ln\left(\frac{p_{X_j|\Pi_j}(1|\underline{\pi}_j)}{1 - p_{X_j|\Pi_j}(1|\underline{\pi}_j)}\right) = \left(\theta_{j,0} + \sum_{k=1}^{p_j} \theta_{j,k}\pi_{j,k}\right) \tag{13}$$

where $\underline{\pi}_j = (\pi_{j1}, \dots, \pi_{jp_j})$, the configuration of Π_j, is a sequence of ± 1 corresponding to the states of the parent variables. The parent variables are only permitted to influence $\ln \frac{p}{1-p}$ linearly; no interactions are permitted. This permits a large number of parents, since the number of parameters is linear, rather than exponential, in the number of parents.

The algorithm works in two stages: like the MMPC, it first produces candidate parent children sets for each variable. Having constrained the search space, it then uses a search and score algorithm to determine the candidate parent / children sets. Having determined the parent / children sets, it runs the hill climbing part of the MMHC algorithm of Tsamardinos, Brown and Aliferis to obtain the structure, keeping the conditional probabilities of the form in equation (13). For a vector $\underline{x} = (x_1, \dots, x_d)$ of 1's and -1's, let Π_j denote all the variables without j. That is, all variables permitted as possible parents for j at this stage. Let $\underline{\tilde{x}}^{(j)}$ denote the vector $\underline{x}$ without x_j. Let

$$LL(j, \underline{\theta}_j, \underline{x}) = \log p_{X_j|\Pi_j}(x_j|\underline{\tilde{x}}^{(j)})$$

denote the log likelihood function and, for $\mathbf{x}$ the data matrix with rows $\underline{x}_{(1)}, \dots, \underline{x}_{(n)}$, let

$$LL(j, \underline{\theta}_j, \mathbf{x}) = \sum_{k=1}^{n} LL(j, \underline{\theta}_j, \underline{x}_{(k)}).$$

The parameters $\underline{\theta}_j$ are chosen to maximise the $L1$ regularisation score function,

$$L_1R(\underline{\theta}_j, \mathbf{x}) = LL(j, \underline{\theta}_j, \mathbf{x}) - \lambda\|\underline{\theta}_j\|_1,$$

where $\|\underline{\theta}_j\|_1 = \sum_{k=1}^{d-1} |\theta_{jk}|$ and λ is chosen appropriately. The sum is over the parameters corresponding to dependence on parent variables; the parameter $\theta_{j,0}$ is not included. The article [82] has some discussion about the appropriate choice of λ.

The L^1 regularisation, if λ is appropriately chosen, has the effect of choosing vectors $\underline{\theta}_j$ with a substantial number of zero components. Because of this property, it tends to favours a lower number of parameters in the model. For this reason, L^1 regularisation is a technique that is developing increasing importance.

Gibbs sampling. A related approach to the problem of structure learning is found in 2005 by Bulashevska and Eils [5]. The structure learning algorithm is intended for analysis of gene expression data, to locate gene regulatory interactions. As with Schmidt, Niculescu-Mizil and Murphy [82], the parents influence the offspring independently of each other and the algorithm forms 'noisy OR' and 'noisy AND' gates.

The parent sets are chosen using Gibbs sampling. The generic techniques of Gibbs sampling are found by Gammerman and Lopes [32] in 2007.

6.6. Faithfulness and 'real world' data. The Recursive Autonomy Identification algorithm was analysed by B. Barros [2] in 2012, applying it both to data simulated from test networks and to a financial data set. When applied to simulated data, simulated from the ALARM network, the algorithm performed very well; the performance was consistent with the results described by Yehezkel and Lerner [96]. For a data set generated by a probability distribution for which there exists a faithful DAG, the results verified that the algorithm is efficient and produces a graph that corresponds well to the distribution that generated the data, with low computational overheads. The feature of the algorithm of making all required tests with smaller conditioning sets before moving on to larger increases accuracy over methods that do not do this. The additional use made of the structure, identifying the chain components of the essential graph at each stage, ensures that fewer statistical calls (references to the data set) are required.

Some features were noted in the performance of the algorithm. In earlier stages, some contradictory directions appeared. That is, pairs of immoralities $X \rightarrow Y \leftarrow Z$, $Y \rightarrow Z \leftarrow W$, in situations where the edge $Y \sim Z$ would be deleted in subsequent rounds of the algorithm following tests with larger conditioning sets. The direction chosen for the edge during that round was dictated by which immorality appeared first. If the test $X \perp Z | S_{X,Z}$, yielding a sep-set $S_{X,Z}$ was carried out first, then the edge would take the direction $Y \leftarrow Z$. After carrying out the CI tests and determining the directions, Meek's orientation rules were applied to determine the structures for the next round of the algorithm.

The algorithm worked very well; with 10000 observations, it produced a graph that had the correct skeleton and only 4 edges with incorrect orientation.

The test of performance of an algorithm, MMHC, RAI, TPDA, or any other algorithm, is based on the ability of the algorithm to recover a probability distribution used to simulate data. There are several standard networks, including the ALARM network, that are used. Data is simulated from the network and the algorithm applied to the simulated data. Freedman and Humphreys [28], p 33,34, are somewhat scathing in their assessment of this procedure for verifying the utility of an algorithm, of using simulated data from a distribution known to have good properties. They write,

> The ALARM network is supposed to represent causal relations between variables relevant to hospital emergency rooms, and Spirtes Glymour Scheines (1993) [85] p 11 claim to have discovered almost all the adjacencies and edge directions 'from sample data'. However, these 'sample data' are simulated; the hospitals and patients exist only in the computer program. The assumptions made by SGS (1993) [85] are all satisfied by fiat, having been programmed into the computer: the question of whether they are satisfied in the real world is not addressed. After all, computer programs operate on numbers, not on blood pressures or pulmonary ventilation levels (two of the many evocative labels on nodes in the ALARM network).

Freedman and Humphreys continue by stating, These kinds of simulations tell us very little about the extent to which modelling assumptions hold true for substantive applications.

The constraint based algorithms TPDA, MMPC, Fast and RAI all depend crucially on the modelling assumption that there is a DAG that is faithful to the set of conditional dependence / independence statements that can be established. We pinpoint two difficulties that can arise in the 'real world'; interaction effects without main effects and hidden common causes.

