Recursive Causality in Bayesian Networks and Self-Fibring Networks

DOV GABBAY AND JON WILLIAMSON

Causal relations can themselves take part in causal relations. The fact that smoking causes cancer (SC), for instance, causes government to restrict tobacco advertising (A), which helps prevent smoking (S), which in turn helps prevent cancer (C). This causal chain is depicted in Figure 1, and further examples will be given in Section 1.

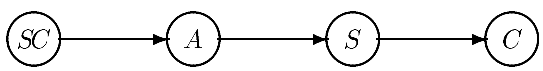


Figure 1. SC: smoking causes cancer; A: tobacco advertising; S: smoking; C: cancer

So causal models need to be able to treat causal relationships as causes and effects. This observation motivates an extension the Bayesian network causal calculus (Section 2) to allow nodes that themselves take Bayesian networks as values. Such networks will be called *recursive Bayesian networks* (Section 3).

Because recursive Bayesian networks make causal and probabilistic claims at different levels of their recursive structure, there is a danger that the network might contradict itself. Hence we need to ensure that the network is consistent, as explained in Section 4. Having done this, in Section 5 we propose a new Markov condition: under this condition a recursive Bayesian network determines a joint probability distribution over its domain.

In Section 6 we compare our approach to other generalisations of Bayesian networks, and in Section 7 we show by analogy with recursive Bayesian networks how recursive causality can be modelled in structural equation models. A similar analogy motivates the application of recursive Bayesian networks to a non-causal domain, namely the modelling of arguments (Section 8).

A recursive Bayesian network is an instance of a very general structure called a self-fibring information network, whose properties are explored in Section 9 and Section 10.

Laws and Models in in Science, 171–219.

1 Causal Relations as Causes

It is almost universally accepted that causality is an asymmetric binary relation.[1] But the question of what the causal relation relates is much more controversial: the relata of causality have variously taken to be single-case events, properties, propositions, facts, sentences and more. In this paper we shall only add to the controversy, by dealing with cases in which causal relations themselves are included as relata of causality. Our aim is to shed light more on the processes of causal reasoning, especially formalisations of causal reasoning, than on the metaphysics of causality.

More generally we shall consider sets of causal relations, represented by directed causal graphs such as that of Figure 1, as relata of causality. (A single causal relationship is then represented by a causal graph consisting of two nodes referring to the relata and an arrow from cause to effect.) If, as in Figure 1, a causal graph G contains a causal relation or causal graph as a value of a node, we shall call G a *recursive causal graph* and say that it represents *recursive causality*.

Perhaps the best way to get a feel for the importance and pervasiveness of recursive causality is through a series of examples.

Policy decisions are often influenced by causal relations. As we have already seen, smoking causing cancer itself causes restrictions on advertising. Similarly, monetary policy makers reduce interest rates (R) because interest rate reductions boost the economy (E) by causing borrowing increases (B) which in turn allow investment (I). Here we have a causal chain as in Figure 2 forming the value of node RE in Figure 3.

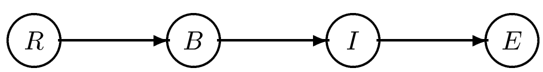


Figure 2. R: interest rate reduction; B: borrowing; I: investment; E: economic boost

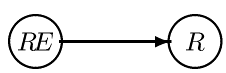


Figure 3. RE: interest rate reduction causing economic boost; R: interest rate reduction

Policy need not be made for us: we often decide how we behave on the basis of perceived causal relationships. It is plausible that drinking red wine causes an

[1] Not quite universally: [Mellor, 1995] disagrees for example.

increase in anti-oxidants which in turn reduces cholesterol deposits, and this apparent causal relationship causes some people to increase their red wine consumption. This example highlights two important points. Firstly, it is a belief in the causal relationship which directly causes the policy change, not the causal relationship itself. The belief in the causal relationship may itself be caused by the relationship, but it may not be—it may be a false belief or it may be true by accident. Likewise, if a causal relationship exists but no one believes that it exists, there will be no policy change. Secondly, the policy decision need not be rational on the basis of the actual causal relationship that causes the decision: drinking red wine may do more harm than good.

A contract can be thought of as a causal relationship, and the existence of a contract can be an important factor in making a decision. A contract in which production of commodity C is purchased at price P may be thought of as a causal relationship $C \longrightarrow P$, and the existence of this causal relationship can in turn cause the producer to invest in further means of production, or even other commodities. For example, a Fair Trade chocolate company has a long-term contract with a co-operative of Ghanaian cocoa producers to purchase (P) cocoa (C) at a price advantageous to the producer as in Figure 4. The existence of this contract (CP) allows the co-operative to invest in community projects such as schools (S), as in Figure 5.

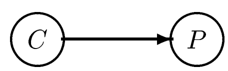


Figure 4. C: cocoa production; P: purchase

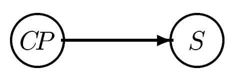


Figure 5. CP: cocoa production causing payment; S school investment

An insurance contract is an important instance of this example of recursive causality. Insuring a building against fire may be thought of as a causal relationship of the form 'insurance contract causes [fire F causes remuneration R]' or $[C \longrightarrow P] \longrightarrow [F \longrightarrow R]$ for short, where as before C is the commodity (i.e. the contract) and P is payment of the premium. The existence of such an insurance policy can cause the policy holder to commit arson (A) and set fire to her building and thereby get remunerated: $[[C \longrightarrow P] \longrightarrow [F \longrightarrow R]] \longrightarrow A \longrightarrow F \longrightarrow R$. Causality in this relationship is nested at three levels. Insurance companies will clearly want to limit the probability of remuneration given that arson has occurred.

Thus we see that recursive causality is particularly pervasive in decision-making scenarios. However, recursive causality may occur in other situations too—situations in which it is the causal relationship itself, rather than someone's belief in the relationship, that does the causing. Pre-emption is an important case of recursive causality, where the pre-empting causal relationship prevents the pre-empted relationship: [poisoning causing death] prevents [heart failure causing death].[2] Context-specific causality may also be thought of recursively: a causal relationship that only occurs in a particular context (such as susceptibility to disease amongst immune-deficient people) can often be thought of in terms of the context causing the causal relationship.

Arguably prevention is often best interpreted in terms of recursive causality: when taking mineral supplements prevents goitre, what is really happening is that taking mineral supplements prevents [poor diet causing goitre]—this is because there are other causes of goitre such as various defects of the thyroid gland, taking mineral supplements does not inhibit these causal chains and thus does not prevent goitre simpliciter. (In many such cases, however, the recursive nature can be eliminated by identifying a particular component of the causal chain which is prevented. Poor diet (D) causes goitre (G) via iodine deficiency (I) and mineral supplements (S) prevent iodine deficiency and so this example might be adequately represented by Figure 6, which is not recursive. Of course the recursive aspect can not be eliminated if no suitable intermediate variable I is known to the modeller.)

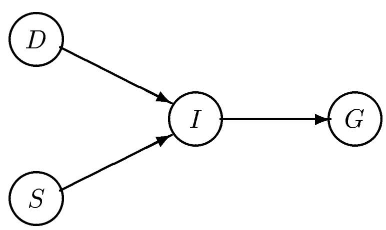


Figure 6. D: poor diet; S: mineral supplements; I: iodine deficiency; G: goitre

Recursive causality is clearly a widespread phenomenon. The question now arises as to how recursive causality ought to influence our reasoning mechanisms. After a brief introduction to Bayesian networks in Section 2 we shall extend the Bayesian network formalism to cope with recursive causality (Section 3) and then discuss some related extensions of Bayesian networks (Section 6). Later we shall see that this approach to causal reasoning generalises in an interesting way.

[2]We suggest that this is a simpler and more natural way of representing pre-emption than the proposal of Section Section 10.1.3, 10.3.3, 10.3.5 of [Pearl, 2000].

2 Bayesian Networks

A Bayesian network is defined over a finite domain $V = \{V_1, \ldots, V_n\}$ of variables. In principle there are no size restrictions on the set of possible values that each variable may take, but often in practice each variable will have only a finite number of possible values. For simplicity we shall restrict our attention to two-valued variables, and denote the assignment of V_i to its values by v_i and $\neg v_i$ respectively, for $i = 1, \ldots, n$. An assignment u to a subset $U \subseteq V$ of variables is a conjunction of assignments to each of the variables in U. For example $v_1 \wedge \neg v_2 \wedge \neg v_5$ is an assignment to $\{V_1, V_2, V_5\}$.

A (causally interpreted) *Bayesian network* b on V consists of two components:

- A directed acyclic graph G with nodes from V, representing the causal relations amongst the variables.

- A probability specification S. For each $V_i \in V$, S specifies the probability distribution of V_i conditional on its parents (direct causes in G), i.e. S consists of statements of the form '$p(v_i|par_i) = x_{i,par_i}$' for each $i = 1, \ldots, n$ and assignment par_i of values to the parents of V_i, and where each $x_{i,par_i} \in [0, 1]$. If the value of a variable V_i is known then V_i is said to be *instantiated* to that value and the corresponding probability specifiers $p(v_i|par_i)$ are 1 or 0 according to whether v_i or $\neg v_i$ is the instantiated value.

The graph and probability specification of a Bayesian network are linked by a fundamental assumption known as the *causal Markov condition*. This says that conditional on its parents, any node is probabilistically independent of all other nodes apart from its descendants, written $V_i \perp\!\!\!\perp ND_i \mid Par_i$ where ND_i and Par_i are respectively the sets of non-descendants and parents of V_i.

A Bayesian network suffices to determine a joint probability distribution over its nodes, since for each assignment v on V,

$$(1) \quad p(v) = \prod_{i=1}^{n} p(v_i|par_i)$$

where v_i is the assignment v gives to V_i, and par_i is the assignment v gives to the parents Par_i of V_i.

Bayesian networks are used because they offer the opportunity of an efficient representation of a joint probability distribution over V. While 2^n different probabilities $p(v)$ specify the joint distribution, these values may (depending on the structure of the causal graph G) be determined via eqn 1 from relatively few values in the probability specification S. Furthermore, a number of algorithms have been developed for determining marginal probabilities from a Bayesian network, often very quickly (but this again depends on the structure of G).[3] Causal graphs are often sparse, and thus lead to efficient Bayesian network representations. Moreover the causal interpretation of the graph ensures that the causal Markov condition is a good default assumption, even if the conditional independence relationships it posits do not always hold in practice.[4]

3 Extension to Recursive Causality

As noted in Section 1, causal relationships often act as causes or effects themselves. In a Bayesian network, however, the nodes tend to be thought of as simple variables, not complex causal relationships. Thus we need to generalise the concept of Bayesian network so that nodes in its causal graph G can signify complex causal relationships. On the other hand, we would like to retain the essential features of ordinary networks, namely the ability to represent joint distributions efficiently, and the ability to perform probabilistic inference efficiently.

The essential step is this. We shall allow variables to take Bayesian networks as values. If a variable takes Bayesian networks as values we will call it a *network variable* to distinguish it from a *simple variable* whose values do not contain such structure. Thus S, which signifies 'payment of subsidy to farmer' and takes value true (s) or false ($\neg s$) is a simple variable. But an example of a network variable is A, which stands for 'agricultural policy' and takes value a signifying the Bayesian network containing the graph of Figure 7 and the specification $\{p_a(f) = 0.1, p_a(s|f) = 0.9, p_a(s|\neg f) = 0.2\}$, where F is a simple variable signifying 'farming', or value $\neg a$ signifying Bayesian net with graph of Figure 8 and specification $\{p_{\neg a}(f) = 0.1, p_{\neg a}(s) = 0.2\}$. Here a is a policy in which farming causes subsidy and $\neg a$ is a policy in which there is no such causal relationship. For simplicity we shall consider network variables with at most two values, but the theory that follows applies to network variables which take any finite number of values.

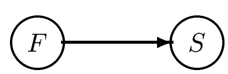


Figure 7. Graph of a: farming causes subsidy

[3]See [Neqpolitan, 1990] for a detailed discussion of the properties of Bayesian networks and key inference algorithms.

[4]See [Williamson, 2001] on this point. While Bayesian networks were originally developed with a causal interpretation in mind [Pearl, 1988], a joint probability distribution can also be represented by a Bayesian network whose graph does not admit a viable causal interpretation. If a Bayesian network is not causally interpreted then causal justifications of the Markov condition do not apply, and an independent justification is required. Thus in Section 5 we define a network called a *flattening* which contains arrows that do not correspond to causal relations, and we also provide a justification for the Markov condition.

