Learning Causal Networks from Data

Ramon Sangüesa i Solé

Report LSI-96–19–R
Learning causal networks from data

Ramon Sangüesa i Solé

Departament de Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya
c/ Pau Gargallo, 5
28028 Barcelona
Spain

phone: 34-3-401 79 88
e-mail: sanguesa@lsi.upc.es
ABSTRACT

Causal concepts play a crucial role in many reasoning tasks. Organized as a model revealing the causal structure of a domain, they can guide inference through relevant knowledge. This is a specially difficult knowledge to acquire, so some methods for automating the induction of causal models from data have been put forth. Here we review those that have a DAG (Directed Acyclic Graph) representation. Most work has been done on the problem of recovering belief nets from data but some extensions are appearing that claim to exhibit a true causal semantics. We'll review the analogies between belief networks and "true" causal networks and to what extent methods for learning belief networks can be used in learning causal representations.

1. Introduction

Reasoning in terms of cause and effect is an strategy that arises in many tasks. For example, diagnosis is usually defined as the task of finding the causes (illnesses) from the observed effects (symptoms). Similarly, prediction can be understood as the description of a future plausible situation where observed effects will be in accordance with the known causal structure of the phenomenon being studied. Causal models are a summary of the knowledge about a phenomenon expressed in terms of causation. Many areas of the applied sciences (econometry, biomedics, engineering, etc.) have used such a term to refer to models that yield explanations, allow for prediction and facilitate planning and decision making.

Causal reasoning, can be viewed as inference guided by a causation theory. That kind of inference can be further specialized into inductive, deductive or abductive causal reasoning. Inductive causal reasoning aims at building a causal model of the phenomenon being observed, it is widely used in statistics, econometry and the biomedical sciences. Deductive causal reasoning provides causal explanations given a causal model and a description (data) of the phenomena that has to be explained. Prediction too, could be seen as a kind of deduction from a given model and a present known situation in order to reach a future situation causally consistent with what is presently known. Abductive causal reasoning amounts to reasoning with a causal model in order to find the possible causes of a given phenomena, the causal model being known. This could be a crude approximation to diagnosis.

Causal concepts, are in fact central to accepting explanations, predictions, etc. as plausible. It has been argued that causation is a basical concept in commonsense reasoning, as fundamental as time or space. We will not discuss here such a claim because our aim is a more modest one: describing and evaluating several methods for building causal models through the recovery of causal schemas from data. We will also make some claims about how to build such models and discuss our work in that direction.

Causal models have being seen as meta-models by advocates of second-generation expert systems [Steels 1985]. The importance of causal models seems to lie in that they allow for a focusing of inference on the concepts or phenomena that are really relevant to the case, this is why having a causal model aids in guiding inference and gives a higher level schema of the reasoning task for the domain at hand. Usually such higher level schemas are given by experience but they are difficult to build. Consequently much effort has been devoted to devise methods for automatically building causal models.
2. Causation and the discovery process

For the purpose of this presentation, causal discovery is equated to a learning process. This identification is not wholly agreed within the Knowledge Discovery in Database and Data Mining communities, and there are some discrepant views on such identification [Kodorattoff 1995]. In any case, the level of development of the current methods doesn't allow for more sophisticated approaches. This particular learning process takes as point of departure data about the phenomenon being studied, a causation theory, and a learning method.

- **Data** about the objects of interest involved in the phenomenon whose causal structure is to be discovered. Different kinds of objects can be engaged in different causal relations: events, episodes, processes, states, etc. Data are just the syntactical description of those objects; data can be subjective (i.e. a summarization of expert's opinion) or objective (coming from data files or measurement records)

- The **causation theory** is a description of the conditions that should be met by objects represented by data in order to state that a causal relation exists between the objects the data come from;

- Taking the causation theory as background knowledge or as bias [Ugoff ], the learning method identifies potential causal relations that form the basis of the model being built. We'll view the learning process as a search procedure [Mitchell 1981] and classify different learning methods in terms of the heuristics and evaluation functions used and the number of models that give as a result. Information about the complexity of the methods will also be considered.

- The result of the learning process is a **causal model** of the phenomenon under study. Such model is built by composing the previously identified causal relations. The causal model can be seen as a theory of the phenomenon being modeled. This theory can later be used deductively or abductively to fulfill predictive or explanatory tasks. The kind of tasks that can be performed with the resulting model depends on the properties of the causation theory used as background knowledge during the learning process. This implies that, although some causation theories are more general than others, no one is completely adequate for all reasoning tasks for all domains, so when choosing a discovery method it will be important to ascertain first what kind of causation theory is more adequate to guide it [Simmons 1992].

We'll review discovery methods taking the preceding aspects as discriminating criteria. This will allow us to answer the following questions:

- What kind of phenomena can be interpreted by the method? That amounts to he question of what causal relations can be identified with that causation theory which will give an idea of the area of interest of the method, in the sense of what kind of generic tasks can be used with what kind of objects (engineered devices, general processes, etc.)

- What is the resulting model like? What kind of knowledge representation it uses? As we will see, there is a tendency to favour graphical or mixed models (such as causal networks). This will allow us to discuss what inference methods the model can support and how they are implemented.

- What are the properties of the search method? This will allow us to pinpoint possible improvements for each specific method.
- What are the properties of the data? This will allow us to discriminate how well the discovery methods adapt to data that are not ideal (i.e. missing data, noise...).

In order to review the current discovery methods in the terms just discussed, it is necessary to make some concepts about causation quite precise. To be more specific we'll have to know which are the parameters that distinguish the different proposals about causation, the different causation theories. This is the aim of the next section.

**2.1. Causation theories**

In this section we'll state some important issues for distinguishing theories that try to characterize the causal relation. In doing so, our goal is twofold. Firstly we'll clarify the traits that allow for distinguishing causal associations from other types of association and, secondly, we'll be able to compare how the different causation theories formalize the common concepts underlying causation and so we'll be in a position to ascertain their respective merits.

In the most abstract way, causation is understood as a relation between two phenomena or classes of phenomena: one, the cause, is responsible for the occurrence of the other one, the effect. In a sense, the occurrence of causes 'produces' the occurrence of effects.

In order to classify the different causation theories, it is important to know which are the concepts that form the basis of a causation theory. Causation theories differ in the following aspects [De Sosa 1993], [Sobel 1994b]:

- the way in which causation is considered to be produced (deterministically/non-deterministically). The first consideration implies characterization of causation in terms of logical conditions; the second in terms of valid statistical associations between events that are different from spurious association.

- the agent producing causation: uniquely by the *intervention* of an external agent to the experimenter or because it is a process independent of the experimenter that implies certain kind of regularity in nature (manipulative account of causation/non-manipulative account of causation)

- the way in which causes and effects are distinguished. This is the problem of causal ordering. Usually causes are assumed to precede their effects. So, time can be used in order to establish precedence.

- the acceptance or not of the Principle of the Common Cause [De Sosa 1993]: this is a principle due to Hans Reichenbach stating that bewteen to objects of interest A, B either A causes B or B causes A or some other common causes C, exists that both causes A and B.

In a nutshell a causation theory can be understood as a triplet: \( < P, M, I > \)

- \( P \): a language for describing the phenomena of interest: most often than not this will be variables and constraints on variables
- \( M \): a language for describing valid causal models. This involves criteria for establishing causal ordering and criteria for deciding on valid causal association (probabilistic or otherwise)
- I: rules for inference: how to build correct explanations, correct predictions, correct deductions using the model.

3. Causation in AI

As we have already said, there is a growing interest in causal discovery in AI, in automating the identification of causes and effects. The most fundamental motivation is in guiding inference in accord to the known causal structure of the world.

There are many references to "causal models", "causal association" etc., in the AI literature. Interest in causation arises, for example, in commonsense reasoning [Kuipers 1984] and automated diagnosis. [Doyle 1989], [Davis 1983], [Bylander 1990], [Hudlická 1988]. There are also references in qualitative reasoning and modeling [Forbus 1983]. Posterior developments such as second generation expert systems posit also the use of a causal model of the domain as meta-level for expert systems [Console 1989], [Simmons 1992]. The need for diagnosis appears also in engineered devices, which resulted in the motion of "mythical causality" [DeKleer and Brown 1986] and theories of causal order [Simon and Iwasaki 1986]. Several other attempts at defining the causality principle and causal reasoning have been contributed by other workers related to AI, most notably those dealing with default and nonmonotonic reasoning [Shoham 1991], [Shoham 1988].