Interaction effects without main effects. Example 1 gives an example of a situation where these constraint based algorithms will miss key associations between the variables. Any situation where factors taken individually give no information, but where there are two-factor, or higher order factor interaction without main effects, will not be detected. If applied to genetic data, for example, the algorithm will not be able to detect situations where a single gene by itself has no apparent effect, but where the genome pathway may be opened by two genes acting together.

This situation will not lead to internal inconsistencies in the functioning of the algorithms; associations of this type will simply be missed and the output will be a DAG that does not show these associations, but it may not lead to reversed edges (situations where the algorithm has to choose between two contradictory directions for an edge).

Hidden variables. In a 'real world' situation, there may well be hidden variables which are not measured and the experimenter may be unaware of their existence. This can lead to reversed edges, as the following example illustrates. Suppose that X, Y, Z, W are variables that are recorded, while H is a hidden variable, a common cause of X and Y, whose presence is not suspected by the researcher. Suppose that the causal relations between H, X, Y, Z, W are given by figure 11.

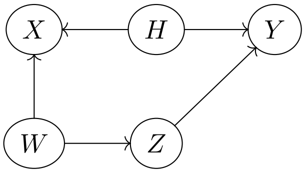

Figure 11: H is hidden and does not appear in the data matrix

If the RAI algorithm is applied to the variables X, Y, Z, W, whose associations are described by the d-connection statements of the DAG in figure 11, then $X \perp Z | W$, giving $X \rightarrow Y \leftarrow Z$ and $Y \perp W | Z$, giving the immorality $Y \rightarrow X \leftarrow W$. Even if there is a perfect oracle (sufficient data to give correct results for each CI test so that the results are consistent with the probability distribution over (X, Y, Z, W)), the edge between X and Y is a *reversed edge*, $X \leftrightarrow Y$. This notation means that, from the CI tests, one test gives a direction $X \rightarrow Y$; the other gives a direction $X \leftarrow Y$ and the algorithm will choose the direction depending on the order in which the tests are carried out.

In the RAI algorithm, the direction that an edge takes in the output graph, under such circumstances is determined by the order of the variables; if the test results $X \perp Z|W$ appears first, the output graph will contain $X \to Y$ and thus the graph will contain the false d-separation statement $W \perp Y|\{X, Z\}$, while if the result $W \perp Y|Z$ appears first, the output graph will contain the edge $Y \to X$ and the false d-separation statement $X \perp Z|\{W, Y\}$. The two possibilities are given in figure 12.

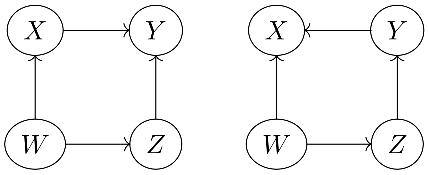

Figure 12: Possible outputs applying constraint based algorithm to variables (X, Y, Z, W) from figure 11

Contradictory results from CI tests. Suppose that the test results give $X \not\perp Y$. One would therefore hope that X and Y are d-connected in the output graph. The decision to remove an edge $X \sim Y$ as soon as a test result 'do not reject $X \perp Y|S$' appears can lead to a situation where the edge $X \sim Y$ is removed, even if it leads to a graph where $X \perp Y\|_{\mathcal{G}}\phi$ (X and Y are d-separated by the empty set). The constraint based algorithms discussed here do not have a mechanism to ensure that the final graph contains those d-connection statements that have been established through rejecting independence.

The scope of structure learning. Algorithms can detect associations, at the level of 'descriptive statistics', without reference to the process that generates the data and the nature of randomness. At the level of descriptive statistics, the scope of constraint based algorithms is viewed along the following lines: from the $n \times d$ data matrix, an empirical distribution can be established (or, at least, if d is very large, empirical probability distributions of the marginalisation to subsets of the variables can be established). Any test result that produces $X \not\perp Y|S$ corresponds to a d-connection statement that is to be retained in the output graph; any test result where $X \not\perp Y|S$ is not rejected does not have to be retained in the output graph. The output graph attempts to have as few edges as possible, while retaining all the d-connection statements that were established through rejecting independence.

For large numbers of variables, there are clear difficulties that make serious inferential statistics impossible. The assumption is that the $n \times d$ data matrix represents n independent instantiations of a d-random vector $\underline{X}$. This assumption, together with an assumption that n is sufficiently large for a central limit theorem effect to hold is required for the test statistics to be approximately χ^2. Even if the nominal significance level α chosen for rejecting a null hypothesis can be considered as a measure of a probability in any serious way the number of tests required is large that the overall significance level could be close to 1. In terms of descriptive statistics, the output graph can be informative, but it is difficult to reach inferential conclusions from the output of these algorithms.

Application of Fast and RAI to financial data. After testing the Fast and RAI algorithms on the training example of the ALARM network, where it performed well, the work of Barros [2] proceeded to run these algorithms on a financial data set, composed of the closing values of 18 stock market indices (Amsterdam stock index, Austrian traded index, Brussels stock index, etc ...) from 1st January 2005 to 1st January 2011, approximately 1000 instantiations of 18 variables.

The aim of the thesis was to detect *changes* in associations between the variables, to learn a structure, detect when the structure was no longer appropriate and update.

In the financial data set, the raw RAI algorithm gave no independence statements after the first round; for each pair of variables (X, Y), the result was 'reject independence'. Therefore, any pair of variables should be d-connected in the output graph. Yet the output graph, following application of the raw RAI algorithm, gave pairs of d-separated variables, which indicates that conditional independence was falsely accepted due to weak tests.

In order to deal with the situation where 'accept independence' from tests with large conditioning sets contradicted d-connection statements with lower order conditioning sets, Barros adopted a more conservative approach than the argumentation of Bromberg and Margaritis [4] and modified the algorithm so that it did not accept an independence statement that resulted in a d-separation in the output graph contradicting a dependence statement that has already been established. This modification worked well.

The output still gave a large number of 'reversed edges'. While the ALARM network gave one or two, the financial data set gave approximately 28 reversed edges, indicating situations that appeared in the DAG in figure 11, with possible output graphs corresponding to figure 12.

The presence of a *substantial* number of 'common cause' hidden variables would explain this.