Figure 8. Graph of $\neg a$: no causal relationship between farming and subsidy

A *recursive Bayesian network* is then a Bayesian network containing at least one network variable. For example the network with graph Figure 9 and specification $\{p(l) = 0.7, p(a|l) = 0.95, p(a|\neg l) = 0.4\}$, representing the causal relationship between lobbying and agricultural policy, is a recursive Bayesian network, where the simple variable L stands for 'lobbying' and takes value true or false, and A is the network variable signifying 'agricultural policy' discussed above.

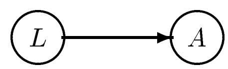


Figure 9. Lobbying causes agricultural policy

We shall allow network variables to take recursive Bayesian networks (as well as the standard Bayesian networks of Section 2) as values. In this way a recursive Bayesian network represents a hierarchical structure.

If a variable C is a network variable then the variables that occur as nodes in the Bayesian networks that are the values of C are called the *direct inferiors* of C, and each such variable has C as a *direct superior*. *Inferior* and *superior* are the transitive closures of these relations: thus E is inferior to C iff it is directly inferior to C or directly inferior to a variable D that is inferior to C. The variables that occur in the same local network as C are called its *peers*.

A recursive Bayesian network $b = (G, S)$ conveys information on a number of levels. The variables that are nodes in G are *level 1*; any variables directly inferior to level 1 variables are *level 2*, and so on. The network b itself can be associated with a network variable B that is instantiated to value b, and we can speak of B as the *level 0 variable*. (We have not specified the other possible values of B: for concreteness we can suppose that B is a single-valued network variable which only takes value b.) The *depth* of the network is the maximum level attained by a variable. A Bayesian network is *non-recursive* if its depth is 1; it is *well-founded* if its depth is finite. We shall restrict our discussion to *finite* networks: well-founded networks whose levels are each of finite size.

For $i \geq 0$ let $\mathcal{V}_i$ be the set of level i variables, and let $\mathcal{G}_i$ and $\mathcal{S}_i$ be the set of graphs and specifications respectively that occur in networks that are values of level i variables. Thus $\mathcal{V}_0 = \{B\}, \mathcal{G}_0 = \{G\}$ and $\mathcal{S}_0 = \{S\}$. The domain of b is the set $V = \bigcup_i \mathcal{V}_i$ of variables at all levels.

Note that V contains the level 0 variable B itself and thus contains all the structure of b. In our example $V = \{B, L, A, F, S\}$ where the level 0 network variable B takes value b whose graph is Figure 9 and whose probability specification is $\{p(l) = 0.7, p(a|l) = 0.95, p(a|\neg l) = 0.4\}$ and the only other network variable is A whose value a has graph of Figure 7 and specification $\{p_a(f) = 0.1, p_a(s|f) = 0.9, p_a(s|\neg f) = 0.2\}$ and whose value $\neg a$ has graph of Figure 8 and specification $\{p_{\neg a}(f) = 0.1, p_{\neg a}(s) = 0.2\}$; then V itself determines all the structure of the recursive Bayesian network in question. Consequently we can talk of 'recursive Bayesian network b on domain V' and 'recursive Bayesian network of V' interchangeably.

A network variable V_i can be thought of as a simple variable V_i' if one drops the Bayesian network interpretation of each of its values: V_i' is the *simplification* of V_i. A recursive network b can then be interpreted as a non-recursive network b' on domain $\mathcal{V}_1' = \{V_i' : V_i \in \mathcal{V}_1\}$: then b' is called the *simplification* of b.

A variable may well occur more than once in a recursive Bayesian network, in which case it might have more than one level.[5] Note that in a well-founded network no variable can be its own superior or inferior. A recursive Bayesian network makes causal and probabilistic claims at all its various levels, and if variables occur more than once in the network, these claims might contradict each other. We shall examine this possibility now.

4 Consistency

Network variables that occur in the domain of a recursive Bayesian network $b = (G, S)$ can be interpreted as making causal and probabilistic claims about the world. Any network variable that is instantiated to a particular value asserts the validity of the network to which it is instantiated. In particular the level 0 network variable B asserts its instantiated value b, i.e. it asserts the causal relations in G, the probabilistic independence relationships one can derive from G via the causal Markov condition, and the probabilistic claims made by the probability specification S. A network variable that is not instantiated asserts the weaker claim that precisely one of its possible values is correct. A recursive Bayesian network is consistent if these claims do not contradict each other.

In order to give a more precise formulation of the consistency requirement we need first to define consistency of non-recursive Bayesian networks. There are three desiderata: consistency with respect to causal claims (*causal consistency*), consistency with respect to implied probabilistic independencies (*Markov consis-*

[5] While one might think that there will be no repetition of variables if all variables correspond to single-case events, this is not so. Event A causing event B causes an agent to change her belief about the relationship between A and B, this belief being represented by network variable C whose value $\neg c$ has B causing A and whose value c has A causing B. Here A and B occur more than once in the network but need not be repeatably instantiatable variables—they may be single-case events.

tency) and consistency with respect to probabilistic specifiers (*probabilistic consistency*).

First causal consistency. A *chain* $A \rightsquigarrow B$ from node A to node B in a directed acyclic graph is a sequence of nodes in the graph, beginning with A and ending with B, such that there is an arrow from each node to its successor. A *subchain* of a chain c from A to B is a chain from A to B involving nodes in c in the same order, though not necessarily all the nodes in c. Thus Figure 10 contains both the chain (A, C, B) and its subchain (A, B). The *interior* of a chain $A \rightsquigarrow B$ is defined as the subchain involving all nodes between A and B in the chain, not including A and B themselves. A *path* between A and B is a sequence of nodes in the graph, beginning with A and ending with B, such that there is an arrow between each node and its successor (the direction of the arrow is unimportant).

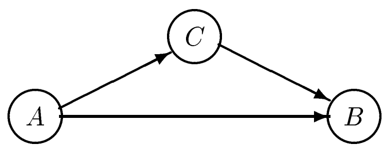


Figure 10.

The restriction $G_{\downharpoonleft W}$ of causal graph G defined on variables V to the set of variables $W \subseteq V$ is defined as follows: for variables $A, B \in W$, there is an arrow $A \longrightarrow B$ in $G_{\downharpoonleft W}$ if and only if $A \longrightarrow B$ is in G or, $A \rightsquigarrow B$ is in G and the variables in the interior of this chain are in $V \backslash W$. Thus G and $G_{\downharpoonleft W}$ agree as to the causal relationships amongst variables in W. It is not hard to see that for $X \subseteq W \subseteq V, G_{\downharpoonleft W \downharpoonleft X} = G_{\downharpoonleft X}$.

Two causal graphs G on V and H on W are *causally consistent* if there is a third (directed and acyclic) causal graph F on $U = V \cup W$ such that $F_{\downharpoonleft V} = G$ and $F_{\downharpoonleft W} = H$. Thus G and H are causally consistent if there is a model F of the causal relationships in both G and H. Such an F is called a *causal supergraph* of G and H.

Figure 11 and Figure 12 are causally consistent for instance, because the latter graph is the restriction of the former to $\{A, B, C\}$. However, Figure 10 is not causally consistent with Figure 11: they do not agree as to the causal chains between A, B and C. Similarly Figure 10 and Figure 12 are causally inconsistent.

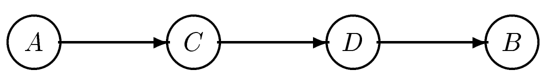


Figure 11.

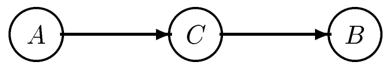


Figure 12.

Note that if G and H are causally consistent and nodes A and B occur in both G and H then there is a chain $A \rightsquigarrow B$ in G iff there is a chain $A \rightsquigarrow B$ in H. We will define two non-recursive Bayesian networks to be *causally consistent* if their causal graphs are causally consistent.

Another important consistency requirement is Markov consistency. Two causal graphs G and H are *Markov consistent* if they posit (via the causal Markov condition) the same set of conditional independence relationships on the nodes they share. Figure 11 and Figure 12 are Markov consistent because on their shared nodes A, C, B they each imply just that A and B are probabilistically independent conditional on C. Figure 10 is not Markov consistent with either of these graphs because it does not imply this independency. Two non-recursive Bayesian networks are *Markov consistent* if their causal graphs are Markov consistent.

Note that Markov consistency does not imply causal consistency: for instance two different complete graphs on the same set of nodes (graphs, such as Figure 10, in which each pair of nodes is connected by some arrow) are Markov consistent, since neither graph implies any independence relationships, but causally inconsistent because where they differ, they differ as to the causal claims they make. Neither does causal consistency of a pair of causal graphs imply Markov consistency: Figure 13 and Figure 14 are causally consistent but Figure 14 implies that A and B are probabilistically independent, while Figure 13 does not.

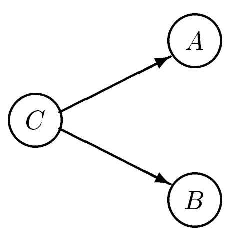


Figure 13.

In fact we have the following. Let $Com_G(X)$ be the set of closest common causes of X according to G, that is, the set of causes C of X that are causes of at least two nodes A and B in X for which some pair of chains from C to A and C to B only have node C in common. Then,

Figure 14.

PROPOSITION 1. *Suppose G and H are causal graphs on V and W respectively. G and H are Markov consistent if they are causally consistent and their shared nodes are closed under closest common causes ('cccc' for short), $Com_G(V \cap W) \cup Com_H(V \cap W) \subseteq V \cap W$.*

Proof. Suppose $X \perp\!\!\!\perp_G Y \mid Z$ for some $X, Y, Z \subseteq V \cap W$. Then for each $A \in X$ and $B \in Y$, Z *D-separates* A from B in G: every path between A and B is *blocked* by Z, i.e. every path contains (i) a structure $\longrightarrow C \longrightarrow$ with C in Z, or (ii) a structure $\longleftarrow C \longrightarrow$ with C in Z, or (iii) a structure $\longrightarrow C \longleftarrow$ where Z contains neither C nor any of its descendants.[6] G and H are causally consistent so there is a causal supergraph F on $V \cup W$ ($G = F_{\downharpoonright V}$ and $H = F_{\downharpoonright W}$). Now consider a path between A and B in F. Such a path either (a) is a chain $(A \rightsquigarrow B$ or $B \rightsquigarrow A)$, (b) contains some C where $C \rightsquigarrow A$ and $C \rightsquigarrow B$, or (c) contains a $\longrightarrow C \longleftarrow$ structure. In case (a) there must be in G a subchain of this chain which is blocked by Z so the original chain in F must also be blocked by Z. Similarly in case (b), since G and H are cccc there must be a blocked subpath in G which has $C \rightsquigarrow A$ and $C \rightsquigarrow B$. In case (c), either there is a corresponding subpath in G which is blocked, or C and its descendants are not in Z so the path in F is blocked in any case. Thus $X \perp\!\!\!\perp_F Y \mid Z$. Next take the restriction $F_{\downharpoonright W} = H$. Paths between A and B in H must be blocked by Z since they are subpaths of paths in F that are blocked by Z and all variables in Z occur in H. Thus $X \perp\!\!\!\perp_H Y \mid Z$, as required. ■

Note that while (under the assumption of causal consistency) closure under closest common causes is a sufficient condition for Markov consistency, it is not a necessary condition: Figure 13 and Figure 15 are Markov consistent because neither imply any independencies just amongst their shared nodes A and B, but the set of shared nodes is not closed under closest common causes.

[6] D-separation is a necessary and sufficient condition for deciding the conditional independencies implied by a causal graph under the causal Markov condition. See[Pearl, 1988, Section 3.3.1].

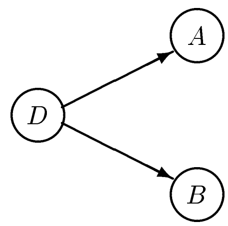


Figure 15.

Markov consistency is quite a strong condition. It is not sufficient merely to require that the pair of causal graphs imply sets of conditional independence relations that are consistent with each other—in fact any two graphs satisfy this property. The motivation behind indexconsistency!MarkovMarkov consistency is based on the fact that a cause and its effect are usually probabilistically dependent conditional on the effect's other causes (this property is known as the *causal dependence condition*), in which case probabilistic independencies that are not implied by the causal Markov condition are unlikely to occur. For example, while the fact that C causes A and B (Figure 13) is consistent with A and B being unconditionally independent (Figure 14), it makes their independence extremely unlikely: if A and B have a common cause then the occurrence of assignment a of A may be attributable to the common cause which then renders b more likely (less likely, if the common cause is a preventative), in which case A and B are unconditionally dependent. Thus Figure 13 and Figure 14 are not compatible, and we need the stronger condition that independence constraints implied by each graph should agree on the set of nodes that occur in both graphs.