All these methods have different semantics for the causality relation. Presently, however, the most agreed upon concept of causation used in AI stems from the work of Judea Pearl in belief networks [Pearl 1988], [Pearl 1995a] that has been taken as a reference for the interpretation of causal relations. The underlying formalism has correlates in Decision Theory and in Planning [Howard 1990]. It can be understood as a hybrid model (involving qualitative and quantitative aspects) of causality inspired from several sources, mainly statistical ideas on causality as correlation but also by ideas about probabilistic causation [Salmon 1993], [Suppes 1970]. In Pearl's formulation, causal order is established atemporally in terms of direction of association; causal association is non-deterministic and the principle of common cause is used; objects of interest are variables and the representation language is mainly graphical.

It is important to remark that this is the research area where most work has been done on learning causal schemas.

Other graphical representations tied with causality and which have some degree of equivalence with Pearl's networks are: statistical association graphs [Pearl 1994c], causal chains [Cohen 1993], Heckerman's modification of influence diagrams [Heckerman 1995] and Spirtes causal schemas [Spirtes 1991, 1994].

Non-graphical representations of causation have also received some attention from the point of view of learning. Let's just mention the work by Pazzani [Pazzani 1990] which centers on the idea of using temporal frame representations to induce causal associations and also the system developped by Pandurang [Pandurang 1994] who is using criteria taken from Simon's and Iwasaki work [Iwasaki 1986] on causal ordering to build a causal model.

4. Graphical representations of causality: Causal networks

The network representation for causality has some precedents in AI. For example, [Peng & Reggia 1987] developed a representation for causal links in
diagnosis domains, the causal abductive network, and gave many algorithms for reasoning with them. Similar work can be found in statistics: causal accounts are the center of a whole area devoted to graphical models in statistics [Buntine 1995], [Whitakker 1990], [Madigan 1993].

However, as we mentioned in the previous section, the network representation that has taken a central point and has been used as a reference are bayesian networks [Pearl 1988], that have strong similarities with other formalisms as influence diagrams [Howard 1990]. Bayesian or belief networks are representations with well-known methods for inference [Pearl 1988], [Spiegelhalter 1990]. Many other graphical formalisms used in statistics have been put into equivalence with them, for instance association graphs [Pearl and Wermuth 1994] and even non-graphical formalisms as Simon's theory on causal ordering [Iwasaki 1986] have been referred to them [Simon 1994]. They are the basic reference for dealing with causality. The reason for that may lie, in my opinion, to their precise semantics. Some critics have stressed the difference between these representations for belief associations and truly causal representations [Sobel 1994a, 1994b]. Consequently, the basic formalism has been extended to account for a manipulative account of causation [Pearl 1993, 1994, 1994b, 1995, 1995b], Heckerman [1995, 1995b].

We will develop a substantial part of our presentation around the concept of bayesian network. This will help us in understanding the problems of inferring causal structures from data which share most of the problems with learning bayesian networks. Later we will be able to assess more precisely the difference with the formalisms we have set ourselves the task to be learned.

4.1 Bayesian networks

In a general sense a Bayesian network can be seen as graphical representation of a joint probability distribution on a set of variables, the domain variables. Per se, this information is not enough to represent causal knowledge. It has to be augmented with several other statements that may be explicitly represented.

These statements are:

- independence statements: they represent that some variables have an influence on the behaviour of other variables in the domain (dependency relation) or that some other ones have no mutual influence

- causal statements: Some [Heckerman 1995], [Pearl 1995] have argued that the previous two requirements are not sufficient for representing wholly the (probabilistic) causation relationships existing among variables and have to be augmented with stronger assumptions.

With this aim other conditions have been put forth in order to establish causal links in accordance with an intervention model [Pearl 1995] or a decision theoretic account of causality [Heckerman 1995]. We'll review them briefly later on.

In a nutshell, given the variables of a problem domain \(U = \{x_1, ..., x_n\}\) a Bayesian Network is a Directed Acyclic Graph (DAG) where nodes represent variables in \(U\) and links stand for direct association between variables, that usually are interpreted as direct causal influences. The strength of association between variables is expressed in terms of conditional probabilities in groups (or clusters) of parent and child nodes in the network. It is important to realize that there exists two different components in a bayesian network: a quantitative one (the conditional probability values
on the links) and a qualitative one (the topology of the DAG). Among the properties of Bayesian networks that are to be remarked are their ability to factorise joint probability distributions and their graphical criteria for establishing independence only by taking into account the topology of a graph (the d-separation criterion). Will discuss them in the following.

For example, let's see what kind of information a simple bayesian network can convey:

```
      Age
       |     
      v     v
      v     v
Occupation  Climate
            /      |
         /        |
        /         |
       v          v
       Disease    Symptoms
```

The corresponding functional decomposition:

\[
p(Age, Occupation, Climate, Disease, Symptoms) = p(Age)p(Occupation)p(Climate)p(Disease|Age, Occupation, Climate) p(Symptoms|Disease)
\]

And some probability tables to specify the strength of association:

| Age < 45 | 0.46 |
| Age ≥ 45 | 0.54 |


<table>
<thead>
<tr>
<th>Symptoms</th>
<th>Disease</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>stomach ulcer</td>
</tr>
<tr>
<td>stomach pain</td>
<td>0.80</td>
</tr>
<tr>
<td>chest pain</td>
<td>0.15</td>
</tr>
<tr>
<td>neither</td>
<td>0.05</td>
</tr>
</tbody>
</table>

In general, given a DAG D and a joint distribution P over a set U \{x_1, ..., x_n\} D represents P if there exists a one-to-one correspondence between the variables in U and the nodes in D such that P can be decomposed recursively as the product:

\[
P(x_1, ..., x_n) = \Pi P(x_i|pa_i)
\]

where pa_i are the direct predecessors (parents or direct causes) of x_i in D.

That means, that each variable x_i is conditionally independent of all its other predecessors \{x_1, ..., x_{i-1}\} \setminus pa_i.

Which can be expressed as conditional independence statements:

\[I(Occupation \mid \varnothing \mid Age)\]
I(Climate ⊥ Age, Occupation)
I(Symptoms ⊥ Disease ⊥ Age, Occupation, Climate)

Each statement of the form I(X|Y|Z) is read as "X is independent of Z, given Y". This expression is an extension of the classical concept of independence of variables where X, Y, Z are interpreted as simple variables with some given values. Note that here I(X|Y|Z) is to be understood as "for all instantiations of all variables in X, Y and Z".

The notion of independence, however, can be defined in such a way as to remove any relationship with probability. Criteria for independence have been proposed for other other uncertainty formalisms [Fonck 1991a, 1991b] as well as in other areas of interest as databases [Studený 1993].

From such studies, a possible axiomatic view of independence relations has been agreed upon. The following axiomatization resumes the desired properties for a relation to qualify as a relation of independence.

1. **Trivial independence**
   I(X|Z|∅)
   A null information modifies in no way the information one already has on X

2. **Symmetry**
   I(X|Z|Y) =⇒ I(Y|Z|X)
   Given a state of knowledge Z, if knowing Y gives no information on the value that X may take, then knowing X will give no information on the value that Y could take.

3. **Decomposition**
   I(X|Z|Y ∪ W) =⇒ I(X|Z|Y)
   If both Y and W are irrelevant for the value of X then each one of them, taken separately, should be taken as irrelevant for the value of X

4. **Weak Union**
   I(X|Z|Y ∪ W) =⇒ I(Y|Z U Y|X)
   When knowing Y, an information taken as irrelevant for X, then this information cannot make any other irrelevant information W become relevant for knowing X

5. **Contraction**
   I(X|Z|Y) & I(X|Z U Y|W) =⇒ I(X|Z|Y U W)
   If W is taken as an irrelevant information for X after knowing irrelevant information Y, then W should be irrelevant for the value of X before knowing Y

6. **Intersection**
   I(X|Z U W|Y) & I(X|Z U Y|W) =⇒ I(X|Z|Y U W)
   If two combined elements of information Y and W are relevant for X then, at least one of them should be relevant for X, when the other one is joined with a previous information Z
Any set of independence assertions about a collection of data that reflects the independence implicit in the data (any Dependency Model of the data) that satisfies axioms 2-5 is called a semi-graphoid. If it also satisfies axiom 6, it is called a graphoid. [Pearl 1985].

The interesting thing about bayesian networks, and in general, of belief networks, is that they can be taken as a representation of a dependency model. If this is so, it is important to know which mappings can be established between the topology of the network and its associated dependency properties. The notion of d-separation is central to that task.