This was a randomly chosen 'real world' data set and probably not appropriate for an algorithm based on a 'faithfulness' assumption. The variables here do not satisfy one of the motivating features of the faithfulness assumption, that the variables stand in *causal* relation to each other; their association is more likely to be a result of hidden common causes, such as government policies, or global financial considerations that influence the various stock markets.

The same difficulties seemed to arise in other applications. The RAI algorithm was applied to the genetic data found in Friedman et. al. [27]. Tentative results seem to give substantially different output depending on the input order of the variables, suggesting hidden common causes.

Conclusion. Constraint based algorithms offer a fast approach, which is convenient with data matrices when d, the number of variables, is very large. They can be many times faster than search and score algorithms. Unfortunately, these algorithms tend to assume 'faithfulness' and work on the principle of removing an edge whenever a conditional independence test gives the result 'do not reject $X \perp Y|S$'. This leads to several difficulties. Firstly, since tests with larger conditioning sets are weaker, it can lead to situations where deletion of an edge can contradict earlier d-connection statements. This difficulty is present even if there is a faithful DAG corresponding to the independence structure. Secondly, two-factor, or higher order interactions are not detected if there are no 'main effects'. Thirdly, hidden variables can lead to contradictory edges, resulting in d-separation statements not present in the probability distribution. If there is no faithful DAG that describes the underlying independence structure, this can manifest itself in other ways.

Modifications to remove the first of these difficulties have been considered, for example by Bromberg and Margaritis [4] using argumentation and the more conservative approach of Barros [2] retaining all dependence statements that have been established through rejecting independence.

The second and third of these difficulties have not been fully addressed by constraint based algorithms.

7. The 'Causal Discovery' Controversy.

The discussion about structure learning has described various methods to locate structures that represent the independence relations within a data set. All these methods, search and score, constraint based, hybrid, yield results that fall under the heading of *descriptive statistics*. The search and score methods simply examine some of the available structures and choose the structure with the highest score of those examined. On the 'classical' side, there is no measure of confidence for the structure chosen; on the 'Bayesian' side, even if a prior distribution is placed over the structure space and the posterior used as the basis of a score function, there is no posterior assessment of the probability for the structure to lie in a certain subspace of the set of possible structures; only a small number of structures are visited and the structure chosen is the one visited that gives the largest score. With constraint based methods, even if the hypothesis that the data matrix represents n instantiations of i.i.d. random vectors holds, the number of tests is so large that even with a small nominal significance level for each test, the overall significance level approaches 1.

The output structure can give useful information at the level of descriptive statistics, but little or no formal inference can be made. This is generally the case in multivariate statistics, where methods are often more successful as descriptive than inferential tools.

Assume, though, that statistical associations have been established. Substantial parts of the literature suggest claims that a rigorous engine for inferring *causation* from association has been established. For example, Spirtes, Glymour and Scheines [85] (further references as SGS) claim to have algorithms for discovering causal relations based only on empirical data. The underlying assumption seems to be that, for a large class of problems, when immoralities are learned from data and Meek's rules then applied, cause to effect can be inferred for the directed edges of the essential graph. In 2007 Schmidt, Niculesu-Mizil and Murphy [82] write, explaining why they are constructing techniques to produce *directed* graphs,

> '... undirected models cannot be used to model causality in the sense of Pearl [69], which is useful in many domains such as molecular biology, where interventions can be performed.'

The thrust of the quote is that directed edges whose direction can be interpreted as cause to effect, can be learned from data. But placing a causal interpretation on a directed arrow in a p-DAG that has been learned purely by applying a structure learning algorithm to data can be misleading.

In a situation where interventions can be performed, a *causal* directed graph can be obtained from the undirected graph through further controlled experiments. Consider the situation on three variables (X, Y, Z) where $X \perp Z | Y$, but $X \not\perp Y$,

$X \not\perp Z$, $Y \not\perp Z$, $Y \not\perp X|Z$ and $Y \not\perp Z|X$. There are three DAGs along which the distribution $p_{X,Y,Z}$ may be factorised, given in figure 13.

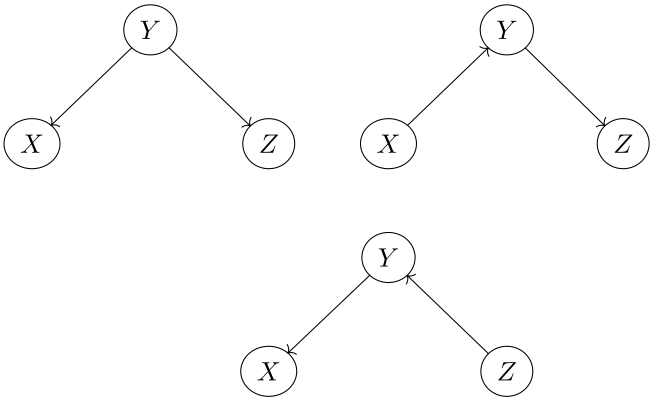

Figure 13: Three Markov equivalent DAGs

Suppose that an intervention may be carried out on the variable Y, forcing its state. This has the effect of removing arrows from parents of Y to Y. If the state $Y \leftarrow y$ is forced, this gives the graphs in figure 14. If all the states of Y can be explored, in a controlled experiment, by randomly assigning levels of the 'treatment' variable Y, the causal structure can be determined from the Markov structure, but not otherwise.

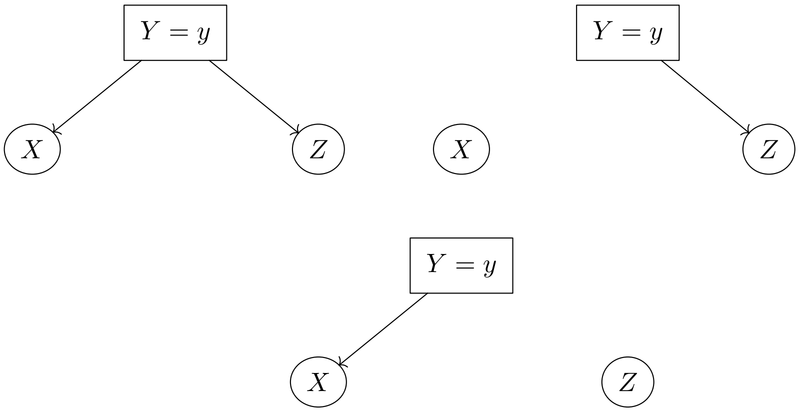

Figure 14: Intervention $Y \leftarrow y$ in figure 13

Markowetz and Spang [57] discuss the application of intervention calculus for perturbation experiments that are inferring gene function and regulatory pathways.
As Freedman and Humphreys [28] point out in 1999, commenting on automated causal learning, 'these claims are premature at best and the examples used in SGS to illustrate the algorithms are indicative of failure rather than success.' They point

out that 'the gap between association and causation has yet to be bridged.'