Finally we turn to probabilistic consistency. Two causally consistent non-recursive Bayesian networks (G, S) and (H, T), defined over V and W respectively, are *probabilistically consistent* if there is some non-recursive Bayesian network (F, R), defined over $V \cup W$ and where F is a causal supergraph of G and H, whose induced probability function satisfies all the equalities in $S \cup T$. Such a network is called a *causal supernet* of (G, S) and (H, T).

PROPOSITION 2. *Suppose two non-recursive Bayesian networks (G, S) and (H, T) are causally consistent, probabilistically consistent and closed under closest common causes (cccc). Then there is a causal supernet (F, R) of (G, S) and (H, T) that is cccc with (G, S) and (H, T).*

Proof. Because (G, S) and (H, T) are causally and probabilistically consistent, there is a supernet (E, Q), of (G, S) and (H, T). If E is cccc with G and H then we set $(F, R) = (E, Q)$ and we are done. Otherwise, if E is not cccc with G say, then there is some Y-structure of the form of Figure 16 in E, where Figure 17 is the corresponding structure in G. (In these diagrams take the arrows to signify the existence of causal chains rather than direct causal relations.) Note that B must be in G or H, since the domain of a causal supergraph of G and H is the union of the domains of G and H; B cannot be in G since otherwise by causal consistency the chain from A to C in G would go via B; hence B is in H. Note also that not both of C and D can be in H, for otherwise G and H are not cccc. Suppose then that D is not in H. Then the chain from B to D is not in G or H. Construct F by taking E, removing the chain from B to D and including a chain from A to D, as in Figure 18. (Do this for all such Y-structures not replicated in G.) F remains a causal supergraph of G and H, since the chain from B to H was redundant. Moreover F is now cccc with G. Next construct the associated probability specification R by determining specifiers from (E, Q). Thus if the causal chain from A to D is direct we can set $p(d|a) = \sum_b p_{(E,Q)}(d|b)p_{(E,Q)}(b|a)$ in R. It is not hard to see that $p_{(F,R)}$ agres with $p_{(E,Q)}$ on the specifiers in S and T so the new network is also a causal supernet of (G, S). If E is not cccc with H then repeat this algorithm, to yield a causal supernet of (G, S) and (H, T) that is cccc with (G, S) and (H, T). ■

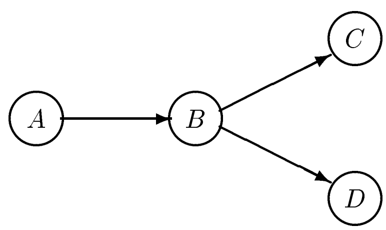


Figure 16. B is the closest common cause of C and D

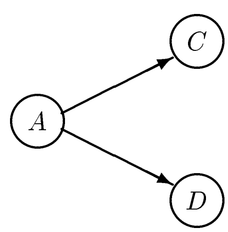


Figure 17. A is the closest common cause of C and D.

Note that the requirement that G and H are cccc in the above result is essential. If G is Figure 16 and H is Figure 17 then there is no causal supergraph of G and

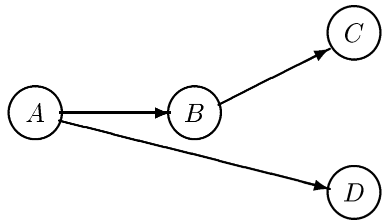


Figure 18. A is the closest common cause of C and D.

H that is cccc with G and H.

PROPOSITION 3. *Suppose two non-recursive Bayesian networks are causally consistent, probabilistically consistent and cccc. Then they determine the same probability function over the variables they share.*

Proof. Suppose (G, S) and (H, T) are causally and probabilistically consistent and cccc. Then by Proposition 2 there is a causal supernet (F, R) that is cccc with both nets. By Proposition 1 F is Markov consistent with G and H.

Next note that (G, S) and (F, R) determine the same probability function over variables V of (G, S):

$$p_{(G,S)}(v) = \prod_{v_i \in V} p_{(G,S)}(v_i | par_i^G)$$

where par_i^G is the state of the parents of V_i according to G that is consistent with assignment v to V,

$$= \prod_{v_i \in V} p_{(F,R)}(v_i | par_i^G)$$

since (F, R) is a causal supernet of (G, S),

$$= \prod_{v_i \in V} p_{(F,R)}(v_i | v_1, \ldots, v_{i-1}) = p_{(F,R)}(v)$$

where it is supposed that the variables $V_1, \ldots, V_n$ in V are ordered G-ancestrally, i.e. no descendants of V_i in G occur before V_i in the order. This last step follows because $V_i \perp\!\!\!\perp_G V_1, \ldots, V_{i-1} \mid Par_i^G$ implies $V_i \perp\!\!\!\perp_F V_1, \ldots, V_{i-1} \mid Par_i^G$ by Markov consistency.

Similarly (H, T) and (F, R) determine the same probability function over the variables of (H, T). Hence (G, S) and (H, T) determine the same probability function over variables they share. ∎

Because Proposition 3 is a desirable property in itself we shall adopt closure under closest common causes as a consistency condition. We shall say that two non-recursive networks are *consistent* if they are causally and probabilistically consistent, and cccc. By Proposition 1 consistency implies Markov consistency.

Having elucidated concepts of consistency for non-recursive networks, we can now say what it means for a recursive network to be consistent.

An assignment v of values to variables in V, the domain of a recursive Bayesian network b, assigns values to all the simple variables and network variables that occur in b. Take for instance the recursive Bayesian network b of Figure 9: here $V = \{B, L, A, F, S\}$ and $b \wedge l \wedge \neg a \wedge f \wedge \neg s$ is an example of an assignment to V. (Note that the level 0 variable B only takes one value b and so must always be assigned this value.) Consider the assignment of values v gives to network variables in V. In our example, the network variables are B and A and these are assigned values b and $\neg a$ respectively. Each such value is itself a recursive Bayesian network, and when simplified induces a non-recursive Bayesian network. Let $\underline{b}_v$ denote the set of recursive Bayesian networks induced by v (i.e. the set of values v assigns to network variables of b) and let $\underline{b}'_v$ denote the set of non-recursive Bayesian networks formed by simplifying the networks in $\underline{b}_v$.

Assignment v is *consistent* if each pair of networks in $\underline{b}'_v$ is consistent (i.e. if each pair of values of network variables is consistent, when these values are interpreted non-recursively). A recursive Bayesian network is *consistent* if it has some consistent assignment v of values to V. A consistent assignment of values to the variables in a network can be thought of as a model or possible world, in which case consistency corresponds to satisfiability by a model.

In sum, if a recursive Bayesian network is not to be self-contradictory there must be some assignment under which all pairs of network variables satisfy three regularity conditions: causal consistency, probabilistic consistency and closure under closest common causes.

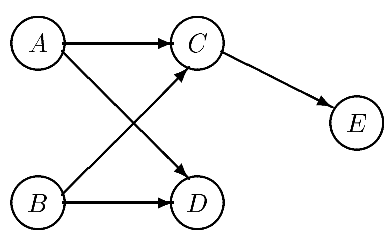


Figure 19. Graph G_1

Note that it is easy to turn a recursive network into one that is causally consistent, by ensuring that causal chains correspond for some assignment, and then cccc (and so Markov consistent), by ensuring that shared nodes of pairs of graphs also share closest common causes, for some assignment. In order to make G_2 in Figure 20 causally consistent with graph G_1 of Figure 19, for example, we need to introduce a chain that corresponds to the chain (D, F, E) in G_2, by adding an arrow from D to E in G_1. In order to make G_2 and G_1 cccc (and so Markov consistent) we need to add B to G_2 as a closest common cause of C and D. The modified graphs are depicted in Figure 21 and Figure 22.

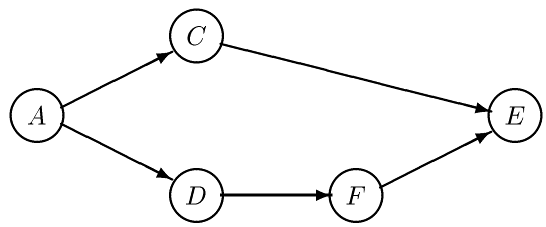


Figure 20. Graph G_2

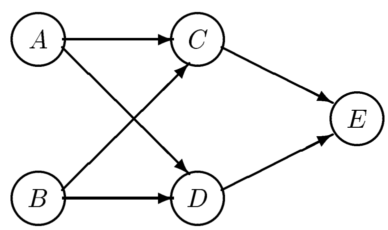


Figure 21. Graph H_1

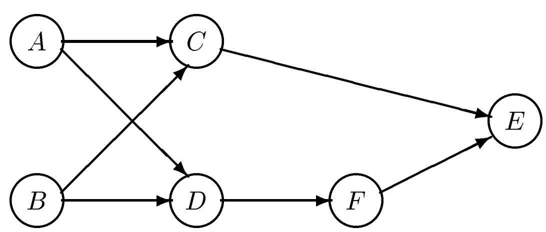


Figure 22. Graph H_2

Similarly in practice one would not expect each probability specification to be provided independently and then to have the problem of checking consistency—

ne would expect to use conditional distributions in one specification to determine distributions in others. For example, a probability specification on H_2 in Figure 22 would completely determine a probability specification on H_1 in Figure 21.

5 Joint Distributions

Any non-recursive Bayesian network is subject to the causal Markov condition (Section 2) which determines a joint probability distribution over the variables of the network from its graph and probability specification. We shall suppose that recursive Bayesian networks also satisfy the causal Markov condition. A recursive Bayesian network contains network variables whose values are interpreted as (recursive or non-recursive) Bayesian networks. Thus a recursive Bayesian network suffices to determine a hierarchy of joint probability distributions p_a on the (level 1) variables of a, for each a that occurs as the value of a network variable. (I.e. a recursive Bayesian network b determines a joint distribution on each network in $\underline{b}'_v$ for each consistent assignment v to the domain of b.) Standard Bayesian network algorithms can be used to perform inference in a recursive Bayesian network, and the range of causal-probabilistic questions that can be addressed is substantially increased. For example one can answer questions like 'what is the probability of a subsidy given farming?' (see Figure 7) and 'what is the probability of lobbying given agricultural policy $\neg a$?' (see Figure 9).

Certain questions remain unanswered however. We can not as yet determine the probability of one node conditional on another if the nodes only occur at different levels of the network. For example we can not answer the question 'what is the probability of subsidy given lobbying?' While we have a hierarchy of joint distributions, we have not yet specified a single joint distribution over the set of nodes in the union of the graph, i.e. over the recursive network as a whole.

In fact as we shall see, a recursive network does determine such an over-arching joint distribution if we make an extra independence assumption, called the *recursive Markov condition*: each variable is probabilistically independent of those other variables that are neither its inferiors nor its peers, conditional on its direct superiors.

A precise explication of the causal Markov condition and recursive Markov condition will be given shortly.

Given a recursive Bayesian network domain V and a consistent assignment v of values to V, we construct a non-recursive Bayesian network, the *flattening*, $v^{\downarrow}$, of v as follows. The domain of $v^{\downarrow}$ is V itself. The graph $G^{\downarrow}$ of $v^{\downarrow}$ has variables in V as nodes, each variable occurring only once in the graph. Add an arrow from V_i to V_j in $G^{\downarrow}$ if

- V_i is a parent of V_j in v (i.e. there is an arrow from V_i to V_j in the graph of some value of v) or

- V_i is a direct superior of V_j in v (i.e. V_j occurs in the graph of the value that v assigns to V_i).

We will describe the probability specification $S^{\downarrow}$ of $v^{\downarrow}$ in due course. First to some properties of the graph $G^{\downarrow}$.