**Definition 1: d-separation** [Pearl 1988]
If X, Y, and Z are three disjoint subsets of nodes in a directed acyclic graph D, then Z is said to d-separate X from Y, iff there is no path from a node in X to anode in Y where the following conditions hold: (1) every node with converging arrows either is or has a descendant in Z and (2) every other node is outside Z. A path satisfying this two conditions is said to be active otherwise it is said to be blocked by Z.

**EXAMPLE** [Pearl 1995]: Given the following bayesian network

![Bayesian Network Diagram](image)

Where X1 takes values in the set {winter, spring, summer, fall} and the other variables are binary-valued. The sets X = {X2} and Y = {X3} are d-separated by Z = {X1}; the path X2->X1->X3 is blocked by X1 which belongs to Z and the path X2->X4<-X3 is blocked because X4 as well as all its descendants lie outside Z. On the other hand, X and Y are not d-separated by Z'={(X1, X5)} because the path X2->X4<-X3 is made active by X5, which is a descendant of X4 and belongs to Z'.

If we assume that behind a collection of data there exists a dependency model, M, than the following definitions express the possible relations between the dependency model M and its graphical representation, the DAG D

**Definition 2 I-map**
A DAG $D$ is said to be an I-map [Pearl 1988] of a Dependency model every d-separation relation in $D$ corresponds to an independency relation in $M$. That is, given $X, Y, Z$ three disjoint sets of nodes in $D$
\[ d-(XIZ|Y)D \Rightarrow I(XIZ|Y)M \]
Example: a trivial example is when $D$ is a complete graph.

**Definition 3** minimal I-map
For a given DAG $G$, that is an I-map for a given dependency model $M$, it is minimal if no other DAG $G'$ with less links than $G$ is an I-map for $M$.

**Definition 4** D-map
A DAG $D$ is a D-map [Pearl 1988] for a Dependency Model $M$ if every independency relation in $M$ has a one-to-one relation with a d-separation relations in $D$. That is, given $X, Y, Z$ three disjoint node sets it happens that
\[ d-(XIZ|Y)D \Leftarrow I(XIZ|Y)M \]
Example: when $D$ is a completely disconnected graph.

**Definition 5** Perfect Map
A DAG is a perfect map of a model $M$ if is an I-map and a D-map of the model $M$.

Given a Dependency Model, there can exist several different graphical representations for the same independency relations in the model. These representations are isomorphic. An typical example is the following one. Knowing that $x$ and $z$ are marginally independent but when $y$ is known both are conditionally independent, the following structures are isomorphic.

\[ x \leftarrow y \leftarrow z = x \rightarrow y \rightarrow z = x \leftarrow y \rightarrow z \]

This property has important implications for learning.

For a DAG to be isomorphical to a Dependency Model, $M$, the following conditions are to be met [Pearl 1985]:

1. **Symmetry**
   \[ I(XIZ|Y)M \Leftrightarrow I(XIYZ)M \]

2. **Composition/Decomposition**
   \[ I(XIZ|Y U W)M \Leftrightarrow I(XIZ|Y)M & I(XIZ|W)M \]

3. **Weak Union**
   \[ I(XIZ|Y U W)M \Leftrightarrow I(XIZ|Y|Y)M \]

4. **Contraction**
   \[ I(XIZ|Y)M & I(XIZ U Y|W)M \Rightarrow I(XIZ|Y U W)M \]

5. **Intersection**
   \[ I(XIZ U W|Y)M & I(XIZ U Y|W)M \Rightarrow I(XIZ|Y U W)M \]

6. **Weak transitivity**
   \[ I(XIZ|Y)M & I(XIZ U W|Y)M \Rightarrow I(XIZ|W)M o I(WIZ|Y)M \]

7. **Cordality**
   \[ I(xly U z |lw)M & I(ylx U wiz)M \Rightarrow I(xly|W)M o I(xlz|W)M \]
Letters in lowercase represent individual variables.

The $d$-separation criterion has been proved to be a necessary and sufficient condition in relation to the set of distributions represented by a given DAG there is a one-to-one correspondence between the set of independencies implied by recursive decomposition of probability distributions and the $d$-separation on a DAG.

4.2. Other approaches within the DAG model

All these properties of independence are common to other formalisms for uncertainty representation. For example Fonck [1992] devised possibilistic networks, a specialization of possibilistic hypergraphs [Dubois 1990]. She gave methods for inference in such structures [Fonck 1992, 1994a, 1994b]. In these networks uncertainty is assumed to be represented by a possibililty distribution [Dubois and Prade 1988]. Fonck later described inference algorithms for such networks. These inference mechanisms are essentially the same ones as the ones proposed by Pearl in his original work. Fonck proved that the $d$-separation criterion is also valid for possibilistic networks. She used a characterization of possibilistic independence in terms of the conjunction operator used.

On the other hand, Kruse and Gebhardt [Gebhardt & Kruse 1994] defined a similar construct based on his characterization of possibility in terms of his "context model" [Kruse 1989].

Analogously, Parsons [Parsons 1992] has proposed a characterization of possibilistic networks that draws on Fonck previous work but refers to qualitative concepts of influence between variables.

These approaches, with the exception of Parsons', stress that independence relations (whatever the underlying uncertainty formalism may be) can be characterized by means of the $d$-separation criterion. This is important, because it gives a level of abstraction above details due to the nature of the uncertainty formalism used.

A further development in the direction of a higher abstraction is Shenoy work on valuation systems [Shenoy 1991] which has been given an operational aspect by Cano and so establishes the conditions for propagating uncertainty values in DAGs [Cano et al 1993].

In any case, these methods represent an advance in the direction of providing inference mechanisms based on uncertainty formalisms other than probability. They are not changes in learning methods but in representation. Moreover, even if there is a clear sense of unity in the way that independence properties carry over to different formalisms thanks to a structural criterion, these characterizations still miss some of the characteristics of causal relations.

If several variables are tied by a causal relation, then they surely also participate in a dependence relation identifiable by the $d$-separation criterion but the inverse assertion is not necessarily true, as Pearl [Pearl and Verma 1991, Pearl 1995] forcibly argued.

Several other assumptions have been introduced in order to derive truly causal networks. As we have stressed and the beginning of this presentation, that is equated to the proposal of several new characterizations of causality, i.e. several new criteria for the identification of causation. The novelty in relation with other criteria previously used for example in statistics [Cohen 1993] or even in AI [Spirtes 1991] where
causality is characterized in terms of constraints on correlations, is that the new formalization are based on an extension of the independence model.

Two interesting departures from the basic belief graphical model are to be remarked. One is Heckeman's characterization of probabilistic causal networks in terms of decision theory [Heckerman 1995] and the other one is Pearl's new account for causality in terms of his probabilistic calculus of intervention [Pearl 1994], [Pearl 1995]. Finally Cooper [Cooper 1995] has put forth conditions to graphically identify certain causal relations in terms of their independency properties.

4.3 Heckerman's decision-based view of causality

The main concept behind this approach is the idea of "unresponsiveness", that allows Heckerman to define the causal relation.

In order to understand it, one has to resort to the formalism of a bayesian network into an influence diagrams. [Howard 1990].

An influence diagram is a representation used to represent decisions and its consequences. Its structure is a DAG where nodes are of different types. Decision nodes represent the possibility to take a decision (i.e selecting an alternative); chance nodes represent variables in the domain that may affect decisions; deterministic nodes are those nodes that are deterministic functions of their parents; utility node is a final node that assesses the utility of adopting the final decision. Arcs that point to chance nodes represent conditional dependence relations. Bayesian networks can be seen as influence diagrams with no decision nodes.

In Heckman's proposal, causality is a relation determined by unresponsiveness. A variable $x$ is said to be unresponsive to a decision $d$ if, no matter which alternative is chosen for decision $d$, the outcome of $x$ is the same.

Heckerman and Shachter [Heckerman & Shachter 1994] state that a set of variables $C$ a causes for $x$ with respect to a set of decisions $D$ if:

1. $x$ does not belong to $C$
2. $C$ is a minimal set of variables such that $x(C)$ is unresponsive to $D$

$x(C)$ is said to be a causal mechanism.
Heckerman and Shachter argue, that their formulation is identical to Pearl’s concept of causation (see below) with the exception that, in their view, Pearl requires mechanisms to be independent, and their proposal allows for dependent mechanisms (see [Heckerman and Shachter 1995] for their argumentation).