7.1. Faithfulness and the great leap of faith. One of the leading assumptions behind 'causal discovery' is the assumption that distributions of interest satisfy the faithfulness assumption, that there is a DAG $\mathcal{G}$ with variable set $V = (U, O)$ where U denotes the unobserved variables and O the observed variables and a probability distribution $\mathbb{P}$ over (U, O) such that $\mathbb{P}$ factorises along $\mathcal{G}$ and $\mathcal{G}$ gives a faithful graphical representation of the independence structure.

This is described as follows;

> ' .... the faithfulness condition can be thought of as the assumption that conditional independence relations are due to causal structure rather than to accidents of parameter values.' Spirtes et. al. (2000) [86]

Another statement of the same principle is found in in 1995 by Meek [58]

> In cases where $\mathcal{P}(\mathcal{G})$ (the set of distributions that factorise along a graph $\mathcal{G}$) can be parametrised by a family of distributions with a parameter of finite dimensions, the set of unfaithful distributions typically has Lebesgue measure zero. (Spirtes et. al. (2000) [86] pp 42 - 2)

This assumption, that the set of observable variables O may be extended to a set $V = (U, O)$ where U represents unobserved common causes, or confounders, and that there will exist a DAG over V that is faithful to the probability distribution over V, is re-stated in Robins, Scheines, Spirtes and Wasserman [78]. There is strong interest in classes of faithful distributions in the literature; the work of Zhang and Spirtes [97] requires that the class of distributions under consideration satisfy a stronger assumption than faithfulness in order to obtain uniform consistency in causal inference for a certain class of problems; Robins et. al. [78] illustrates non-existence of uniform consistency when only faithfulness is assumed, because of the possibility of non-faithful distributions in the closure of the set of distributions under consideration.

Example 1 gives an instance of a situation where the probability distribution does not have faithful graphical representation. For the variables $(Y_1, Y_2, Y_3, X_1, X_2, X_3)$, the DAG that best represents the associations between the variables is given by figure 7. For these variables, $X_1 \perp Y_2$ and $X_1 \perp Y_3$, but $X_1 \not\perp \{Y_2, Y_3\}$, *composition* is not satisfied. In this situation the influence of Y_2 and Y_3 on X_1 is not seen if the variables are considered separately, but the interaction effect is decisive. Suppose that $O = (X_1, X_2, X_3)$, the values for (X_1, X_2, X_3) are observable and $U = (Y_1, Y_2, Y_3)$, the results of (Y_1, Y_2, Y_3) are hidden. Clearly, the set of distributions over 6 binary variables that factorises over the DAG in figure 7 can be described by a finite parameter space; 15 parameters are required to describe the entire set of distributions; the parameter space is $[0, 1]^{15}$. Furthermore, it is clear that the parameters to describe the distribution over $(Y_1, Y_2, Y_3, X_1, X_2, X_3)$ in example 1 correspond to exactly one point in the parameter space, which has Lebesgue measure zero. Nevertheless, examples where knowledge of two causes is required to explain the effect and where knowledge only of a single cause tells you nothing about an effect arise all the time in practise, in the real world.

Furthermore, the parametrisation of any distribution that has an independence structure has Lebesgue measure zero in the parameter space of all distributions over the variables in question. Meek's argument can equally well be used to argue against searching for any independence structure at all.

Faithfulness appears a convenient hypothesis to produce beautiful mathematics (and the relation between DAGs and probability distributions under this assumption has produced a very elegant and attractive mathematical theory), but it is difficult to see that it necessarily applies to real world situations; the real world does not respect the fact that the set of parameters that describe the situation have Lebesgue measure zero in a mathematical parameter space. Divergence between 'real world' behaviour and the assumption that it should fit into a convenient mathematical framework has been termed 'The Mind Projection Fallacy' by E.T. Jaynes [39].

7.2. Inferring non-causation and causation. In 2003 Robins, Scheines, Spirtes and Wasserman [78] describe situations where *non*-causation can be inferred. A situation where such an inference can be made is given by figure 11 representing the *causal* associations between variables, where H is hidden and X, Y, W are observable. In this example, X is not a cause of Y, neither is Y a cause of X. This can be inferred from the CI tests; from the results $X \perp Z | W$ and $Y \perp W | Z$, it is possible to infer that the relation between X and Y is not cause to effect in either direction and that a common cause H would explain the test results.

The discovery of an immorality, though, does not necessarily imply causation. Suppose H_1 and H_2 are hidden and X, Z, Y are observable in figure 15. The distribution over (X, Z, Y) factorises according to figure 16.

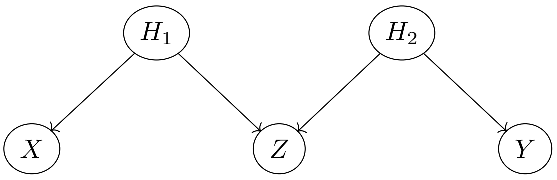

Figure 15: H_1 and H_2 hidden

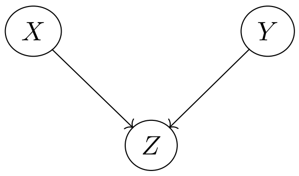

Figure 16: DAG for (X, Y, Z) from figure 15

If one were using immoralities as a guide to causation, one would conclude that X and Y were common causes of Z. As Freedman and Humphreys point out in [28], commenting in SGS on a DAG produced from a sociological data set,

> The graph says, for instance, that race and religion cause region of residence.

In the context, this illustrates very well that inferring causality from directed arrows leads to non-sensical conclusions and raises a timely note of caution when inferring causality.

7.3. Summarising causal discovery. Freedman and Humphreys go on to summarise the attempts to automate 'causal discovery' with the example of smoking and lung cancer,

> The epidemiologists discovered an important truth - smoking is bad for you. The epidemiologists made this discovery by looking at the data and using their brains, two skills that are not readily automated. .... The examples in SGS (1993) [85] count against the automation principle, not for it.'