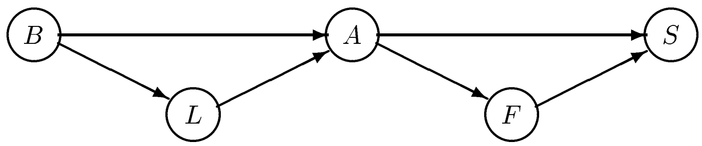


Figure 23. Example flattening

Note that $G^{\downarrow}$ may or may not be acyclic. If we take our farming example $V = \{B, L, A, F, S\}$ of Section 3, then the graph of the flattening $(b \wedge \neg l \wedge a \wedge f \wedge s)^{\downarrow}$ is depicted in Figure 23 and is acyclic. But the graph of the flattening of assignment $b \wedge c \wedge d \wedge e$ to $\{B, C, D, E\}$, where B is the level 0 network variable whose value b has graph $C \longrightarrow D$, C and E are simple variables and D is a network variable whose assigned value d has the graph $E \longrightarrow C$, is cyclic. The graph in a non-recursive Bayesian network must be acyclic in order to apply standard Bayesian network algorithms, and this requirement extends to recursive Bayesian networks: we will focus on consistent *acyclic* assignments to a recursive Bayesian network domain, those consistent assignments v that lead to an acyclic graph in the flattening $v^{\downarrow}$.[7]

By focussing on consistent acyclic assignments v, the following explications of the two independence conditions become plausible. Given a consistent acyclic assignment v, let PND_i^v be the set of variables that are peers but not descendants of V_i in v, NIP_i^v be the non-inferiors or peers of V_i, and $DSup_i^v$ be the direct superiors of V_i. As before, Par_i^v are the parents of V_i and ND_i^v are the non-descendants of V_i. None of these sets are taken to include V_i itself.

Causal Markov Condition (CMC) For each $i = 1, \ldots, n$ and $DSup_i^v \subseteq X \subseteq NIP_i^v$, $V_i \perp\!\!\!\perp PND_i^v \mid Par_i^v, X$.

Recursive Markov Condition (RMC) For each $i = 1, \ldots, n$ and $Par_i^v \subseteq X \subseteq PND_i^v$, $V_i \perp\!\!\!\perp NIP_i^v \mid DSup_i^v, X$.

Then the graph of the flattening has the following property:

PROPOSITION 4. *Suppose v is a consistent acyclic assignment to a recursive Bayesian network domain V. Then the probabilistic independencies implied by v*

[7]Cyclic Bayesian networks have been studied to some extent, but are less tractable than the acyclic case: see [Spirtes, 1995] and [, Neal 2000].

via the causal Markov condition and the recursive Markov condition are just those implied by the graph $G^{\downarrow}$ of the flattening $v^{\downarrow}$ via the causal Markov condition.

Proof. Order the variables in V ancestrally with respect to $G^{\downarrow}$, i.e. no descendants of V_i in $G^{\downarrow}$ occur before V_i in the ordering—this is always possible because $G^{\downarrow}$ is acyclic.

First we shall show that CMC and RMC for v imply CMC for $G^{\downarrow}$. By Corollary 3 of [Pearl, 1988] it suffices to show that $V_i \perp\!\!\!\perp V_1, \ldots, V_{i-1} \mid Par_i^{G^{\downarrow}}$ for any $V_i \in V$. By CMC, $V_i \perp\!\!\!\perp PND_i^v \mid Par_i^v, DSup_i^v$, and by RMC, $V_i \perp\!\!\!\perp NIP_i^v \mid DSup_i^v, PND_i^v$. Applying contraction,[8]

$$V_i \perp\!\!\!\perp PND_i^v \cup NIP_i^v \mid Par_i^v, DSup_i^v.$$

Now $\{V_1, \ldots, V_{i-1}\} \subseteq PND_i^v \cup NIP_i^v$ since the variables are ordered ancestrally and v is acyclic, and the parents of V_i in $G^{\downarrow}$ are just the parents and direct superiors of V_i in v, $Par_i^{G^{\downarrow}} = Par_i^v \cup DSup_i^v$, so $V_i \perp\!\!\!\perp V_1, \ldots, V_{i-1} \mid Par_i^{G^{\downarrow}}$ as required.

Next we shall see that CMC for $G^{\downarrow}$ implies CMC and RMC for v. In fact this follows straightforwardly by D-separation. $Par_i^v \cup X$ D-separates V_i and PND_i^v in $G^{\downarrow}$ for any $DSup_i^v \subseteq X \subseteq NIP_i^v$, since $Par_i^v \cup X$ includes the parents of V_i in $G^{\downarrow}$ and (by acyclicity of v) PND_i^v are non-descendants of V_i in $G^{\downarrow}$, so CMC holds. $DSup_i^v \cup X$ D-separates V_i and NIP_i^v in $G^{\downarrow}$ for any $Par_i^v \subseteq X \subseteq PND_i^v$, since $DSup_i^v \cup X$ includes the parents of V_i in $G^{\downarrow}$ and (by acyclicity of v) NIP_i^v are non-descendants of V_i in $G^{\downarrow}$, so RMC holds. ■

Having defined the graph $G^{\downarrow}$ in the flattening $v^{\downarrow}$ of v, and examined its properties, we shall move on to define the probability specification $S^{\downarrow}$ of $v^{\downarrow}$. In the specification $S^{\downarrow}$ we need to provide a value for $p(v_i|par_i^{G^{\downarrow}})$ for each value v_i of V_i and assignment $par_i^{G^{\downarrow}}$ of the parents $Par_i^{G^{\downarrow}}$ of V_i in $G^{\downarrow}$. If V_i only occurs once in the recursive Bayesian network determined by v then we can define

$$p(v_i|par_i^{G^{\downarrow}}) = p(v_i|dsup_i^v \wedge par_i^v) = p_{dsup_i^v}(v_i|par_i^v),$$

which is provided in the specification of the value of V_i's direct superior in v. If V_i occurs more than once in the recursive Bayesian network determined by v then the specifications of v contain $p_{dsup_i^G}(v_i|par_i^G)$ for each graph G in v in which V_i occurs. Then $DSup_i^v = \bigcup_G DSup_i^G$ and $Par_i^v = \bigcup_G Par_i^G$, with the unions taken over all such G. Now the specifiers $p_{dsup_i^G}(v_i|par_i^G)$ constrain the value of $p_{dsup_i^v}(v_i|par_i^v)$ but may not determine it completely. These are linear constraints, though, and thus there is a unique value for $p_{dsup_i^v}(v_i|par_i^v)$ which maximises

[8] Contraction is the following property of probabilistic independence: $R \perp\!\!\!\perp S|T$ and $R \perp\!\!\!\perp U|S,T \Rightarrow R \perp\!\!\!\perp S, U|T$. See e.g. [Pearl, 1988].

entropy subject to the constraints holding—this can be taken as its optimal value,[9] and $p(v_i|par_i^{G^\downarrow})$ can be set to this value.[10]

Having fully defined the flattening $v^\downarrow = (G^\downarrow, S^\downarrow)$ and shown that the causal Markov condition holds, we have a (non-recursive) Bayesian network,[11] which can be used to determine a probability function over assignments to v:

PROPOSITION 5. *A recursive Bayesian network determines a unique joint distribution over consistent acyclic assignments v of values to its domain, defined by*

$$p(v) = \prod_{i=1}^{n} p(v_i|par_i^{G^\downarrow})$$

where $G^\downarrow$ is the graph in the flattening $v^\downarrow$ of v and $p(v_i|par_i^{G^\downarrow})$ is the value in the specification $S^\downarrow$ of $v^\downarrow$. (As usual v_i is the value v assigns to V_i and $par_i^{G^\downarrow}$ is the assignment v gives to the parents of V_i according to $G^\downarrow$.)[12]

While a flattening is a useful concept to explain how a joint distribution is defined, there is no need to actually construct flattenings when performing calculations with recursive networks—indeed that would be most undesirable, given that there are exponentially many assignments and thus exponentially many flattenings which would need to be constructed and stored. By Proposition 5, only the probabilities $p(v_i|par_i^v \wedge dsup_i^v)$ need to be determined, and in many cases (i.e. when V_i occurs only once in v) these are already stored in the recursive network.

The concept of flattening, in which a mapping is created between a recursive network and a corresponding non-recursive network, also helps us understand how standard inference algorithms for non-recursive Bayesian networks can be directly applied to recursive networks. For example, message-passing propagation algorithms[13] can be directly applied to recursive networks, as long as messages are passed between direct superior and direct inferior as well as between parent and child. Moreover, recursive Bayesian networks can be used to reason about interventions just as can non-recursive networks: when one intervenes to fix the value of a variable one must treat that variable as a root node in the network, ignoring any connections between the node and its parents or direct superiors.[14] In effect, tools

[9] [Jaynes, 1957].

[10] See [Williamson, 2002] for more on maximising entropy.

[11] Note that this Bayesian network is not causally interpreted, since arrows from superiors to direct inferiors are not causal arrows.

[12] Here the domain of p is the set of assignments to V, and p is unique over consistent acyclic assignments. If one wants to take just the set of consistent acyclic assignments as domain of p (equivalently, to award probability 0 to inconsistent or cyclic assignments) then one must renormalise, i.e. divide $p(v)$ by $\sum p(v)$ where the sum is taken over all consistent acyclic assignments.

[13] See [Pearl, 1988] [Neqpolitan, 1990].

[14] [Pearl, 2000] Section 1.3.1.

for handling non-recursive Bayesian networks can be easily mapped to recursive networks.

A word on the plausibility of the recursive Markov condition. It was shown in [Williamson, 2001] that the causal Markov condition can be justified as follows: suppose an agent's background knowledge consists of the components of a causally interpreted Bayesian network—knowledge of causal relationships embodied by the causal graph and knowledge of probabilities encapsulated in the corresponding probability specification—then the agent's degrees of belief ought to satisfy the causal Markov condition.[15] This justification rests on the acceptance of the maximum entropy principle (which says that an agent's belief function should be the probability function, out of all those that satisfy the constraints imposed by background knowledge, that has maximum entropy) and the causal irrelevance principle (which says that if an agent learns of the existence of new variables which are not causes of any of the old variables, then her degrees of belief concerning the old variables should not change). An analogous justification can be provided for the recursive Markov condition. Plausibly, learning of new variables that are not superiors (or causes) of old variables should not lead to any change in degrees of belief over the old domain. Now if an agent's background knowledge takes the form of the components of a recursive Bayesian network then the maximum entropy function, and thus the agent's degrees of belief, will satisfy the recursive Markov condition as well as the causal Markov condition. Thus a justification can be given for both the causal Markov condition and the recursive Markov condition.

6 Related Work

Bayesian networks have been extended in a variety of ways, and some of these are loosely connected with the recursive Bayesian networks introduced above.

Recursive Bayesian multinets generalise Bayesian networks along the following lines.[16] First, Bayesian networks are generalised to *Bayesian multinets* which represent context-specific independence relationships by a set of Bayesian networks, each of which represents the conditional independencies which operate in a fixed context. By creating a variable C whose assignments yield different contexts, a Bayesian multinet may be represented by decision tree whose root is C and whose leaves are the Bayesian networks. The idea behind recursive Bayesian multinets is to extend the depth of such decision trees. Leaf nodes are still Bayesian networks, but there may be several decision nodes. For example, Figure 24 depicts a recursive Bayesian multinet in which there are three decision nodes, C_1, C_2 and C_3, and four Bayesian networks B_1, B_2, B_3, B_4. Node C_1 has two possible contexts as values; under the first node C_2 comes into operation; this has two possible contexts as values; under the first Bayesian network B_1 describes the domain; un-

[15] See also [Williamson, 2002].

[16] [Peña *et al.*, 2002].

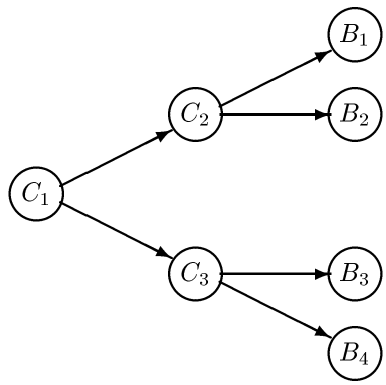


Figure 24. A recursive Bayesian multinet

der the second B_2 applies, and so on. Figure 24 is recursive in the sense that depending on the value of C_1, a different multinet is brought into play—the multinet on C_2, B_1, B_2 or that on C_3, B_3, B_4. Thus recursive Bayesian multinets are rather different to our recursive Bayesian networks: they are applicable to context-specific causality where the contexts need to be described by multiple variables, [17] not to general instances of recursive causality, and consequently they are structurally different, being decision trees whose leaves are Bayesian networks rather than Bayesian networks whose nodes take Bayesian networks as values.