4.4 Pearl’s intervention view of causality

Pearl has developed a new interpretation for causal networks based on the idea of intervention. This is in accordance with other interpretations of causality widely accepted among experimental disciplines [Sobel 1994b]

The key idea in Pearl’s new work is in finding structural equivalence to causal influence and in defining how a change in a variables value due to external intervention affects the structure of related probability distributions.

4.4.1 The probabilistic action calculus

The first change in the bayesian network defined by Pearl is the explicit representation of a causal mechanism [Simon 1991]. DAGs where each parent-child subgraph is a deterministic function entitles as a causal DAG. In this case a child $X_i$ with parents $pa_i$ represents a deterministic function:

$$X_i = f_i (pa_i, \varepsilon_i)$$

for $i = 1 \ldots n$, $n$ being the cardinality of the set of domain variables and $pa_i$ the set of parents for variable in a given DAG

$\varepsilon_i \ 1 \leq i \leq n$ are mutually independent disturbances

functions allow for calculating the precise effects of interventions. The simplest of interventions (i.e. an external action) is the setting of a simple variable, that is, forcing a variable, say $X_i$, to take a given value, $x_i$. This atomic intervention (set($X_i = x_i)$) according to Pearl, amounts to isolating $X_i$ from the influence of the previous
functional mechanism \( X_i = f_i(p_{ai}, \epsilon_i) \) and setting it under the influence of a new mechanism that makes \( x_i \) constant while all other mechanisms are left unperturbed. That is, the new corresponding graph is a subgraph of the original one where all arrows entering \( X_i \) are wiped out.

Pearl suggests entering a new variable in the system in order to represent the operation of an external intervention. So a new variable \( F_i \) is created and the following convention is made:

\[
X_i = I(F_i, p_{ai}, \epsilon_i)
\]

where \( I \) is a function defined as:

\[
I(a, b, c) = f_i(a, c) \quad \text{when} \quad b = f_i
\]

in this way the action of any external intervention that may alter is represented by another parent node of \( X_i \). The effect is analysed through bayesian conditionalization.

The effect of an atomic intervention of the type \((\text{set}(X_i = x_i))\) is encoded by adding to the graph a new node \( F_i \) and a link connecting it to \( X_i \). \( F_i \) represents the deterministic function but is treated like a variable that can take values in \{set(\(x_i\)), idle\}\(^1\), \( x_i \) has the same domain as \( X_i \) and "idle" means no intervention.

The new parent set \( p_{ai} \) of \( X_i \) is its previous parent set \( p_{ai} \) and the \( F_i \) node. It fulfills the following condition:

\[
P(x_i|p_{ai}) = P(x_i|p_{ai}) \quad \text{if} \quad F_i = \text{idle}
\]

but

\[
P(x_i|p_{ai}) = 0 \quad \text{if} \quad \text{set}(x'_i) \quad \text{and} \quad x'_i \neq x_i
\]

\[
P(x_i|p_{ai}) = 1 \quad \text{if} \quad \text{set}(x'_i) \quad \text{and} \quad x'_i = x_i
\]

Graphically, then, we have a graph with no external intervention:

![Graph](image)

A the graph corresponding to an external intervention \( F_i \).

---

\(^1\) See [Heckerman et al 1995] for a similar construct on decisions.
So the effect of the intervention set(x'_i) transform the original probability distribution P(x_1, ..., x_n) into the new distribution P(x_1, ..., x_n | F_i = set(x'_i)).

The relation bewteen pre- and post- intervention joint distributions can be expressed, thanks to the decomposability property of bayesian networks, as:

$$P(x_1, ..., x_n | F_i = set(x'_i)) = P(x_1, ..., x_n) / P(x_i | pa_i) = \Pi_{j \neq i} P(x_j | pa_j) \text{ if } x_i = x'_i$$
$$0 \text{ otherwise}$$

Graphically this is equivalent to remove the links between X_i and pa_i and leaving the rest of the network as it was before.

4.4.2 Identifiability of causal effects in observational data

The interest of Pearl's action calculus is that allows for the identifiability of causal effects by means of graphical criteria. The criteria for identification of a causal effect is that upon the execution of an action by external agent in setting a variable (do (x) action) the related probability distributions should be altered. If no alteration appears, then no truly causal effect can be said to have taken place. So, if a change in a given variable has no effect on the other variables linked to it in the DAG reflecting dependence relations, no causal relation can be said to exist among those variables.

The important twist in Pearl's work lies in setting graphical conditions for determining which graphs can be subject to such a test. That is, to say which graphical conditions are to be met by dependency graph in order to test them for causal association. If a DAG does not meet such conditions, one cannot infer causal effects from manipulating it. Consequently, it has to be rejected as a representation of causality in the domain.

Pearl's conditions are the following ones [Galles and Pearl 1995]:

A necessary and sufficient condition for the identifiability of the causal effect of set of variables X on Y is that the DAG G containing X and Y satisfies one of the following conditions:

1. There is no direct path from X to Y in G
2. There is no back-door path from X to Y in G (i.e. there is no link into X)
3. There exists a set of nodes B that blocks all paths from X to Y.
4. There exists sets of nodes $Z_1$ and $Z_2$ such that:
   - no element of $Z_2$ is a descendant of X
   - $Z_2$ blocks every directed path from X to Y in $C_\neg X$
   - $Z_2$ blocks all path in $G_\neg X$ between X and $Z_1$ in $C_\neg X$

where $C_\neg X$ is the graph obtained by deleting from G all arrows pointing to X.

So if a graph G has one of these properties, then, it can have a causal interpretation, in the sense given in the previous section.

4.5 Cooper's partial conditions on causal structures

Cooper [Cooper 1995], has tried to devise some new criteria to derive causality from observational data, that interestingly enough make use of independence criteria. We'll comment briefly on his characterizations.

<table>
<thead>
<tr>
<th>Causal network structure</th>
<th>Identifying relations</th>
</tr>
</thead>
<tbody>
<tr>
<td>w -&gt; x</td>
<td>r1, r2, r3, r4, r5, r6, r7</td>
</tr>
<tr>
<td>w -&gt; x</td>
<td>r3, r4, r7</td>
</tr>
<tr>
<td>w -&gt; x</td>
<td>r2, r5, r7</td>
</tr>
<tr>
<td>w -&gt; x</td>
<td>r4, r5, r6, r7</td>
</tr>
<tr>
<td>h</td>
<td></td>
</tr>
<tr>
<td>y &lt;- z</td>
<td></td>
</tr>
</tbody>
</table>

Cooper takes as a bayesian network as the representation of the causal relations among the variables in a domain. Then he lists a set of several relations that he qualifies as truly causal and he studies what kinds of independence relations are satisfied by such relations. He identified the following seven relations of independence in terms of d-separation as uniquely identified the structures depicted in the above figure.

R1: I(x,y)
R2: I(x,z | y)
R3: I(w, y | x)
R4: I(w, z | x)
R5: I(w, z | y)
R6: I(x,z)
R7: I(w, y)

Relations R1 to R7 are tests of independence in terms of the d-separation criterion. The important result is that he proved that these seven relations are sufficient to distinguish among the four network structures.

5. Learning belief networks

The learning problem for this kind of networks can be stated as follows:

"Given a set of data, infer the corresponding topology for the causal network that may have generated them together with the corresponding uncertainty distribution"
We use here the term "uncertainty distribution" in order to allow for uncertainty formalisms other than probability to be used. However, in reviewing current methods will make use of probabilistic examples.

As we said in section 1 the process of causal discovery, as a learning problem, differs basically in the search procedure and also in the function used to rank tentative resulting models. Basically the problems in any approach to such kind of discovery are centered around the high complexity of learning the topology (structure) of the DAG [Dechter 1992]. Once the topology is known, finding the conditional probability tables is straightforward, although some efforts have been aimed at improving its learning. For instance, Musick [Musick 1994] uses neural networks to learn the conditional probability tables.

Additional problems that some methods are able to tackle with are unmeasured variables [Scheines 1994], missing values [Heckerman 1994] and instrumental unobserved variables [Pearl 1995a].

We could divide the methods by taking those that start with an assumption about the structure and then infer the distribution or conversely, start with an assumption about an uncertainty distribution and try to recover the corresponding structure.

To make things clearer let's see which are the dimensions of learning such structures.