The conclusion drawn by the authors of this article is that the output produced by structure learning algorithms provides invaluable information. It can give good information about associations and can certainly point towards the possibility of causal relations, but they do not even begin to automate the process of learning causality; it is still necessary for researchers to use their brains to design experiments, examine the data and use their brains again, taking into account circumstances and contexts additional to the raw data, to reach conclusions. As the example from SGS, extended by Freedman and Humphreys [28] shows, causation cannot be deduced from the presence of an immorality and, indeed, cannot be inferred from the output of structure learning algorithms alone.

8. Conclusion. The subject of Bayesian networks arguably has its origins in the short article from 1853 by A. Cayley [8]. Cayley presents a *causal* network and exploits the factorisation of a joint probability distribution into its component parts based on causal principles. Cayley introduces the idea of the 'or' gate and points towards the use of techniques of algebraic geometry in Bayesian networks.

Graphical models provide an interaction between probability and graph theory, where separation statements in a graph imply independence statements in a probability distribution, so that certain independence statements can be established simply from d-separation statements in the corresponding DAG. Human intelligence finds in graphs a very natural and lucid (or graphic!) way of expressing connections between variables.

Bayesian networks are a versatile tool of artificial intelligence, as any artificial intelligence in real life must be able to reason probabilistically, in order to cope with uncertainty. They have a wide range of applications; for example, reliability theory, system security and in bioinformatics, where Bayesian network structure learning techniques are used to locate genome pathways. A potentially important field of applications is expert systems in medical diagnostics. Bayesian networks are useful in machine learning, i.e. supervised classification when the 'naïve Bayes' assumption is dropped.

If there are causal relations between variables, then the DAG provides a natural tool for representing a natural factorisation of the probability distribution, which has been exploited by Pearl to develop the so-called *intervention* calculus. This provides a natural language for describing controlled experiments.

One topic where further development is necessary is that of structure learning, learning the independence structure from data. The discussion of constraint based versus search and score algorithms indicates that, while constraint based algorithms have a substantial advantage in terms of computational speed, some of the assumptions on which these algorithms are based mean that there are certain important classes of association that the algorithms simply cannot detect. An important task is to relax these assumptions without increasing the computational complexity too drastically.

The problem of learning from data a graph that expresses the independence structure continues to be a major challenge. The number of possible graphs increases super-exponentially in the number of nodes, so that it is not possible to examine all structures available. There are two main philosophies behind structure learning techniques; search-and-score and constraint based. There are also hybrid algorithms, which use ideas from both. Search-and-score tend to employ a stochastic process to run through the space of structures, scoring each structure and then choosing the structure visited that has the highest score. Constraint based algorithms set up constraints through tests for conditional independence. If conditional independence is rejected, the corresponding *d*-connection statement should be present in the output graph; the output tries to give the graph with fewest edges that respects all the connection statements that have been established.

While constraint based algorithms tend to be overwhelmingly faster and less computationally demanding, they tend to have restrictive assumptions. Those that rely on faithfulness tend to perform rather poorly when the underlying distribution does not satisfy this assumption. This happens, for example, if there are hidden common causes. The algorithms also fail to detect situations where there are interaction effects without main effects; X_1 by itself gives no information about X_3 and X_2 by itself gives no information about X_3, but $\{X_1, X_2\}$ together give full information about X_3.

The main challenge in structure learning is to develop algorithms that have the computational efficiency of the constraint based algorithms, while relaxing assumptions such as faithfulness, or composition, for the underlying distribution.

Closely connected with the assumption of faithfulness is the idea of causation. Constraint based algorithms assuming faithfuless will work well in situations where there are cause - to - effect relations between the variables. Learning causality from associations, though, seems premature without a substantial number of additional assumptions on the nature of the data available.

References

[1] Akaike, H. (1974) *A new look at the statistical model identification* IEEE Transactions on Automatic Control vol 19 no 6 pp 716–723.

[2] Barros, B. (2012) *Incremental Learning Algorithms for Financial Data Modelling* Master Thesis, Linköping University, Department of Mathematics LiTH-MAT-INT-A–2012/01–SE (supervisor: J.M. Noble)

[3] Boutilier, C.; Friedman, N.; Godszmidt, M.; Koller, D. (1996) *Context specific independence in Bayesian networks* Proceedings of the Twelfth International Conference on Uncertainty in Artificial Intelligence pp. 115 – 123 Morgan Kaufmann Publishers Inc.

[4] Bromberg, F.; Margaritis, D. (2009) *Improving the reliability of causal discovery from small data sets using argumentation* Journal of Machine Learning Research vol. 10 pp. 301 - 340

[5] Bulashevska, S.; Eils, R. (2005) *Inferring genetic regulatory logic from expression data* Bioinformatics vol 21 no 11 pp 2706 - 2713

[6] Capitanio, A.; Azzalini, A.; Stanghellini, E. (2003) *Graphical models for skew-normal variates* Scandinavian Journal of Statistics vol 30 pp 129 - 144

[7] Castelo, R.; Siebes, A. (2000) *Priors on network structures. Biasing the search for Bayesian networks* International Journal of Approximate Reasoning vol 24 pp 39 - 57

[8] Cayley, A. (1853) *Note on a Question in the Theory of Probabilities* The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science vol. VI. - fourth series July - December, 1853, Taylor and Francis. p. 259

[9] Cayley, A. (1854) *On the theory of groups as depending on the symbolic equation* $\theta^n = 1$ Phil. Mag. vol. 7 no. 4 pp 40 - 47

[10] Cayley, A. (1858) *A Memoir on the Theory of Matrices* Phil. Trans. of the Royal Soc. of London, vol 148 p. 24

[11] Cayley, A. (1869) *A Memoir on Cubic Surfaces* Philosophical Transactions of the Royal Society of London (The Royal Society) vol 159 pp 231–326

[12] Cayley, A. (1878) *Desiderata and suggestions: No. 2. The Theory of groups: graphical representation* Amer. J. Math. vol. 1 no. 2, 174–176

[13] Cayley, A. (1889) *A Theorem on Trees* Quarterly Journal of Mathematics vol 23, pp 276–378

[14] Cheng, J.; Greiner, R.; Kelly, J.; Bell, D. A.; Liu, W. (2002) *Learning Bayesian networks from data: An information-theory based approach* Artificial Intelligence vol 137 pp 43 - 90.