Recursive relational Bayesian networks generalise the expressive power of the domain over which Bayesian networks are defined. [18] Bayesian networks are essentially propositional in the sense that they are defined on variables, and the assignment of a value to a variable can be thought of as a proposition which is true if the assignment holds and false otherwise. We have made this explicit by representing the two possible assignments to variable A by a and $\neg a$ respectively. *Relational Bayesian networks* generalise Bayesian networks by enabling them to represent probability distributions over more fine-grained linguistic structures, in particular certain sub-languages of first-order logical languages. Recursive relational Bayesian networks generalise further by allowing more complex probabilistic constraints to operate, and by allowing the probability of an atom that instantiates a node to depend recursively on other instantiations as well as the node's parents.[19] Thus in the transition from relational Bayesian networks to recursive relational Bayesian networks the Markovian property of a node being dependent

[17]The particular application that motivated their introduction was data clustering—see[Peña *et al.*, 2002].

[18][Jaeger, 2002].

[19]See [Jaeger, 2002] for the details.

just on its parents (not further non-descendants) is lost. Therefore recursive relational Bayesian networks and recursive Bayesian networks differ fundamentally with respect to both motivating applications and formal properties.

Object-oriented Bayesian networks were developed as a formalism for representing large-scale Bayesian networks efficiently.[20] Object-oriented Bayesian networks are defined over *objects*, of which a variable is but one example. Such networks are in principle very general, and recursive Bayesian networks are instances of object-oriented Bayesian networks in as much as recursive Bayesian networks can be formulated as objects in the object-oriented programming sense. Moreover in practice object-oriented Bayesian networks often look much like recursive Bayesian networks, in that such a network may contain several Bayesian networks as nodes, each of which contains further Bayesian networks as nodes and so on.[21] However, there is an important difference between the semantics of such object-oriented Bayesian networks and that of recursive Bayesian networks, and this difference is dictated by their motivating applications. Object-oriented Bayesian networks tend to be used to organise information contained in several Bayesian networks: each such Bayesian network is viewed as a single object node in order to hide much of its information that is not relevant to computations being carried out in the containing network. Hence when there is an arrow from one Bayesian network B_1 to another B_2 in the containing network, this arrow hides a number of arrows from output variables (which are often leaf variables) of B_1 to input variables (often root variables) of B_2. So by expanding each Bayesian network node, an object-oriented Bayesian network can be expanded into one single non-recursive, non-object-oriented Bayesian network. In contrast, in a recursive Bayesian network, recursive indexBayesian network!recursiveBayesian networks occur as *values* of nodes not as nodes themselves, and when one recursive Bayesian network b_1 causes another b_2 in a containing recursive Bayesian network b, it is not output variables of b_1 that cause input variables of b_2, it is b_1 *as a whole* that causes b_2 *as a whole*. Correspondingly, there is no straightforward mapping of a recursive Bayesian network on V to a Bayesian network on V: mappings (flattenings) are relative to assignment v to V. Thus while object-oriented Bayesian networks are in principle very general, in practice they are often used to represent very large Bayesian networks more compactly by reducing sub-networks into single nodes. In such cases the arrows between nodes in an object-oriented Bayesian network are interpreted very differently to arrows between nodes in a recursive Bayesian network, and issues such as causal, Markov and probabilistic consistency do not arise in the former formalism.

[20] [Koller & Pfeffer, 1997].
[21] See [Neil *et al.*, 2000] for example.

Hierarchical Bayesian networks (HBNs) were developed as a way to allow nodes in a Bayesian network to contain arbitrary lower-level structure.[22] Thus recursive Bayesian networks can be viewed as one kind of HBN, in which lower-level structures are of the same type as higher-level structures, namely Bayesian network structures. In fact, HBNs were developed along quite similar lines to recursive Bayesian networks, and even have a concept of flattening. However, there are a number of important differences. As mentioned, HBNs are rather more general in that they allow arbitrary structure. It is questionable whether this extra generality can be motivated by causal considerations: certainly HBNs seem to have been developed in order to achieve extra generality, while recursive Bayesian networks were created in order to model an important class of causal claims. HBNs have been developed in most detail in the case considered in this paper, namely where lower-level structure corresponds to causal connections. However, the lower-level structures are not exactly Bayesian networks in HBNs: one must specify the probability of each variable conditional on its parents in its local graph *and all variables higher up the hierarchy*. Thus HBNs have much larger size complexity than recursive Bayesian networks. HBNs do not adopt our recursive Markov condition—they only assume that a variable is probabilistically independent of all nodes that are not its descendants conditional on its parents and *all higher-level variables*. This has its advantages and its disadvantages: on the one hand it is a weaker assumption and thus less open to question, on the other it leads to the larger size of HBNs. Finally, variables can only appear once in a HBN, but they can appear more than once in a recursive Bayesian network—we would argue that repeated variables are well-motivated in terms of recursive causality (Section 1). Thus HBNs are more restrictive than recursive Bayesian networks in one respect, and more general in another, and have quite different probabilistic structure. However, they share common ground too, and where one formalism is inappropriate, the other might well be applicable.

7 Structural Equation Models

Of course, a Bayesian network is not the only type of causal model, and the extension of Bayesian networks to recursive Bayesian networks can be paralleled in other types of causal model.

After Bayesian networks, perhaps the most widely applied type of causal model is the *structural equation model*. This consists of a 'pseudo-deterministic' equation determining the value of each effect as a function of the values of its direct causes and an error variable:

$$V_i = f(Par_i, \varepsilon_i),$$

[22] [Gyftodimos & Flach, 2002].

for $i = 1, \ldots, n$ and where Par_i is the set of direct causes of V_i and each error variable ε_i is independently distributed. Typically the function f will be linear. The effect is always written on the left-hand side of the equation; by adopting this convention one can determine the causal structure (in the shape of a causal graph) from the set of equations. Structural equations are quite restrictive—they only allow variables to vary with their direct causes (and independent error variables)—and one can prove that the causal Markov condition holds given this restriction. If we specify the probability distribution of each root variable (the variables which have no causes) then we have a Bayesian network, since a structural equation determines the probability distribution of each non-root variable conditional on its parents in the causal graph. A Bayesian network does not determine pseudo-deterministic functional relationships however, and so a structural equation model is a stronger kind of causal model than a Bayesian network.

Structural equation models can be extended to model recursive causality as follows. A *recursive structural equation model* takes not only simple variables as members of its domain, but also *SEM-variables* which take structural equation models as values (including a level 0 variable which takes as its only value the top-level model).[23] As with recursive Bayesian networks we can impose natural consistency conditions on a recursive structural equation model: causal consistency and consistency of functional equations. Given an assignment to the domain, we can create a corresponding, non-recursive structural equation model, its *flattening*, and define a pseudo-deterministic functional model over the whole domain by constructing an equation for each variable as a function of its direct superiors as well as its direct causes (and an error variable).

We see, then, that the move from an ordinary Bayesian network to a recursive Bayesian network can be mirrored in other types of causal model. In the following sections we will study this move from a more general point of view. We will see that the strategy of rendering a general network structure recursive can be applied in various interesting ways—not just to recursive causality.

8 Argumentation Networks

Recursive networks are not just useful for reasoning with causal relationships—they can also be used to reason with other relationships that behave analogously to causality. In this section we shall briefly consider the relation of support between arguments.

In an *argumentation framework*, one considers arguments as relata and attacking as a relation between arguments.[24] Consider the following example.[25] Hal is a

[23]Warning: in the past, acyclic structural equation models have occasionally been called 'recursive structural equation models'—clearly 'recursive' is being used in a different sense here.

[24][Dung, 1995].

[25]Due to [Coleman, 1992] and discussed in [Bench-Capon, 2003] Section 7.

diabetic who loses his insulin; he proceeds to the house of another diabetic, Carla, enters the house and uses some of her insulin. Was Hal justified? The argument (A_1) 'Hal was justified since his life being in danger allowed warranted his drastic measures' is attacked by (A_2) 'it is wrong to break in to another's property' which is in turn attacked by (A_3) 'Hal's subsequently compensating Carla warrants the intrusion'. This argument framework is typically represented by the picture of Figure 25.[26]

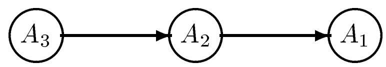


Figure 25. Hal-Carla argumentation framework

One can represent the interplay of arguments at a more fine-grained level by (i) considering propositions as the primary objects of interest, and (ii) taking into account the notion of support as well as that of attack. By taking propositions as nodes and including an arrow from one proposition to another if the former supports or attacks the latter, we can represent an argument graphically. In our example, let C represent Hal compensates Carla', B 'Hal breaks in to Carla's House', W 'Breaking in to a house is wrong' and D 'Hal's life is in danger'. Then we can represent the argument by $[C \longrightarrow^+ B] \longrightarrow^- [W \longrightarrow^- B] \longrightarrow^- [D \longrightarrow^+ B]$ (here a plus indicates support and a minus indicates attack). In general the fine structure of an argument is most naturally represented recursively as a network of arguments and propositions. We call this kind of representation a *recursive argumentation network*.

If a quantitative representation is required, recursive Bayesian networks can be directly applied here. The nodes or variables in the network are either *simple arguments*, i.e. propositions, taking values true or false, or *network arguments*, which take recursive Bayesian networks as values. In our example C is a simple argument with values c or $\neg c$ while A_2 is a network argument with values a_2 referring to $W \longrightarrow B$ (with associated probability specifiers $p(w), p(b|\pm w)$) or $\neg a_2$ representing W, B (with $p(w), p(b)$). Instead of interpreting the arrows as causal relationships, indicating causation or prevention, we interpret them as support relationships, indicating support or attack. The probability $p(v_i|par_i)$ of an assignment v_i to a variable conditional on an assignment par_i to its parents is interpreted as the probability that v_i is *acceptable* given that par_i is *acceptable*. Thus instead of representing support or attack by pluses and minuses, degree of support is represented by conditional probability distributions. If consistency and acyclicity conditions are satisfied, non-local degrees of support can be gleaned from the joint probability distribution defined over all variables.

[26] [Bench-Capon, 2003].

Note that Bench-Capon argues that the evaluation of an argument may depend on accepted values.[27] In our example, the evaluation of the argument depends on whether health is valued more than property, in which case property argument A_2 may not defeat health argument A_1, or vice versa. These value propositions can be modelled explicitly in the network, so that, for example, A_1 depends on value proposition 'health is valued over property' as well as argument A_2.

In sum, relations of support behave analogously to causal relations and arguments are recursive structures; these two observations motivate the use of recursive Bayesian networks to model arguments. This leads us in turn to the question of how to characterise the concept of an abstract recursive network. In the next two sections we explore this question in the context of *input-output* or *information networks*.

9 Self-Fibring Networks: Overview

This section shows how our recursive network approach fits within a more general concept of substituting one network inside another (referred to as self-fibring of networks).

We will focus attention on *information networks*, which are directed acyclic graphs whose roots are *inputs*, whose leaves are *outputs* and whose arrows indicate the flow of information from input to output. Thus if we have

$$\begin{tikzcd} t_1 \arrow[dr] & \\ \vdots & t \\ t_k \arrow[ur] & \end{tikzcd}$$

then we propagate the input from t_i into t. If $\boldsymbol{V}(x)$ is the value at node x, then we need a propagation function f yielding $\boldsymbol{V}(t) = f(\boldsymbol{V}(t_1), \ldots, \boldsymbol{V}(t_n))$. Note that there may be a constraint $\phi(t_1, \ldots, t_n)$ on the inputs: only if a set of value of inputs satisfies ϕ will those values be admissible.

In Bayesian networks, the arrows correspond to causal direction rather than to the flow of information. But an *application* of a Bayesian network can be construed as an information network as follows. When a Bayesian network is applied, the values of a set of variables are observed. These variables are the inputs. They are instantiated to their observed values in the Bayesian network, and this change is propagated around the network, typically using message-passing algorithms,[28]

[27] [Bench-Capon, 2003, Section 5].
[28] [Pearl, 1988].

until the probabilities of further variables of interest (the outputs) can be ascertained. Thus in message-passing algorithms information flows from the inputs to the outputs via the arrows of the Bayesian network, though not normally in accordance with the direction of the arrows in the original Bayesian network. Suppose for instance that Figure 26 is the graph of a Bayesian network, that the value of B is observed and that the probability of C is required. Then in determining the probability of C, information flows from B to C along the pathways between B and C of the original Bayesian network graph, as depicted in Figure 27.

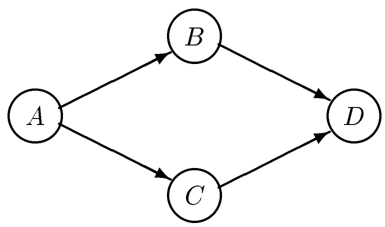


Figure 26. The graph of a Bayesian network.