The Search space

Given a data base as simple as the following one [Buntine 1995]

<table>
<thead>
<tr>
<th>Case</th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>t</td>
<td>f</td>
<td>t</td>
</tr>
<tr>
<td>2</td>
<td>t</td>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>3</td>
<td>f</td>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>4</td>
<td>f</td>
<td>t</td>
<td>t</td>
</tr>
</tbody>
</table>

The following structures are possible bayesian networks compatible with such data:

![Diagrams of Bayesian networks](image)

To have an idea of how large the search space can become let's take, for example the first structure, the one with three independent variables. Here three probability distributions have to assessed p(A), p(B) and p(C). All variables being binary, each probability is specified by a single real number in [0,1]. The parameters to be learned
can be defined as $\phi_a$. In order to learn the fourth structure, probabilities $p(a)$, $p(b)$ and $p(c|b)$ are to be assessed, this means a family of parameters $\phi_a$ whose values are in $\mathbb{R}^4$. $p(c|b)$ has to be ascertained for two values $p(c|b=t)$ and $p(c|b=f)$. For any given bayesian network structure, the probability table $p(X/Y)$ for any two variables $X, Y$ is a subset of the real space of $(|X| - 1)|Y|$ dimensions (where $|X|$ is the number of different values for a variables $X$. A completely disconnected network such that $|U| = k$ and every variable is binary (as the upper left one in the figure) needs between $k$ and $2^k - 1$ values to specify its conditional probability tables. In case that continuous variables are used and that they follow a Gaussian distribution a node with $k$ parents will need $k(k+1)/2$ values to specify the mean and covariance matrices.

Fortunately, equivalence properties between bayesian networks have been identified [Verma and Pearl 1990] that allow for a reduction in the number of different networks. There exist networks that represent the same independence statements.

For example, for networks created with just three variables, there are 25 possible networks (varying arrow orientation and connectivity). However, there are only eleven different equivalence classes (where each class contains networks reflecting the same dependency model).

Given that there may be $k(k-1)/2$ undirected arcs in a network of $k$ variables, there are $2^{k(k-1)/2}$ different networks of $k$ binary variables. If a direction on variables is imposed such that an arc can only point to a variable that appears later in the order, then the number of directed networks is $2^{k(k-1)/2}$.

Finding and selecting a network

Over these search space some way has to be found in order to select the "best" network that reflects existing dependence in data.

Much of the methods we will review, make use of standard statistical sampling methods to derive the needed parameters (structure plus probability distribution).

Here, one assumes that a given structure has generated the data, and then some measure of compatibility between such assumption and the probability of obtaining the data has to be devised.

Given a collection of data, during the learning process, different networks may be possible alternatives, even after considering the dependency equivalences that may exist among them. In general, methods that resort to quality measures have derived some form of establishing the overall quality of a networks in terms of its constituents, reducing quality measures to the sum of the quality of all given child-parent configurations, this is due to the property of factorization over distribution which is inherent to bayesian networks.

$$\text{Quality (Network | data)} = \Sigma \text{quality (x | pa}_x\text{, data)}$$

There are other possibilities in choosing different alternative structures. Methods based on conditional independence tests choose a variable to become a new node according to a previous given order and when, still, several variables are eligible they make a random choice.
In general, one can distinguish two great groups of methods. The first ones are based on the application of conditional tests between variables and the construction of the structure of the DAG based on the result of such test; then the conditional probability tables (the quantitative part of the network) are calculated from the data. The second ones are methods based on goodness of fit test between the probability distribution of a tentative DAG and the true joint distribution implied by the data. We'll review them in the same order.

There exists, too, a mixed method, the CB algorithm [Singh and Valtorta 1995] that first derives a structure by means of CI-tests between variables, and then generates an order that is fed to the K2 algorithm. The important thing is that this method allows to reduce the complexity of CI-tests to order one tests.

One of our current proposals is also a hybrid method, but as we will see we exploit the relation between CI tests and goodness of fit in a different way, incrementally as the DAG is built.

5.1 Conditional independence test methods

Algorithms in this class resort to the qualitative properties of the networks in order to build the corresponding belief network. They usually take as input a set of dependence relations among variables or sets of variables in the domain. The output is a belief network that reflects those relationships. Let’s remember, that, given a dependency model several different networks may reflect the same dependencies up to isomorphism. All of these algorithms return a structure that has to be completed in the by calculating the associated conditional independency tables.

Simple structures

The following method was devised by Geiger, Paz and Pearl [Geiger 1990] in order to build singly connected structures.

**The Geiger, Paz and Pearl algorithm (I)**

Input: a list of dependences between the variables in a domain U.
Output: a polytree or an error message
1. Build a complete undirected graph
2. Build the Markov network G₀ erasing every arc x - y such that
   I(x | U \ {x, y}) \ y \ M
3. Build Gₐ by erasing every arc such that I(x | \ \ y) \ M
4. Turn each arc x - y in Gₐ into x -> y if y has a neighbour such that I(x | \ \ z) \ M and x - z does not belong to Gₐ
5. Direct the rest of links without creating new head to head connexions. If this is not possible, return error
6. If the resulting polytree is not an I-map, then return error

The same authors put forth another method that is able to recover polytrees.

**The Geiger, Paz and Pearl algorithm (II)**

Input: a list of dependences between the variables in a domain U.
Output: a polytree or an error message
1. Build a complete undirected graph
2. Erase every arc x - y such that I(x | U \ {x, y}) \ y \ M
3. Erase every arc x - y such that I(x | \ \ y) \ M
4. Turn each arc x - y and y - z into x -> y and y -> whenever x - y - z is in the graph and I(x | \ \ y) \ M
5. Direct the rest of links without creating new head to head connexions. If this is not possible, return error.
6. If the resulting graph does not represent the dependency model well, then return error.

Remarks: a head to head connexion is a structure of the form \( x \rightarrow y \leftarrow z \)
A graph is said to represent a dependency model well whenever if two nodes \( x \) and \( y \) are connected through a path without head to head arcs, than those nodes are marginally dependent, that is \( I(x \parallel \emptyset | y)_M \) does not hold.

**Spirtes, Glymour and Scheines algorithm**

Input: a list of dependences between the variables in a domain \( U \).
Output: a directed graph
1. Build a complete undirected graph \( H \)
2. For every arc \( x, y \) if there exist a subset \( S \in U \setminus \{x,y\} \) such that \( I(x \parallel S | y) \), erase the arc \( x \rightarrow y \)
3. Let \( K \) be the resulting graph of step 2. Then for every triplet \( x \rightarrow y \leftarrow z \) in \( H \), such that \( z \rightarrow x \) is not in \( H \) if no subset \( S \in U \setminus \{x,y\} \) exists such that \( I(x \parallel S | \{y \}| z) \) then create the orientation \( x \rightarrow y \leftarrow z \)
4. Repeat until no more arcs could be oriented
   4.1 If \( x \rightarrow y \leftarrow z \) is in \( H \) with \( x \) and \( y \) being non-adjacent nodes, orient \( y \rightarrow z \)
   4.2 If there exists a directed path from \( x \) to \( y \) and the connexion \( x \rightarrow y \) also exists then orient \( x \rightarrow y \)

Remarks: This algorithm depends heavily on a theorem by Spirtes, Glymour and Scheines [Spirtes 1993] that states the following conditions for a dependency model to be isomorphic to a DAG.
(1) \( x \) and \( y \), nodes in a DAG, they are adjacent if and only if they are conditionally independent given any set of nodes in the DAG no including \( x \) or \( y \).
(2) For every triplet of nodes \( x, y, z \) such that \( x \) and \( y \) are adjacent but \( x \) and \( z \) are not, \( x \rightarrow y \leftarrow z \) is a subgraph of the DAG if and only if \( x \) and \( z \) are conditionally independent given any set that contains \( y \) but not \( x \) nor \( z \).

Step two is critical, because it needs to search among all possible subsets \( i \in U \setminus \{x, y\} \). The time needed to calculate such test is also exponential.
An improvement is given by the same authors. It carries out the least number of comparisons.
\( Ad(x) \) is the set of adjacent nodes for node \( x \).