[15] Chickering, D. M. (2002) *Optimal structure identification with greedy search* Journal of Machine Learning Research, 507–554.

[16] Chow, C.K.; Liu, C.N. (1968) *Approximating Discrete Probability Distributions with Dependence Trees* IEEE Transactions on Information Theory, vol. IT - 14 no. 3

[17] Cooper, G.F.; Herskovitz, E. (1992)*A Bayesian Method for the Induction of Probabilistic Networks from Data* Machine Learning vol. 9 pp. 309 - 347

[18] Corander, J.; Ekdahl, M.; Koski, T. (2008)*Parallel interacting McMC for learning topologies of graphical models* Data mining and knowledge discovery vol 17 no. 3 pp 431-456 Springer

[19] Corander, J.; Gyllenberg; M.; Koski, T. (2010) *Learning Genetic Population Structures Using Minimization of Stochastic Complexity* Entropy vol 12 no 5 pp 1102 - 1124.

[20] Corander, J; Koski, T; Nyman, H.; Pensar, J. (2012) *Stratified Graphical Models* (submitted)

[21] Dawid, A.P.; Mortera, J.; Pascali, V.L.; Van Boxel, D. (2002) *Probabilistic expert systems for forensic inference from genetic markers* Scandinavian Journal of Statistics vol 29 no 4 pp 577 - 595

[22] Dean, T.; Kanazawa, K. (1989) *A model for reasoning about persistence and causation* Artificial Intelligence vol 93 no 1–2 pp 1–27

[23] Drton, M.; Sturmfels, B.; Sullivant, S. (2009) *Lectures on algebraic statistics* Birkhüser

[24] Durbin, A.K.R.; Eddy, S.; Mitchison, G. (1998) *Biological Sequence Analysis: Probabilistic Models of Protein and Nucleic Acids* Cambridge University Press

[25] Ekdahl, M.; Koski, T. (2006) *Bounds for the Loss in Probability of Correct Classification Under Model Based Approximation* Journal of Machine Learning Research vol 7 pp 2449 - 2480

[26] Fast, A. (2010) *Learning the structure of Bayesian networks with constraint satisfaction* Ph.D. thesis, Graduate School of the University of Massachusetts Amherst, Department of Computer Science

[27] Flores, M.J.; Nicholson, A.E.; Brunskill, A.; Korb, K.B., Mascaro, S. (2011) *Incorporating expert knowledge when learning Bayesian network structure: A medical case study* Artificial Intelligence in Medicine vol 53 pp 181 - 204

[28] Freedman, D.; Humphreys P. (1999) *Are there algorithms that discover causal structure?* Synthese vol 121 pp 29 - 54

[29] Friedman, N.; Nachman, I.; Pe'er, D. (1999) *Learning Bayesian network structure from massive datasets: the 'sparse candidate' algorithm* Proc. Sixteenth Conference on Uncertainty in Artificial Intelligence (UAI '99) pp 196 - 205

[30] Friedman, N.; Linial, M.; Nachman, I.; Pe'er, D. (2000) *Using Bayesian Networks to Analyse Expression Data* Journal of Computational Biology **7** no 3/4 pp 601 - 620

[31] Friedman, N. (2004) *Inferring Cellular Networks Using Probabilistic Graphical Models* Science Vol 303 no 5659 pp 799-805, DOI: 10.1126/science.1094068

[32] Gamerman, D.; Lopes, H.F. (2006) *Markov chain Monte Carlo: stochastic simulation for Bayesian inference* Chapman and Hall CRC

[33] Garcia, L.D.; Stillman, M.; Sturmfels, B. (2005) *Algebraic geometry of Bayesian networks* Journal of Symbolic Computation **39** pp 331–355

[34] Greenland, S.; Pearl, J.; Robins, J.M. (1999) *Causal diagrams for epidemiologic research* Epidemiology pp 37 - 48

[35] Grim, J. (1984) *On structural approximating multivariate discrete probability distributions* Kybernetika vol 20, pp 1 - 17.

[36] Hajek, P.; Havranek, T.; Jirousek, R. (1992) *Processing uncertain information in expert systems* CRC press

[37] Heckerman, D.; Geiger, D.; Chickering, D.M. (1995) *Learning Bayesian Networks: The Combination of Knowledge and Statistical Data* Machine Learning vol. 20 pp. 197 - 243

[38] Humphreys, P.; Freedman, D. (1996)*The grand leap* British Journal for the Philosophy of Science vol 47 pp 113 - 123

[39] Jaynes, E.T. (2003) *Probability Theory. The Logic of Science* Cambridge University Press

[40] Kiiveri, H.; Speed, T.P.; Carlin, J.B. (1984) *Recursive Causal Models* J. Austral. Math. Soc. (series A) vol. 36 pp. 30 - 52

[41] Kellerer, H. G. (1964) *Verteilungsfunktionen mit gegebenen Marginalverteilungen* Zeitschrift für Wahrschenlichkeitstheorie und Verwandte Gebiete vol 3 pp 247 - 270

[42] Kellerer, H. G. (1991) *Indecomposable marginal problems* in: Advances in probability distributions with given marginals: beyond the copulas, Springer Verlag, Berlin, pp 139 - 149

[43] Kłopotek, M.A. (2002) *A New Bayesian Tree Learning Method with Reduced Time and Space Complexity* Fundamenta Informaticae vol. 49 pp 349 - 367

[44] Kłopotek, M.A. (2003)*Reasoning and Learning in Extended Structured Bayesian Networks* Fundamentae Informaticae vol. 58 pp 105 - 137

[45] Koivisto, M.; Sood, K. (2004) *Exact Bayesian Structure Discovery in Bayesian Networks* Journal of Machine Learning Research vol. 5 pp. 549 - 573

[46] Koller, D.; Friedman, N. (2009) *Probabilistic graphical models: principles and techniques* The MIT Press

[47] Korb, K.B.; Wallace, C.S. (1997)*In Search of the Philosopher's Stone: Remarks on Humphreys and Freedman's Critique of Causal Discovery* British Journal for the Philosophy of Science vol 48 pp 543 - 553.