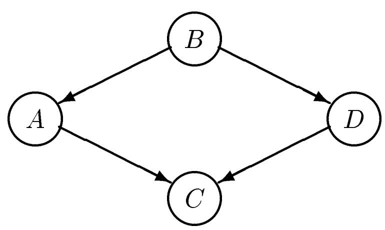


Figure 27. The graph of a corresponding information network.

Note that in general the information network is only a schematic representation of the flow of information: in fact in Bayesian network message-passing propagation algorithms, messages are passed in both directions along arrows, two passes are made of the network, and in multiply connected graphs such as Figure 26 propagation takes place in an associated undirected tree-shaped Markov network formed from the Bayesian network.[29] In singly-connected Bayesian networks though there is a fairly close correspondence between information network and flow of messages.

The question now arises as to how information networks can be *self-fibred*, i.e. substituted one inside the other.

[29]See [Lauritzen & Spiegelhalter, 1988].

There are several options for self fibring. We explain them briefly here and the full definitions come in the next section.

Let $\boldsymbol{B}(X)$ be a network with node X in it. Let $\boldsymbol{A}$ be another network. We want to define $\boldsymbol{C} = \boldsymbol{B}(X/\boldsymbol{A})$, a new network which is the result of substituting $\boldsymbol{A}$ for X.

Already at this stage there are several views to take.

View 1: Syntactical Substitution

Regard the operation at the syntactical level. Define $\boldsymbol{C}$ syntactically and give it meaning / semantics / probabilities derived from the meanings of $\boldsymbol{B}$ and $\boldsymbol{A}$.

View 2: Semantic Insertion

Look at the meaning of $\boldsymbol{B}$ and then define what $\boldsymbol{B}(X/\boldsymbol{A})$ is supposed to be. Here the substitution is not purely syntactic. For example, if $\boldsymbol{B}$ is a Bayesian network where the node X can take two values 0, 1 then if X is 0 we substitute (in a certain way) $\boldsymbol{A}_0$ for X and if X is 1 we substitute $\boldsymbol{A}_1$ (this is the approach taken in Section 3). The "substitution" need not be actual substitution but some operation $\mathit{Ins}(X, \boldsymbol{B}, \boldsymbol{A})$ inserting $\boldsymbol{A}_i$ at the point X inside $\boldsymbol{B}$.

So for example in logic we can have

$$\mathit{Ins}(X, X \to B, C) \stackrel{\text{def}}{=} (X \to C) \to B.$$

Thus in this case

$$\mathit{Ins}(X, \boldsymbol{B}(X), C) = \boldsymbol{B}(X/X \to C).$$

More complex insertions are possible for Bayesian nets. We could convert in the above case the semantic inversion into a syntactic one by splitting each variable Y in the net into two variables Y_0 (for $Y = 0$) and Y_1 (for $Y = 1$). We discuss such manipulations in the next section.

To study our options and to illustrate the ideas of self fibring we begin with a simple two point network

$$A \xrightarrow{f} B$$

The input gives value to A and this is propagated to B, using the function f.

We now give several interpretations for this as implication.

Interpretation 1

The above represents a substructural implication $A \to B$. The semantical interpretation for the substructural $\to$ is via evaluation into an algebraic semigroup $(S, \circ, e)$, where $\circ$ is a binary associative operation and e is the identity.

If the wff $A \to B$ gets value t and the input A get a value a then B gets value $b = t \circ a$.

Here the network function f can be taken as the function

$$\lambda x f_t(x) = \lambda x(t \circ x).$$

Interpretation 2

This interpretation is the modus ponens in a Labelled Deductive System. The rule has the form

$$\frac{\alpha : A,\ \beta : A \to B,\ \varphi(\beta, \alpha)}{\boldsymbol{f}(\beta, \alpha) : B}.$$

Its meaning is that if we prove A with label α and $A \to B$ with label β and (β, α) satisfy the enabling condition φ, then we can deduce B with label $\boldsymbol{f}(\beta, \alpha)$.

The Dempster-Shafer rule is a special case of this. The Dempster-Shafer set up allows for certainty values for $A, B, A \to B$, to be closed intervals of real numbers. Thus if A has value in the real closed interval $[a, b]$ and the implication $A \to B$ has value in the interval $[c, d]$, then B has value in the interval

$$[a, b] \circ [c, d] = \left[\frac{ad + bc - ac}{1 - k}, \frac{bd}{1 - k} \right]$$

with $k = a(1 - d) + c(1 - b)$.

The side condition φ is $\varphi([a, b], [c, d])$ is that $k \neq 1$. Thus to interpret the labelled implication in our network we need to add φ to the link.

$$A \xrightarrow[\boldsymbol{f}_\beta, \varphi]{} B$$

and we have:

$$\lambda x \boldsymbol{f}_\beta(x) = \lambda x \boldsymbol{f}(\beta, x).$$

Interpretation 3

Intuitionistic formulas as types. $A, B, A \to B$ are understood as λ calculus types having λ terms inhabiting them. We read

$$A \xrightarrow[f]{} B$$

as a network, which for any term t of type A, given as input, the network outputs the $f(t)$ term of type B.

Thus f is of type $A \to B$.

Interpretation 4

We can regard

$$A \xrightarrow{f} B$$

as a causal Bayesian network. The variable A can take states $a_1, \ldots, a_k$ and the variable B can take states $b_1, \ldots, b_m$ then the table f must give the conditional probability $P(B|A)$, giving the probability p_{ij} of B being in the state b_j, given A is in the state a_i. We must have $\Sigma_j p_{ij} = 1$.

The matrix is

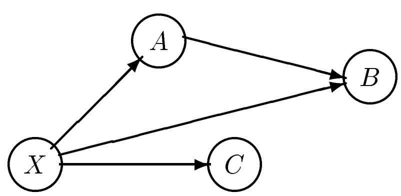
$$P = \begin{pmatrix} p_{11} & p_{1m} \\ \vdots & \vdots \\ p_{k1} & p_{km} \end{pmatrix}$$

If q_i is the probability of A being in the state a_i, $(\Sigma q_i = 1)$ then the probability of B being in the state b_j is $r_j = \Sigma_i p_{ij} q_i$.

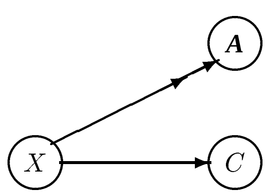
The logical interpretations (1)–(3) allow us to give meaning to self fibred networks where we substitute a network within a network. We have the options here of syntactical substitution (view 1) or semantical insertion (view 2). Our paper chooses semantical insertion, where we have one insertion for $X = 1$ and another for $X = 0$. In the flattening, the insertion makes X a parent to all nodes in the substituted network.

The diagram below shows how this works for $\boldsymbol{B}(X) = X \to C$ and $\boldsymbol{A}_0 = A \to B$.

For the case $X = 0$ we get the network



and for the case $X = 1$, we get



where $X \twoheadrightarrow \boldsymbol{A}$ means that X connects directly to all elements in $\boldsymbol{A}$.

We can adopt a view closer to that of logic and have no insertion if $X = 0$ and yes insertion in case $X = 1$ of a single network. The simplest cases are the following:

$$\Big(A \xrightarrow[f]{} B\Big) \xrightarrow[g]{} C$$

and

$$C \xrightarrow[g]{} \Big(A \xrightarrow[f]{} B\Big).$$

In the first case we took the network

$$X \xrightarrow[g]{} C$$

and substituted for X the network

$$A \xrightarrow[f]{} B$$

The second case is similar.

The question is what meaning do we give to these fibred networks?

Let us consider the first case

$$\Bigg(X = \Big(A \xrightarrow[f]{} B\Big)\Bigg) \xrightarrow[g]{} C$$

The machinery of

$$X \xrightarrow[g]{} C$$

is to accept inputs x of a certain kind at the node X and output $g(x)$ at node C. By letting $X = A \to B$ we must ask: What is the input we are getting for X? We can say the obvious answer is that the input is f. We now have to check whether f is of the kind that can be accepted in our network.

Let us check.

Interpretation 1

Here f can be identified with an element t of the semigroup. So it is OK.

Interpretation 2

Here f can be identified with a label. So again it is OK.

Interpretation 3

Here f is a λ-term of type $A \to B$. To be OK we say X (accepts elements) of type $A \to B$ and g is of type $(A \to B) \to C$. So again we are OK.

Interpretation 4

Here A is a probability distribution for the states of A and f is a matrix of conditional probabilities. We have to deal with that! First let us simplify and say both A, B are two state variables $A = 0, A = 1, B = 0, B = 1$. Even with this simplification, still f is a 2×2 matrix P. It allows for many states not just two. This is not exactly right.

What are our options?

Option 1

Allow for new kinds of inputs for our variables. This option is complicated because of repeated iteration of fibring. We will not pursue it.

Option 2

Extract from the new input (the matrix) a recognisable input for X in $X \to C$, (i.e. a two state input). This method is what we usually do in the area of fibring logics. We need a fibring function ***F*** that will extract two states, yes or no, out of the matrix P. The function is as follows;

- yes if B depends on A in any way
- no if not.

In other words, we read X as a variable getting 1 if the network substituted for it is "on" or "active" and 0 if it is not on.

So for example if the matrix is

$$\begin{pmatrix} p & 1-p \\ p & 1-p \end{pmatrix}$$

we get

$$(q, 1-q) \begin{pmatrix} p & 1-p \\ \\ p & 1-p \end{pmatrix} = (p, 1-p)$$

and thus the probability of B is independent of that of A.

We can talk about the probability of B being independent of A, etc.
Say we have for $0 \leq \varepsilon \leq 1-p$

$$
\begin{aligned}
(q, 1-q)\begin{pmatrix} p & 1-p \\ p+\varepsilon & 1-p-\varepsilon \end{pmatrix} &= \\
(pq + (1-q)p + (1-q)\varepsilon, (1-p)q + (1-q)(1-p) - (1-q)\varepsilon) &= \\
(p + (1-q)\varepsilon, (1-p) - (1-q)\varepsilon). &
\end{aligned}
\tag{2}
$$

The variation is $2\varepsilon(1-q) \leq 2\varepsilon$. So we can give a probability for $\varepsilon = 0$ or $\varepsilon \neq 0$.

We leave this aspect for a moment and discuss the other possibility of fibring, namely

$$C \xrightarrow{g} \left(A \xrightarrow{f} B\right).$$

Here we substitute a network

$$A \xrightarrow{f} B$$

for the variable Y in

$$C \xrightarrow{g} Y$$

The first three interpretations will cope with this very well, because the output of g can modify the f, as they are of the same kind. Can we do something similar in the probabilities case? We again have several options:

1. We can read Y as a variable getting $0, 1$ values indicating whether the network

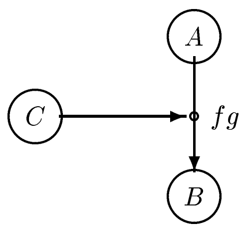
$$A \xrightarrow{f} B$$

is *on* or not. The value of Y is obtained in the network

$$C \xrightarrow{g} Y$$

2. The second option is to use the network up to Y to modify the network which we substitute for Y.[30]

We note that g is a matrix and so is f. Should we modify f by multiplying it by g and set something like



How would this relate to the network:

$$C \xrightarrow{g} A \xrightarrow{f} B$$

Let us check

$$A \xrightarrow{f} B$$

has the matrix

$$\begin{pmatrix} p_1, & 1-p_1 \\ p_2, & 1-p_2 \end{pmatrix}$$

and

[30] In case of neural networks this is the more reasonable option.

$$C \xrightarrow{g} Y$$

has the matrix

$$\begin{pmatrix} \gamma_1, & 1-\gamma_1 \\ \gamma_2, & 1-\gamma_2 \end{pmatrix}$$

gf is

$$\begin{pmatrix} \gamma_1 & 1-\gamma_1 \\ \gamma_2 & 1-\gamma_2 \end{pmatrix} \cdot \begin{pmatrix} P_1 & 1-P_1 \\ P_2 & 1-P_2 \end{pmatrix} =$$

(3)

$$\begin{pmatrix} \gamma_1 p_1 + (1-\gamma_1)p_2, & \gamma_1(1-p_1) + (1-\gamma_1)(1-p_2) \\ \gamma_2 1 p_1 + (1-\gamma_2)p_2, & \gamma_2(1-p_1) + (1-\gamma_2)(1-p_2) \end{pmatrix}$$

This would interpret

$$C \xrightarrow{g} \left(A \xrightarrow{f} B \right).$$

as

$$C \xrightarrow{g} A \xrightarrow{f} B$$

We prefer the first option.