**The PC algorithm**

Input: a list of dependences between the \( n \) variables in a domain \( U \).
Output: a directed graph
1. Create a complete graph \( G \) on the variables in \( U \)
2. \( n := 0 \)
3. Repeat Until \( |Ad(x)| \setminus \{y\} < n \) for each set of ordered pairs \( (x, y) \)
   3.1 Repeat Until all ordered pairs of adjacent variables \( (x,y) \) such that \( |Ad(x) \setminus y| \geq n \) and every subset \( S \) in \( Ad(x) \setminus y \) have been tested for independence.
      Select an ordered pair of variables \( x, y \) adjacent in \( G \) such that \( |Ad(x) \setminus y| < n \);
Select a subset $S$ of $\text{Ad}(x) \mid y$ with cardinality $n$. 
If $I(x \mid S \mid y)$, then erase $x \to y$ from $G$. Store $S$ in the sets Separating(x, y) and Separating (y, x) 
3.2 $n := n + 1$

4. For each triplet of nodes $x, y, z$ where $x$ and $y$ are adjacent, and $y$ and $z$ are adjacent but $x$ and $z$ are not adjacent, orient $x \to y \leftarrow$ if and only if $y$ does not belong to Separating(x, z)
5. Repeat Until no more arcs could be oriented 
5.1 If the structure $x \rightarrow y \rightarrow z$ belongs to $G$, where $x$ and $z$ are not adjacent and no arcs pointing to from $y$ orient $y \rightarrow z$ as $y \rightarrow z$
5.2 If there exists a directed path from $x$ to $y$ and the arc $x \rightarrow y$ exists, turn it into $x \rightarrow y$

Huete & Campos [Huete 1995]

This algorithm is devised to recover a special case of network, that is a causal polytree. Causal polytrees can be seen as simple DAGs where between any two nodes only a single path exists. The name "polytree" stems from the fact that these structures can be seen as a collection of several causal trees fused together where arrows converge head to head ($\rightarrow x \leftarrow$).

For each node $x$ a set $\Lambda_x$ is defined. $\Lambda_x$ contains the set of variables $y$ belonging to $U$ such that $x$ and $y$ are marginally dependent. In a polytree structure, two variables are dependent if there is no head to head node in the path connecting them. The idea is to take a variable $x$, and test for any other two variables, $y, z$ in $\Lambda_x$ if $y$ lies in the path between $x$ and $z$.

In order to do that, a new concept, the sheaf of a node, is defined. The sheaf structure is just the direct parents and descendants (Huete calls them direct causes and effects) of a given node.

The idea of the algorithm is to create a growing partial structure $T$ and to restrict the search for new variables to include between those in $\Lambda_x$ that can affect the sheaf of $x$. De Campos and Huete [De Campos and Huete 1993] proved that if $x$ is a variable in a dependency model $M$ represented by a node in a structure $T$ and $z$ a variable not in $T$ but belongin to $\Lambda_x$, then sheaf of $x$ has to be modified only if one of the two following conditions holds:

1. $I(x \mid z \mid y)$ is true in $M$ for some $y$ belonging to the sheaf of $x$
2. $I(x \mid y \mid z)$ is false in $M$ for all $y$ belonging to the sheaf of $x$

Moreover, if the previous conditions do not hold, then there exist one and only one node $y$ belonging to the sheaf of $x$ such that $I(x \mid y \mid z)$ is true in $M$.

So, the last property helps in directing the search for the variables whose sheaf structure has to be modified.

The algorithm starts with an empty structure and then selects a variable $x$. Then dependent variables with respect to $x$ are also found ($\Lambda_x$). A structure $T$ is built with variables $x$ and those in $\Lambda_x$. Then repeatedly a new variable $y$ from $T$ is selected his
corresponding set $\Lambda_y$ of marginally dependent variables is found and for any $z$ not in $T$ an attempt is made at inserting it in $T$. This is repeated until all variables are in $T$.

A polytree structure is constructed in $O(n^2)$ steps, $n$ being the number of variables. Only marginal and first order conditional independence tests are needed. However no directionality for arcs is recovered. This was improved in an other version of the algorithm [Huete 1995] by using the test of independence on any three variables $x, y, z$. If, by testing if $x$, and $y$ were marginally independent and became dependent given $z$ then they were oriented as $x \rightarrow z \leftarrow y$. Other algorithms were developed to recover DAGs in polynomial time.

5.2 Goodness of fit methods

Each of the following algorithms is devised with a given structure in mind. Some are only able to recover simple structures, some recover only polytrees and some other ones recover DAG.

The rationale is to assume that a graph exists whose nodes correspond to the variables in a data base. Due to the factorization property of belief networks, it is easy to make an assumption about which is the probability distribution it induces, $P^E$ as the product of the distribution of nodes conditioned to their parents. On the other hand, the data base allows for the estimation of a joint probability distribution over its variables $P^D$. What those methods try to attain is a graph which exhibits the minimum distance between $P^E$ and $P^D$. Difference between methods are centered around the type of graph that they allow to recover, measures used in assessing distance between distributions and the way the graph is built applying such measures.

Normally the Kullback-Kleiber entropy measure is used as a distance between distributions. Distance $D$ between two distributions is defined as:

$$D(P^D, P^E) = \Sigma P^D(x_1, ..., x_n) \log \frac{P^D(x_1, ..., x_n)}{P^E(x_1, ..., x_n)}$$

5.2.1 Singly connected networks

To find the structure from data, the maximum weight generating tree is projected and as the weight for each link the following measure of information between variables is taken:

$$I(x_i, x_j) = \Sigma P^D(x_i, x_j) \log \frac{P^D(x_i, x_j)}{P^D(x_i)P^D(x_j)}$$

for all instantions of $x_i$ and $x_j$.

Note that for such measure is minimum (zero) when both variables are independent.

We'll just give next algorithm for the sake of our presentation, as it is the basis for Rebane and Pearl's algorithm [Rebane and Pearl 1989].

Chow and Liu Tree Recovery algorithm
Input: a data base on sobre $x_1, ..., x_n$ variables
Output: a tree-structure reflecting dependences in the data base.
0. $T = \{ \emptyset \}$ the empty tree
1. Calculate for every pair $(x_i, x_j)$ the bidimensional marginal distribution $P(x_i, x_j)$
2. Calculate weights for links between every pair $(x_i, x_j)$ by means of the $I(x_i, x_j)$ measure
3. Select the maximum cost pair $(x_i, x_j)_M$
   $$T = T \cup \{ x_i, x_j \}$$
4. Select the next maximum cost pair $(x_k, x_l)$
   If it does not create a cycle in $T$, add it to $T$
   Else erase $(x_k, x_l)$ from the set of pairs
5. Repeat 4 until including $n - 1$ links

Remarks:
- It can only recover trees
- For a given distribution it can recover different trees depending on the order in which pairs with the same weight are selected
- No arcs are oriented

Rebane and Pearl algorithm

In contrast with the previous algorithm this one is able to give an orientation to links. In a first step it creates the skeleton of the graph (i.e. the result of Chow and Liu algorithm) and the tries to give an orientation to as many as links as possible. Note that several probabilistic dependence relation are indistinguishable in terms of orientation. Given three variables $x, y, z$ one can test if they can be related by the structure $x \rightarrow y \leftarrow z$ if $x$ is marginally independent of $z$. However $x \leftarrow y \rightarrow z$ cannot be distinguished from $x \rightarrow y \rightarrow z$ or $x \leftarrow y \leftarrow z$.

Rebane and Pearl polytree recover algorithm
Input: a database on variables $x_1, \ldots, x_n$
Output: A partially oriented graph

1. $T =$ maximum weight generating tree resulting from Chow and Liu algorithm
2. Select $x, y, z$ such that $I(x|z)$
   Give $x, y, z$ the orientation $x \rightarrow y \leftarrow z$
3. If a subgraph with more than a parent is found, apply the test of marginal independence to adjacent nodes
4. For each node with at least un incoming arc, study the orientation of the rest of the adjacent nodes by means of the marginal independence test
5. Repeat steps from 2 to 4 until no new orientations can be found
6. If there are still some links with no orientation, label them as "undetermined"

5.2.2 DAG algorithms

Cooper and Kerskovitz Entropy-based method

Cooper and Herskovitz [Herskovitz and Cooper 1990] devised a method which took as a quality criterion a measure of the entropy of the distribution implied by the structure being built.
The Kutató algorithm
Input: a database on variables $x_1, ..., x_n$; a fixed value for entropy $\alpha$; an order between variables;
Output: An oriented DAG or an error code
1. Build a DAG on $x_1, ..., x_n$ assume all variables to be marginally independent
2. Calculate the DAG entropy
3. Select a link such that (1) it creates no cycle (2) is the one that creates a new graph $G'$ with minimum entropy (3) links variables $x$, $y$ such that $x$ comes first in the order
4. Give the orientation $x -> y$
4. Repeat steps 2 to 4 until an $\alpha$ entropy level is reached

The complexity of the method can be estimated as follows: given a DAG with $n$ nodes, in order to select the best one $O(n^2)$ are to be made. If all associations are significant then the process is repeated $O(n^3)$ times. So complexity (entropy calculations aside) is $O(n^4)$.