[48] Koski, T.; Noble, J.M. (2009) *Bayesian Networks: An Introduction* Wiley

[49] Langseth, H.; Portinale, L. (2007) *Bayesian networks in reliability* Reliability Engineering and System Safety vol 92 pp 92 - 108

[50] Lauritzen, S.L.; Spiegelhalter, D.J. (1988)*Local Computations of Probabilities on Graphical Structures and their Applications to Expert Systems* Journal of the Royal Statistical Society B (Methodological) vol. 50 no. 2 pp. 157 - 224

[51] Lauritzen, S.L. (1992)*Propagation of Probabilities, Means and Variances in Mixed Graphical Association Models* Journal of the American Statistical Association vol. 78 no. 420 pp. 1098 - 1108

[52] Lewis, P.M. II (1959) *Approximating Probability Distributions to Reduce Storage Requirements* Information and Control vol 2 no 3 pp 214 - 225

[53] Lindley, D. (2002) *Seeing and Doing: The Concept of Causation* International Statistical Review / Revue International de Statistique vol. 70 pp 191 - 197

[54] Madigan, D.; Andersson, S.A.; Perlman, M.D.; Volinsky, C.T. (1996) *Bayesian Model Averaging and Model Selection for Markov Equivalence Classes of Acyclic Digraphs* Communications In Statistics: Theory and Methods vol. 25, no. 11 pp. 2493-2519

[55] Madigan. D.; York, J. (1995) *Bayesian Graphical Models for Discrete Data* International Statistical Review vol. 63 pp. 215 - 232

[56] Malvestuto, F.M. (1988) *Existence of extensions and product extensions for discrete probability distributions* Discrete Mathematics vol 69 pp 61 - 77

[57] Markowetz, F.; Spang, R. (2007)*Inferring Cellular Networks - A Review* BMC bioinformatics vol. 8 (Suppl 6) : S5

[58] Meek, C. (1995) *Causal inference and causal explanation with background knowledge* Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence pp 403 - 410

[59] Moore, A.; Wong, W-K. (2003) *Optimal Reinsertion: A new search operator for accelerated and more accurate Bayesian network structure learning* Proceedings of the Twentieth International Conference on Machine Learning (ICML - 2003), Washington DC

[60] Murphy, K.P. (2002)*Dynamic Bayesian Networks: Representation, Inference and Learning* Ph.D. dissertation, computer science, University of California, Berkeley.

[61] Neapolitan, R.E. (2004) *Learning Bayesian Networks* Pearson Prentice Hall, Upper Saddle River, New Jersey.

[62] Nelson, E. (1987) *Radically Elementary Probability Theory* Princeton University Press

[63] Pearl, J. (1982) *Reverend Bayes on Inference Engines: A Distributed Hierarchical Approach* AAAI - 82 Proceedings pp. 133 - 136

[64] Pearl, J. (1985)*A model of activated memory for evidential reasoning* in *Proceedings of the Cognitive Science Society (1985)* pp 329 - 334

[65] Pearl, J. (1987) *Evidential Reasoning Using Stochastic Simulation of Causal Models* Artificial Intelligence, vol. 32, pp. 245-257.

[66] Pearl, J. (1990) *Probabilistic Reasoning in Intelligent Systems* 2nd revised printing, Morgan and Kaufman Publishers Inc., San Francisco

[67] Pearl, J. (1995)*Causal Diagrams for Empirical Research* Biometrika vol. 82 pp. 669 - 710

[68] Pearl, J. (1995)*Causal Inference from Indirect Experiments* Artificial Intelligence in Medicine vol. 7 pp. 561 - 582

[69] Pearl, J. (2000) *Causality: Models, Reasoning and Inference* Cambridge University Press

[70] Pearl, J.; Geiger, D.; Verma, T. (1989) *Conditional Independence and its Representations*, Kybernetica vol. 25 no. 2 pp. 33 - 44

[71] Pearl, J.; Verma, T. (1987) *The Logic of Representing Dependencies by Directed Acyclic Graphs* Proceedings of the AAAI, Seattle, Washington pp. 374 - 379

[72] Pearl, J. (1988) *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference* Morgan Kaufmann, San Mateo, CA.

[73] Pearl, J. (1995) *Causal diagrams for empirical research* Biometrika vol 82 pp 669–710

[74] Pistone, G.; Riccomagno, E.; Wynn, H. (2001) *Algebraic Statistics: Computational Commutative Algebra in Statistics* Chapman and Hall, Boca Raton.

[75] Pourret, O.; Nam, P.; Naïm, P; Marcot, B. (2008)*Bayesian networks: a practical guide to applications* Wiley

[76] Ramoni, M.; Sebastiani, P. (1997) *Parameter Estimation in Bayesian Networks from Incomplete Databases* Knowledge Media Institute, KMI-TR-57

[77] Rissanen, J. (1978) *Modelling By Shortest Data Description* Automatica, vol 14 pp 465-471

[78] Robins, J.M.; Scheines, R.; Spirtes, P.; Wasserman, L. (2003) *Uniform consistency in causal inference* Biometrika vol 90 no 3 pp 491- 515

[79] Robinson, R.W. (1977)*Counting Unlabelled Acyclic Digraphs* Springer Lecture Notes in Mathematics: Combinatorial Mathematics V, C.H.C. Little (ed.) pp. 28 - 43.

[80] Sadeghi, K.; Lauritzen, S, (2012) *Markov Properties for Mixed Graphs* submitted to Bernoulli, available on arxiv
arxiv.1109.5909v2

[81] Schwartz, G.E. (1978) *Estimating the dimension of a model* Annals of Statistics vol 6 no 2 pp 461–464.

[82] Schmidt, M.; Niculescu-Mizil, A.; Murphy, K. (2007) *Learning graphical model structure using l1-regularization paths* Proceedings of the National Conference on Artificial Intelligence vol 22 no 2 pp 12– 78

[83] Shafer, G. (1996) *Probabilistic Expert Systems* Society for Industrial Mathematics

[84] Spiegelhalter, D.J.; Dawid, A.P.; Lauritzen, S.L.; Cowell, R.G. (1993) *Bayesian analysis in expert systems* Statistical Science pp 219 - 247

[85] Spirtes, P.; Glymour, C.; Scheines, R. (1993) *Causation, Prediction and Search* Lecture Notes in Statistics no. 81 Springer-Verlag New York

[86] Spirtes, P.; Glymour, C.; Scheines, R. (1997) *Reply to Humphrey's and Freedman's Review of Causation, Prediction, and Search* British Journal for the Philosophy of Science vol 48 pp 555 - 568.