Let us now see what to do with networks of the form

$$\begin{tikzcd} A \arrow[dr] & \\ & C \\ B \arrow[ur] & \end{tikzcd}$$

Can we read the $\to$ as implication? The answer is yes for the first three "logical" interpretations. We read it as

$$\langle A, B\rangle \to C$$

or

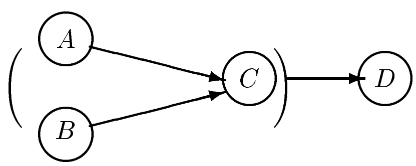
$$A \otimes B \to C$$

where $\otimes$ is a commutative binary operation. It is the multiplicative conjunction in linear logic and is the ordinary conjunction in intuitionistic logic. We have in case of logic that:[31]

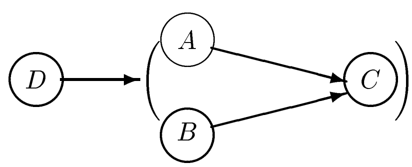
$$(A \otimes B \to C) \equiv (A \to (B \to C)) \equiv (B \to (A \to C))$$

This does not hold in the Bayesian network case. As defined in Section 2 we need a function giving a probability value for C, for each pair of possible values (x, y) for $A, B)$.

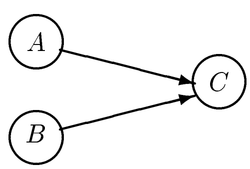
We still need to give meaning to



and

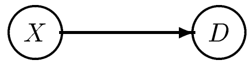


The first is obtained by substituting the network

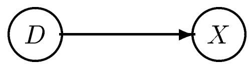


[31]For the Dempster–Shafer rule we calculate $[a, b] \otimes [c, d]$ as $[a, b] \circ [c, d]$.

for X in

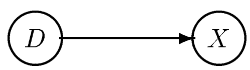


and the other in



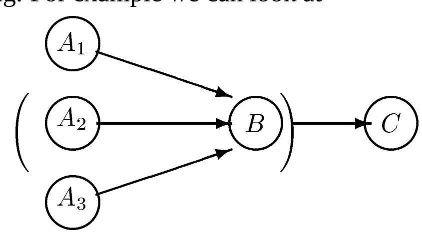
The principles we discovered still hold. In the first case the fibring function gives X values $0, 1$ depending whether we believe in the connection between A, B, C. i.e. the network is “on” or not.

The second case would require modifying the network of A, B, C by using the network

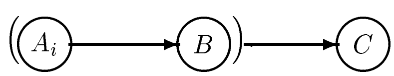


The simplest is to take option 1. The value $x = 1$ means the network with A, B, C is “on” and otherwise it is not.

In any case, the kind of choices we have to make are clear! There is a lot of scope for fine tuning. For example we can look at



as a family of networks of the form



using the probabilities in the substituted network (fix A_j, $j \neq i$ as 0,1) to decide on priorities.

10 Self-Fibring Networks: Theory

The aim of this section is to present a general theory of networks and fibring of networks, in order to put our work into perspective. We begin with an example. Figure 28 shows a typical network.

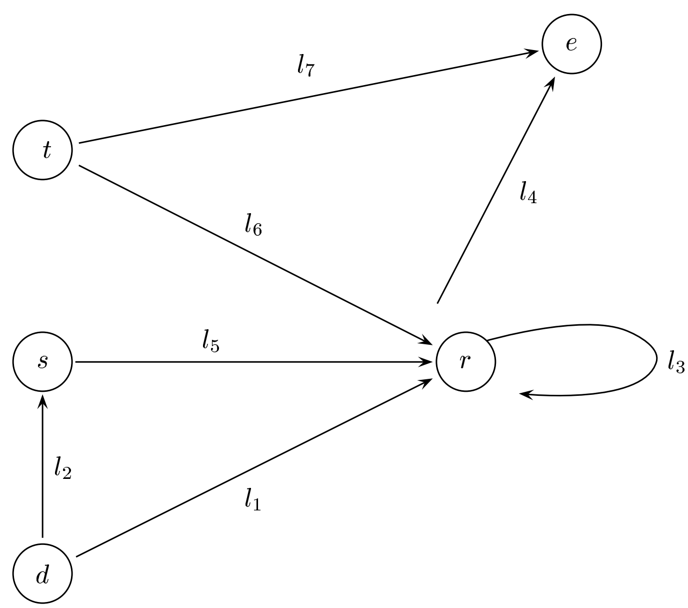


Figure 28.

The nodes of the network is the set $S = \{d, s, t, r, e\}$. We consider d as the input point and e as the output point. The arrows represent connections between nodes. This is a binary relation $R \subseteq S^2$. In Figure 28 we have $R = \{(d, r), (s, r), (t, r), (t, e), (r, e), (r, r), (d, s)\}$. The labels l decorate the connections. So let τ be a function from R into a set of labels L. In Figure 28 we have $L = \{l_1, \ldots, l_7\}$ and $\tau((d, r)) = l_1, \ldots, \tau((t, e)) = l_7$. In addition the nodes are coloured by a colouring function (values which we call colours) giving values in some space Ω. Thus $V(t), V(d), \ldots$ are the colours of the nodes. In Bayesian nets for example, $V(t)$ is a probability.

We also require a family of propagation functions $\mathbb{F}$, giving values in the space

Ω of the form below and such that the following recursive equation holds:

$$V(t) = \mathbb{F}(t, (\tau_1, V(x_1)), (\tau_2, V(x_2)), \ldots, (\tau_n, V(x_n)))$$

where $x_1, \ldots, x_n$ are all the parents of t (i.e. all x_i such that $(x_i, t) \in R$), τ_i is the label of (x_i, t) and $V(x_i)$ are the colours of the nodes x_i, respectively. Note that $\mathbb{F}$ can operate on any number of variables, i.e. n can be arbitrary.

We perceive the colouring to propagate along the network using the arrows, the labels and the function $\mathbb{F}$. In case the network has cycles, we expect V to be implicitly defined by $\mathbb{F}$.

The network in Figure 28 can be interpreted in several ways:

1. It can be interpreted as a map, where the nodes are towns, the labels are distances and the colours are some heuristic numbers to aid some search function(e.g. the labels can give the aerial distance from a central point). We may require the graph to be acyclic. The function $\mathbb{F}$ can give the average distance of the parent nodes from the current node.

2. The network can be Bayesian, in which case we require it to be acyclic. We also require any point $t \neq d$ to either have a parent $\neq d$ (i.e. for some x, $(x, t) \in R, x \neq d$) or to have d alone as a parent. We forbid d itself to have parents.

   Thus d is a dummy point $(d = \top)$ showing the nodes without parents in the rest of the network. The function $\mathbb{F}$ would be the conditional probabilities of a node on its parents.

3. The network can be a neural net with τ, V different weights on the nodes and connections, and $\mathbb{F}$ some meaningful averaging function.

4. The network can be describing a flow problem with τ giving capacities, V giving retention and $\mathbb{F}$ is the obvious function summing up the flow.

We now give a formal definition of a network.

DEFINITION 6. A network has the form

$$\mathcal{N} = (S, R, d, e, \tau, V, L, \mathbb{F}, \Omega)$$

where S is the set of nodes. $d, e \in S$ are the input and output nodes. (We may have several i.e. d_i, e_j.) τ is a labelling function $\tau : R \mapsto L$. L is a set of labels, V is a colouring function on S (range of V is in Ω) and $\mathbb{F}$ is a function giving some value in the space Ω to any finite list of the form $(t, (V_1, l_1), \ldots, (V_n, l_n))$ and the following is required to hold for any $t \in S$ and x_i such that $(x_i, t) \in R$.

- $V(t) = \mathbb{F}(t, (V(x_1), \tau((x_1, t)), \ldots, (V(x_n), \tau(x_n, t))))$ where x_i are *all* the parents of t.

DEFINITION 7 (Fibring function). Let $\mathbb{F}$ be a propagation family for states S and labels L with values in the space Ω. Then a function **F** giving a new propagation function for any triple $(V, l_i, \mathbb{F}), i = 1, \ldots, n$ is called a fibring function. We write **F** as

$$\mathbf{F} : (V, l_i, \mathbb{F}) \mapsto \mathbb{F}_{V,l_i}.$$

so **F** is defined for any set of labels.

Note that n is variable.

We now need to make some distinctions about fibring of network within networks. We give some additional examples.

EXAMPLE 8 (Refinement). Consider a network which is a map

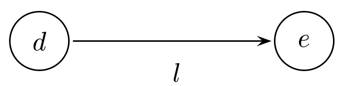


Figure 29.

So d may be Durham and e is Edinburgh. Say the label l is the number of heavy trucks per day one can push through from d to e. We can try and define this map by putting in for e another network, say E which is the map on Edinburgh. This is substituting the actual sorting networks in the UK.

A third simpler example is when d and e are days and we can refine them into hours, see Figure 30

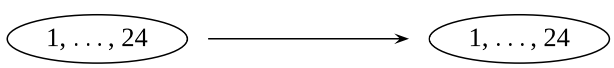


Figure 30.

EXAMPLE 9 (Cut rule in logic). We get fibring/substitution of networks when we consider versions of the cut rule in Labelled Deductive Systems. We give a simple case. Assume our data is a list of formulas, and our language contains $\rightarrow$ only. Thus for example, we may have the list of Figure 31. We can perform modus ponens between any $X \rightarrow Y$ and X, provided X is immediately to its right and the result Y *replaces* $(X \rightarrow Y, X)$ in the list. This way of doing modus ponens

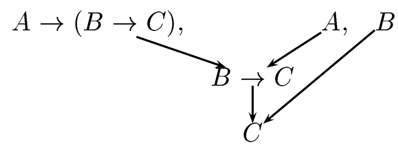


Figure 31.

characterises the one arrow *Lambek Calculus*. How would cut work? Suppose we have a proof of A in Figure 32

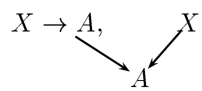


Figure 32.

We can simply substitute the sequence or net for A to get Figure 33

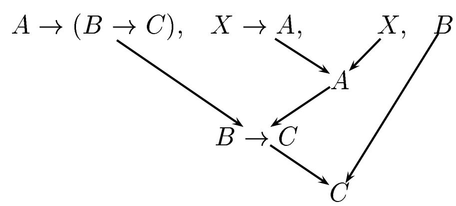


Figure 33.

Suppose now that $X \to Y$ means a version of strict implication.

- If X holds next day then Y holds next day.

The sequence

$$A \to (B \to C), A, B$$

can still be reduced to C but we must keep count of the days.

Consider

$$C \to E, A \to (B \to C), A, B$$

We can get Figure 34 in the Lambek Calculus.

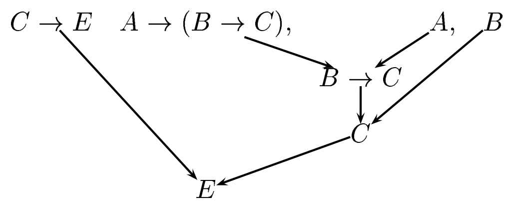


Figure 34.

This will not work in the modal strict implication meaning of $\to$ because we must follow Figure 35:

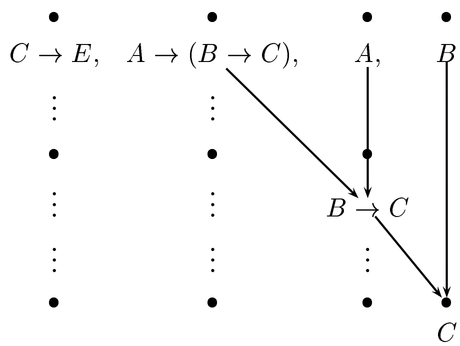


Figure 35.

$C \to E$ is 3 days away from C. We need something like $\top \to (\top \to (C \to E))$.

Thus the network substitution of $(X \to A, X)$ into $(A \to (B \to C), A, B)$ should be different in the strict implication case.

It should give the result in Figure 36

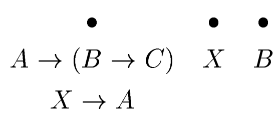


Figure 36.

To summarise: $(X \to Y, X)$ is replaced by (Y) in the Lambek Calculus and is replaced by $(\cdot, Y)$ in the strict implication logic.