Entropy for a DAG $G$ is calculated by the local entropy of a node instantiation given its parents is weighted by the probability that the parents have a given value instantiation.

A bayesian based method: the K2 algorithm

Cooper and Herskovitz [Cooper and Herskovitz 1992] devised an improved version of the previous algorithm which resorted to bayesian criteria.

Given a Data Base $D$, with information on $n$ variables, and a bayesian network $B_s$ with $n$ nodes (one for each variables in $D$) one has to find the bayesian network that maximizes the probability $P(B_s | D)$. They approximate $P(B_s | D)$ by $P(B_s, D)$.

Cooper and Herskovitz merit lies in finding a sound way for calculating the whole probability of a DAG in terms of local parent-children subgraphs.

Their measure is:

$$P(B_s | D) = P(B_s) \Pi g(x_i, pa_i)$$

where $pa_i$ are the sets of parents of the variables $x_i$

where $g(x_i, pa_i)$ is:

$$g(x_i, pa_i(x_i)) = \Pi (r_i - 1)! / (N_{ij} + r_i - 1)! N_{ijk}!$$

where, for each variable $x_i$, $r_i$ is the number of possible instantiations; $N$ is the number of cases in the data base; $wij$ is the $j$-th instantiation of $pa_i$ in the data base; $q_i$ is the number of possible instantiations for $pa_i$; $N_{ijk}$ is the number of cases in $D$ for which $x_i$ takes the value $x_{ik}$ with $pa_i$ instantiated to $wij$; $N_{ij}$ is the sum of $N_{ijk}$ for all values of $k$.

A previous assumption is that all structures are equally probable. An order between variables is given. If $x_i$ precedes $x_j$ in that order, all structure where an arc between $x_i$ and $x_j$ are to be removed, further reducing the possible alternatives. A further restriction is that the number of parents a given node can take is low.
The K2 algorithm proceeds by starting with a single node (the first variable in the order) and then takes the node that increments more the probability of the given structure, calculated by means of the $g$ function. When adding a new parent does not increment the probability, no more nodes are added to the parent set.

**The K2 algorithm**

Input: a set of variables $x_1, ..., x_n$; a given order among them; an upper limit $u$ on the number of parents for a node; a data base on $x_1, ..., x_n$

Output: A DAG with oriented arcs.

For $i:= 1$ to $n$ do
1. $pa_i(x_i) = \emptyset$; $Ok:= true$
2. $P_{old} := g(x_i, pa_i(x_i))$;
3. While $Ok$ and $|pa_i(x_i)| < u$ do
   3.1 Let $z$ be the node in the set of predecessors of $x_i$ that does not belong to $pa_i(x_i)$ which maximizes $g(x_i, pa_i(x_i)) \cup \{z\}$;
   3.2 $P_{new} := g(x_i, pa_i(x_i) \cup \{z\})$
   3.3 If $P_{new} > P_{old}$ Then $P_{old} := P_{new}; pa_i(x_i) := pa_i(x_i) \cup \{z\}$
      Else $Ok := false$

Execution time is in the order $O(Nu^2n^2r)$ with $r$ being the maximum value for $r_i$.
There exists an extension for dealing with continuous variables and missing values.

**Using the minimum description length principle**

The idea behind these methods is the use of the Minimum Description Length principle in order as a measure of fit. The idea is that the best representation for a data base is that model that minimizes its description length. That means, the representation that minimizes, given a codign schema, the sum of the length of encoding:
- the model
- the data, given the model

Lam and Bacchus [Lam and Bacchus 1993, 1994a, 1994b] have given coding schemas for networks as binary strings.

**Network codification:**
For each node (variable) a list of its parents is needed together with the list of the conditional probabilities of the variables, given the parents; then for a graph with $n$ nodes the total description length is:

$$\Sigma [|pa_i(x_i)| \log_2(n) + d(r_i - 1) q_i]$$

Where $|pa_i(x_i)|$ is the number of parents of a given node $x_i$; $d$ is the number of bits needed to represent a numerical value; $r_i$ is the number of different values $x_i$ can take and $q_i$ is the number of possible instantiations that the set of parents of $x_i$ can take.

Consequently, more connected networks have longer descriptions.

**Database codification**
Data are encoded by representing all the values that appear in the data base as a single binary string. They use a Huffman code:
- \( N \sum p(x_i) \log_2 p^*(x_i) \)

where \( N \) is the number of cases in the database, \( p(x_i) \) is the probability of occurrence of the atomic event \( x_i \) and \( p^*(x_i) \) is the probability of the success calculated from the network representing the model.

Given that such encoding requires an exponential number, they take advantage of the factorization property of bayesian networks and calculate the following number:

\[
- N \sum H(x_i, pa_i(x_i)) + N \sum -[\sum p(x_i) \log_2 p(x_i)]
\]

where \( H(x_i, pa_i(x_i)) \) is:

\[
H(x_i, pa_i(x_i)) = \sum p(x_i, pa_i(x_i)) \log_2 p(x_i, pa_i(x_i))/p(x_i) \cdot pa_i(x_i)
\]

the summation is over all parents of \( x_i \)

Total description length for a given node is then:

\[
DL_i = lpa_i(x_i) \log_2 n + d(\tau_i - 1) q_i - N \cdot H(x_i, pa_i(x_i))
\]

The total description length of a DAG is calculated by summing \( DL_i \) over all variables.

In order to search the space of possible networks a best-first search method is used.

Separate sets of candidate graphs are maintained. These sets differ in the number of arcs their graphs have. For a database with \( n \) variables DAGs can have between 0 and \( n(n-1)/2 \) arcs so \( n(n-1)/2 + 1 \) separate sets are maintained.

Before starting search mutual information (as in the Chow and Liu algorithm) is calculated for each pair of nodes and links are sorted accordingly.

Within each set a best-first search is performed. Each element of a set has two components: a candidate network with a the number of arcs corresponding to the set and a pair of nodes between which an arc could be added without causing a cycle. The search takes as a heuristic the total description length of the graph.

5.3 Hybrid algorithms

Singh and Valtorta [Singh and Valtorta 1993, 1994] have devised an algorithm that has a two-step procedure. In the first step it performs a series of conditional independence tests and obtain an ordering among variables; then they start a process similar to the K2 algorithm we reviewed before.

The CB algorithm
Input: \( u \), a limit on the number of parents a node may have; a set \( Z \) of \( n \) variables \( x_1, ..., x_n \)
Output: a DAG

26
1. Start with the complete graph $G_1$ on the set of variables $Z$
   
   $\text{ord} := 0$
   
   $\text{oldpa}_i := \emptyset$ for each $i$ in $1 \leq i \leq n$
   
   $\text{oldprob} := 0$

2. Modify $G_1$ as follows:
   
   For each pair of vertices $a$, $b$ that are adjacent in $G_1$, if $A_1$ has a
   cardinality greater than or equal to $\text{ord}$, and $I(a, S_{ab}, b)$ where $S_{ab}$ is
   contained in of cardinality ord, remove the edge $a$-$b$ and store $S_{ab}$.

   If for all pairs of adjacent vertices $a$, $b$ in $G_1$, $\text{Adj}_{ab}$ has a cardinality $<$
   $\text{ord}$, goto step 10

   If degree of $G_1 > u$ then $\text{ord} := \text{ord} + 1$

   Goto the beginning of step 2

3. Let $G$ be a copy of $G_1$
   
   For each pair of non-adjacent variables $a$, $b$ in $G$, if there is a node $c$
   that is not in $S_{ab}$ and is adjacent to both $a$ and $b$, then orient edges as $a -$
   $\rightarrow c$ and $b - \rightarrow c$ unless this creates a cycle.

   If and edge has already been oriented in the reverse direction, make it
   bidirected.