[87] Spirtes, P.; Glymour. C.; Scheines, R. (2000) *Causation, Prediction and Search* second edition, The MIT press.

[88] Studenỳ, M. (2005) *Probabilistic conditional independence structures* Springer Verlag

[89] Sturmfels, B. (2002) *Solving Systems of Polynomial Equations* In: CBMS Lectures Series, American Mathematical Society.

[90] Tsamardinos, I.; Brown, L.E.; Aliferis, C.F. (2006) *The Max - Min Hill - Climbing Bayesian Network Structure Learning Algorithm* Machine Learning vol. 65 pp. 31 - 78

[91] VanderWeele, T.J.; Robins, J.M. (2007) *Directed Acyclic Graphs, Sufficient Causes, and the Properties of Conditioning on a Common Effect* American Journal of Epidemiology, vol 166 no 9 pp 1096 - 1104

[92] Wiberg, N. (1996) *Codes and Decoding on General Graphs* Linköping Studies in Science and Technology. Dissertation 440 Linköpings Universitet, Linköping

[93] Wright, S. (1921)*Correlation and Causation* Journal of Agricultural Research vol. 20 pp. 557 - 585

[94] Wright, S. (1934) *The method of path coefficients* Ann. Math. Statist. vol 5 pp 161 - 215.

[95] Xie, X.; Geng, Z. (2008)*A recursive method for structural learning of directed acyclic graphs* Journal of machine learning research vol. 9 pp. 459 - 483

[96] Yehezkel, R.; Lerner, B. (2009)*Bayesian network structure learning by recursive autonomy identification* Journal of Machine Learning Research vol. 10 pp 1527 - 1570

[97] Zhang, J; Spirtes, P. (2002)*Strong faithfulness and uniform consistency in causal inference* Proceedings of the nineteenth conference on uncertainty in artificial intelligence pp 632–639, Morgan Kaufmann Publishers Inc.

[98] Zhang, S.; Song, S. (2011)*A Novel Attack Graph Posterior Inference Model Based on Bayesian Networks* Journal of Information Security vol 2 pp 8 - 27

Sieci bayesowskie i ropoznawanie zależności strukturalnych

Streszczenie. Artykuł jest przeglądem problemów analizowanych przy pomocy sieci bayesowskich. Sieć bayesowska jest acyklicznym grafem skierowanym, w którym węzły oznaczają zmienne, a krawędzie prawdopodobieństwa warunkowe czyli wpływy jednych zmiennych na inne. Autor przedstawia zależność między *d*-separowalnością a niezależnością. Znaczna część pracy poświęcona jest dyskusji idei zawartych w pracy Arthura Cayley'a [8], która zawiera szereg pojęć i pomysłów wykorzystywanych w teorii sieci bayesowskich takich jak faktoryzacja rozkładu, zaszumione bramki „LUB” oraz zastosowanie geometrii algebraicznej. Autor omawia również „calculus of intervention”, pomysł pochodzący od Pearla, gdy acykliczny graf skierowany (DAG) przedstawia przyczynowo-skutkową strukturę zależności, oraz związki pomiędzy pracami Cayley'a i Pearla.

Większość zawartego w artykule materiału poświęcona jest rozpoznawaniu i wykrywaniu zależności między zmiennymi w oparciu o dwie główne metodologie: przeszukiwania i klasyfikacji oraz realizacji ograniczeń. Algorytmy oparte na kontroli ograniczeń często opierają się na założeniu, że dane do których algorytm jest stosowany pochodzą z rozkładu spełniającego założenie *wierności* oznaczającego równoważność *d*-separowalności i niezależności. W pracy prezentowane są rozważania dla algorytmów opartych na realizacji ograniczeń w przypadkach gdy założenie *wierności* nie jest spełnione. Przeprowadzono krótką dyskusję kontrowersji związanych z wykrywaniem przypadkowych powiązań.

Słowa kluczowe: sieci bayesowskie, graf acykliczny skierowany, wyznaczanie przyczyn, ocena przyczyny, ocena znaczenia czynnika, graficzny model Markowa, drzewa markowskie, rownoważność Markowa, uczenie maszynowe, systemy uczące się, sztuczna inteligencja, systemy eksperckie, wyszukiwarki internetowe, wykrywanie zależności (structure learning), rozpoznawanie struktury (structure learning).

Timo J. T. Koski Timo Koski got a Ph.D in applied mathematics at Åbo Academy University in Turku, Finland for a thesis studying stochastic differential equations in Hilbert spaces. He moved to Luleå University of Technology in Sweden 1986, where worked on stochastic processes in signal analysis. In 1995 he moved to the statistics and probability group at KTH, Royal Institute of Technology in Stockholm, where his research dealt with statistical classification techniques in bioinformatics. In 2000 he was appointed Professor of Mathematical Statistics at Linköping University in Sweden, where he started collaborating with John Noble and started working on Bayesian networks (as an outgrowth of the classification studies). In 2007 he was appointed Professor of Mathematical Statistics at KTH. He has been visitor University of North Carolina at Chapel Hill. Louisiana State University in Baton Rouge, Institute of Information Theory and Automation of the Academy of Sciences of the Czech Republic and University of Helsinki.

John M. Noble was born in Aberdeen, Scotland, in 1966, but left Scotland in 1988. He obtained his Ph.D. from University of California, Irvine in 1992, where he studied stochastic partial differential equations. In 2000, he joined the mathematical statistics group at the University of Linköping, Sweden, where he started taking an interest in Bayesian networks and began collaboration with Timo Koski. In 2012, he left Linköping and joined the Institute of Applied Mathematics, University of Warsaw. His research interests include stochastic partial differential equations, structure learning for graphical models and generalised diffusions and their applications.

Timo J. T. Koski
KTH Royal Institute of Technology
Department of Mathematics, KTH Royal Institute of Technology, SE - 100 44 Stockholm, Sweden
E-mail: `tjtkoski@kth.se`

John M. Noble
University of Warsaw
Institute of Applied Mathematics, University of Warsaw, ul. Banacha 2, 02-097 Warszawa, Poland
E-mail: `noble@mimuw.edu.pl`

(Received: 15th of May 2012)