4. Try to assign directions to yet's undirected edges in $G$ by applying the
   following rules:

   R1: if $a \rightarrow b$ and $b \leftarrow c$ and $a$ and $c$ are not adjacent then direct $b \rightarrow c$

   R2: if $a \rightarrow b$ and $b \rightarrow c$ and $a \leftarrow c$ then direct $a \rightarrow c$

   R3: if $a \rightarrow b$, $b \leftarrow c$, $a \leftarrow c$, $c \leftarrow d$ and $d \rightarrow a$ then direct $a \rightarrow b$, $c \leftarrow b$

   Moreover if $a \rightarrow b$, $b \rightarrow c$ and $a \leftarrow c$ then direct $a \rightarrow c$

5. Let $p_{ai} := \emptyset$ for every $i$, $1 \leq i \leq n$
   
   For each node $i$, add to $p_{ai}$, the set of vertices $j$ such that for each
   such $j$ there is an edge $j \rightarrow i$ in the partially directed graph $G$

6. For each undirected or bidirected arc in the partially directed graph $G$,
   choose an orientation as described next:

   If $i \rightarrow j$ is an undirected edge an $p_{ai}$ and $p_{aj}$ are the corresponding parent
   sets in $G$ then calculate the following products

   $i_{val} = g(i, p_{ai}) \times g(j, p_{aj} \cup \{i\})$

   $j_{val} = g(j, p_{aj}) \times g(i, p_{ai} \cup \{j\})$

   where $g$ is the measure defined by Cooper and Herskovitz for K2.

   If $i_{val} > j_{val}$ then $p_{aj} \leftarrow p_{ai} \cup \{i\}$ unless the addition of $i \rightarrow j$ creates a
   cycle. In that case choose the reverse orientation and change $p_{ai}$. Do a
   similar thing if $j_{val} > i_{val}$.

7. The sets $p_{ai}$ obtained in step 6 define a DAG.

   Generate an order by performing a topological sort on it.

8. Apply the K2 algorithm to find the set of parents of each node using the
   order in step 6.

   Let $p_{ai}$ be the set of parents found by K2 for node $i$

   Let $\text{newprob} := \Pi g(i, p_{ai})$

9. If $\text{newprob} > \text{oldprob}$ then

   $\text{oldprob} := \text{newprob}$

   $\text{ord} := \text{ord} + 1$
\textit{oldpai} := \text{pa}_i, \text{ for every } i, 1 \leq i \leq n

Discard G
Goto step 2
Else goto step 10
10. Output \textit{oldpai}, for every \textit{i}, \text{1 \leq i \leq n}
Output \textit{oldprob}

The fact that the authors use a chi-square test for testing dependence at a fixed \( \alpha \) level may induce dependency that are a product of chance. The quality of the network hinges critically on the order extracted in the first phase of the algorithm.

5.4 Algorithms for non-probabilistic formalisms

Gebhardt and Kruse [Gebhardt and Kruse 1995] have developed several algorithms to retrieve possibilistic DAGs.

They base their search methods on a heuristic expressed in terms of non-specificity, the counterpart of entropy in possibility theory. They try to find the network that minimizes the expected non-specificity.

They define non-specificity in terms of the Hartley information measure [Klir 1988].

Given a set \( A \), the Hartley information measure is defined as:

\[ H(A) = \log_2 |A| \]

Then given a DAG \( D \), its total non-specificity is:

\[ \text{Nonspec}(D) = \sum \text{H}(\text{pa}_i) \]

The exact algorithm is not clearly stated by their authors. However it seems to start with a single node and at each step the link with minimum non-specificity is added. A node ordering is explicitly used in selecting the next node to consider.

5.5 Discussion

In general, methods relying on goodness of fit heuristics need previous knowledge in the form of an order between variables. Buntine [Buntine 1991] has devised a method that needs no order but, in exchange, it needs an external expert to specify priors on probability distributions. Heckerman et al. [Heckerman et al 1994] have also devised a method of this kind where some previous knowledge in the form of priors on distributions have to be fed to the algorithm.

Goodness of fit methods tend to give more than one resulting network that can be ranked in terms of its probability. There can be several networks with the same probability given the equivalence properties of belief networks.

Methods based on conditional independence test criteria are more abstract in the sense that they only recover structure, where conditional distributions are to be added in order to get a belief network. Many different formalisms can be used to represent uncertainty. So, in principle, such methods are more abstract than the other ones.
However they depend heavily on the dependence list provided at the beginning and some of them resort to probabilistic notions in order to decide on arc orientation. They also need a substantial amount of data to deliver reliable CI tests.

The need for using different uncertainty formalisms arises in many occasions. For example, when data are scarce or fraught with imprecision. This is the case of data coming from sensors. Probability intervals or possibility distributions are good alternatives to represent the uncertainty in these systems.

These uncertainty calculi are also more robust to incomplete data.

Methods for measuring possibilistic information from data have been developed recently by Josslyn [Josslyn 1995]. Josslyn also defined the term "possibilistic process" and "possibilistic model" in contrast to "stochastic processes" and stressed the need to developed a network representation for it. The connection between these methods and the work by Fonck, Huete, Gebhardt and others is still to be made.

6. Learning "true" causal networks

Recently there has been a move towards devising methods to learn networks that embody some kind of causal semantics. The general proposals of Heckerman and Pearl as well as Cooper's list of relations for characterizing "true" causal relations are an improvement over previous rapid identifications between belief networks and causal networks in the sense that they are semantically more correct. All of them rely, to some extent, on a concept of causation linked to the idea of external intervention. However, both have different implications for learning algorithms.

The construction of a causal network, in Heckerman's work, is equated to the construction of an influence diagram in Howard Canonical Form [Howard 1990]. However, during this process predecessors of every utility nodes are known with certainty by the decision maker and also the structure of the arcs. The sole task that remains, then, is to assess the physical probabilities associated with chance nodes. Moreover, all states of decision nodes are known in advance by the decision maker, so little room is left for a learning method. The problem reduces to learning a bayesian network where decision variables are interpreted as chance variables.

In brief, what Heckerman et al. posit the problem of learning Howard Canonical Form influence diagrams, given that the structure is known. That is, only parameters of the structure are to be learnt.

What they finally remark [Heckerman et al. 1995] is to start learning such parameters given an a priori network given by the decision maker.

In Heckerman's proposal, it is important to start with a clear knowledge about which variables in the data are decision variables and which ones are chance variables. This amounts to knowing beforehand, some structure of the domain. Secondly, there is a need to specify a priori knowledge in the form of priors on distributions. More frequently than not, what experts do know well is the existence of qualitative relations between variables or constraints that they have to reflect. In contrast, experts are very poor estimators of probability distributions [Kahneman and Tversky 1982]. So, it seems to me, that such a method could be very well applied to data coming not form passive observations but to experimental data, where a priori one know which are the controlled variables.
In Pearl's calculus of intervention proposal, the distinction of decision and chance
variables, does not exist and, in principle, this could be applied to observational data,
as Pearl has suggested and argued convincingly elsewhere [Pearl 1994], Galles and
Pearl, 1995]. His idea is that causal relations can be deduced from DAG structure and
observational data by several graphic criteria. To learn structures reflecting the whole
formalization of Pearl (i.e. the one where each child-parents cluster has a deterministic
function interpretation) some hurdles have to be removed. The difficult part of the
question lies in assessing the form of the functional ties now replacing the links in the
parent-children subgraphs. Learning functions from observational data is no easy task.
Only some kinds of linear combinations of simple functions have been derived from
data by systems like BACON. An improvement on those results are methods that learn
partitioned sets of equations as the FAHRENHEIT [Zykow 1987] system does.

If one sets to the task of learning causal networks with just observational data, it
is clear that the correct approach must follow the lines of Pearl's proposal has to . This
is what we have set ourselves to do. We'll comment on our proposal later.

Referring to Cooper's criteria on distinguish causal relations, he has not provided
up to now any application of them to learning causal networks. However it may be as
simple as building a belief network by any method and then apply test to check weather
his seven relations hold in the learned network.

No study has been carried on in order to identify if any of these characterizations
of causality can be transferred to other uncertainty formalisms. Let's remark, however,
that both Pearl's and Cooper's definitions rely on graphical conditions, so they may be
used in order to identify causality under other formalisms.
8. References


[Bylander 1990] Bylander, Some causal models are deeper than others. Artificial Intelligence in Medicine, (2), pages 123-128.


In *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, Seattle, pages 293-301.


[Mitchell 1981] Learning as Search


LSI-96-17-R “Rewriting in Categories of Spans”, Miquel Mouserrat, Francesc Rosselló, Joan Torrens, and Gabriel Valiente.


LSI-96-19-R “Learning Causal Networks from Data”, Ramon Sangüesa i Solé.


Hardcopies of reports can be ordered from:

Nuria Sánchez
Departament de Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya
Pau Gargallo, 5
08028 Barcelona, Spain
secrelsi@lsi.upc.es

See also the Department WWW pages, http://www-lsi.upc.es/www/