Thus the network substitutions corresponding to these logics are as follows:

Let

$$N_1 = (x_1, \ldots, x_n, y, z_1, \ldots, z_m)$$
$$N_2 = (u_1, \ldots, u_k)$$

then the *Lambek substitution* is:

$$N_1(y/N_2) = (x_1, \ldots, x_n, u_1, \ldots, u_k, z_1, \ldots, z_m).$$

and the *Strict substitution* is
(when $k = n + 1$)

$$N_1(y/N_2) = (x_1 \wedge u_1, \ldots, x_n \wedge u_n, u_{n+1}, z_1, \ldots, z_m)$$

If $k \neq n$, add $\top$s to the beginning of the shorter one to make them equal and then substitute.

The moral of this example is that if the networks represent some logic, then options for fibring networks represent options for the cut rule in the logic.

REMARK 10. We need to prepare the ground for the general definition of fibring which will follow.

Assume $\mathcal{N}_1$ and $\mathcal{N}_2$ are as in Figures 37 and 38

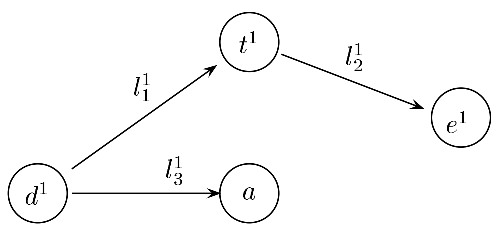


Figure 37.

We have several options in substituting $\mathcal{N}_2$ for t^1, using a fibring function **F**. The most straightforward one is to replace t^1 by $\mathcal{N}_2$ and redirect all arrows coming into t^1 and connect them to all input points d_j^2 of $\mathcal{N}_2$. Similarly all arrows coming out of t^1 will now come out of every output point e_j^2 of $\mathcal{N}_2$.

Figure 39 shows the result.

The function $\mathbb{F}^{1,2}$ is the same as $\mathbb{F}^1$ on nodes from $\mathcal{N}_1$ and is the fibred function $\mathbb{F}^2_{V^1_{(t^1)}, l_1^1, l_2^1}$ obtained by applying **F** to $\mathbb{F}^2$.

Variations can be obtained by changing $\mathbb{F}$ and/or by changing the input output points or $\mathcal{N}_2$ before fibring. So this is quite a general definition. The basic idea is

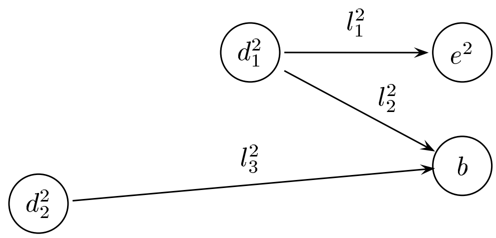


Figure 38.

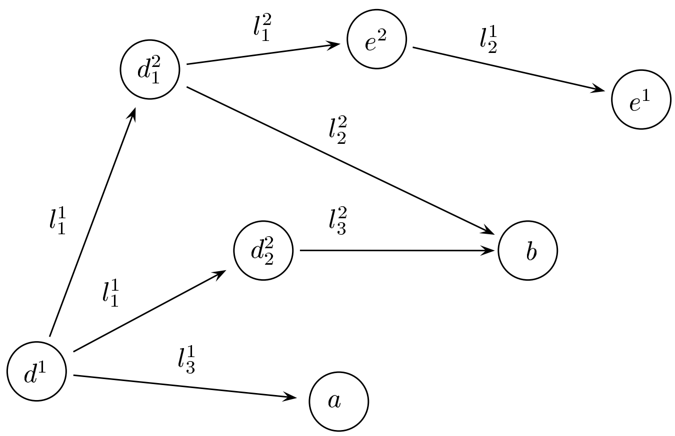
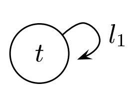


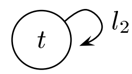
Figure 39.

that the 'environment' of t^1 (namely V^1/t^1) and all labels of connections leading into and out of t^1) change the fibring function $\mathbb{F}^2$ of the substituted network $\mathcal{N}_2$ into $\mathbf{F}(\mathbb{F}^2)$.

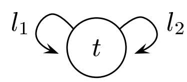
Problems may arise if either $\mathcal{N}_1$ and $\mathcal{N}_2$ have nodes in common or if t^1 is connected to itself. This can cause more than one arrow to occur between two points. For this reason these situations are excluded. To see why this can happen, imagine we substitute



into



where t is both the input and output points. We get by definition the network



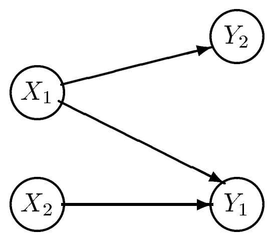
NOTATION 11. If $\mathcal{N}_1, \mathcal{N}_2$ is described using $\rightarrow$ and t^1 is in $\mathcal{N}_1$, we can indicate fibring by substituting $\mathcal{N}_2$ for t^1 and using $\twoheadrightarrow$ to connect into and out of $\mathcal{N}_2$.

The fibring function **F** is suppressed.

Here is an example for our notation:

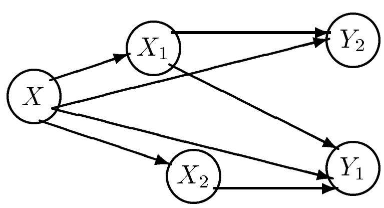
EXAMPLE 12. Example of network using $\rightarrow$:

A =



We use $\twoheadrightarrow$ as a special connection between a node X and a network ***A***.

If ***A*** is the network above and $\twoheadrightarrow$ means that we $\rightarrow$ connect with every node in the network than $X \twoheadrightarrow$ ***A*** means in this case



Arrows coming out of **A** into Y are not drawn.

We now conclude with a general definition:

DEFINITION 13. Let $\mathcal{N}_i = (S^i, R^i, d^i_j, e^i_k, \tau^i, V^i, L^i, \mathbb{F}^i, \Omega^i)$ for $i = 1, 2$, be several networks based on the same set S of nodes (i.e. $S^i \subseteq S$) and sets L and

Ω (i.e. $L^i \subseteq L, \Omega^i \subseteq \Omega$). Assume $S^1 \cap S^2 = \varnothing$. Let **F** be a fibring function and let $t^1 \in S^1$ be a node such that $(t^1, t^2) \notin R^1$ and let l^1_j be all the labels of nodes in $\mathcal{N}_1$ leading into or coming out of t^1. We define the one step fibred system $\mathcal{N}_{1,2} = \mathcal{N}_1(t^1/\mathcal{N}_2)$ as follows:

1. $S^{1,2} = (S^1 \cup S^2) - \{t^1\}$.
2. $R^{1,2} = (R^1 \cup R^2 \cup \{(x, d^2_i) | (x, t^1) \in R^1\} \cup \{(e^2_j, y) | (t^1, y) \in R^1\}) - \{(x, y) \in R^1 | x = t^1$ or $y = t^1\}$.
3. $\{d^{1,2}_i\} = \{d^1_i | t^1 \neq d^1_i\} \cup \{d^2_k | t^1$ is an input point$\}$.
4. $\{e^{1,2}_j\} = \{e^1_j | t^1 \neq e^1_j\} \cup \{e^2_k | t^1$ is an output point$\}$.
5. $\tau^{1,2}((x, y)) =$ • $\tau^1((x, y))$ if $x, y \in S^1$ and $(x, y) \in R^{1,2}$
   - $\tau^2((x, y))$ if $(x, y) \in R^{1,2}$ and $x, y \in S^2$
   - $\tau^1((x, t^1))$ if $x \in S^1, y \in S^2, (x, y) \in R^{1,2}$
   - $\tau^1((t^1, y))$ if $x \in S^2, y \in S^1$ and $(x, y) \in R^{1,2}$
6. $\Omega^{1,2} = \Omega^1 \cup \Omega^2$
7. $L^{1,2} = L^1 \cup L^2$
8. $\mathbb{F}^{1,2}$ is defined as $\mathbf{F}(V^1(t^1), l^1_i, \mathbb{F}^2)$ where l_i are all the labels from other nodes in S^1 leading to t^1 and labels of nodes in S^1 into which t^1 leads. We assume **F** is such that $\mathbb{F}^{1,2} = \mathbb{F}^1$ on points in S^1. This is possible since we assumed $S^1 \cap S^2 = \varnothing$.

11 Conclusion

The moral of this paper is this: recursive structures are rife and benefit from explicit modelling. If causality is to be modelled then it is not always enough to rely on Bayesian networks, for these fail to model recursive structure. Recursive Bayesian networks can be used, however, and these admit joint distributions just as do non-recursive Bayesian networks. Analogously structural equation models can be extended to recursive structural equation models. Recursive Bayesian networks can also be applied to non-causal domains, such as argumentation. A very general type of recursive input-output network, called a self-fibred information network, extends these models and admits interesting applications in logic, where arrows are interpreted as implication.[32,33]

[32] Further results on argumentation networks and on fibring neural networks can be found in [Gabbay & Woods, 2003], [Barringer *et al.*, 2003] and [Garcez *et al.*, 2003].

[33] We thank David Glass for many helpful comments.

BIBLIOGRAPHY

[Barringer *et al.*, 2003] H. Barringer, D. Gabbay & J. Woods. Temporal dynamics of argumentation networks, draft.

[Bench-Capon, 2003] T.J.M. Bench-Capon. Persuasion in practical argument using value based argumentation frameworks. *Journal of Logic and Computation*, **13**, 429–448, 2003.

[Coleman, 1992] J. Coleman. *Risks and Wrongs*. Cambridge University Press, 1992.

[Corfield & Williamson, 2001] D. Corfield & J. Williamson, eds. *Foundations of Bayesianism*. Kluwer Applied Logic Series, Dordrecht: Kluwer Academic Publishers, 2001.

[Dung, 1995] Phan Minh Dung. On the acceptability of arguments and its fundamental role in non-monotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, **77**, 321–357, 1995.

[Gabbay & Woods, 2003] D. Gabbay & J. Woods. The laws of evidence and labelled deduction, *Phinews*, October, 5–46, 2003.

[Garcez *et al.*, 2003] A. Garcez, L. Lamb & D. Gabbay. Fibring of neural nets, draft.

[Gyftodimos & Flach, 2002] E. Gyftodimos & P. Flach. Hierarchical Bayesian Networks: A Probabilistic Reasoning Model for Structured Domains. In E. de Jong & T. Oates, eds. *Proceedings of the International ConferenceML-2002 Workshop on Development of Representations*, pp. 23–30. University of New South Wales, 2002.

[Jaeger, 2002] M. Jaeger. Complex probabilistic modeling with recursive relational Bayesian networks. *Annals of Mathematics and Artificial Intelligence*, to appear.

[Jaynes, 1957] E.T. Jaynes. Information theory and statistical mechanics. *The Physical Review*, **106**, 620–630, 1957.

[Koller & Pfeffer, 1997] D. Koller & A. Pfeffer. Object-oriented Bayesian networks. In *Proceedings of the Thirteenth Annual Conference on Uncertainty in Artificial Intelligence*, pp. 302–313, 1997.

[Lauritzen & Spiegelhalter, 1988] S.L. Lauritzen & D.J. Spiegelhalter. Local computation with probabilities in graphical structures and their applications to expert systems, with discussion. *Journal of the Royal Statistical Society*, **B 50**, 157–254, 1988.

[Mellor, 1995] D.H. Mellor. *The Facts of Causation*. London and New York: Routledge, 1995.

[, Neal 2000] R. M. Neal. On deducing conditional independence from d-separation in causal graphs with feedback. *Journal of Artificial Intelligence Reasearch*, **12**, 87–91, 2000.

[Neqpolitan, 1990] R.E. Neapolitan. *Probabilistic reasoning in expert systems: theory and algorithms*. New York: Wiley, 1990.

[Neil *et al.*, 2000] M. Neil, Norman Fenton & Lars Neilsen. Building large-scale Bayesian networks. *The Knowledge Engineering Review*, **15**, 257–284, 2000.

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

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

[Peña *et al.*, 2002] J.M. Peña, J.A. Lozano & P. Larrañaga. Learning recursive Bayesian multinets for clustering by means of constructive induction. *Machine Learning*, **47**, 63–90, 2002.

[Spirtes, 1995] P. Spirtes. Directed cyclic graphical representation of feedback models. In *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence*, Montreal QU, pp. 491–498. Morgan Kaufmann, 1995.

[Williamson, 2001] Jon Williamson. Foundations for Bayesian networks. In[Corfield & Williamson, 2001, pp. 75–115].

[Williamson, 2002] Jon Williamson. Maximising entropy efficiently. *Electronic Transactions in Artificial Intelligence Journal*, **6**, 2002. www.etaij.org.

Dov Gabbay
Department of Computer Science
King’s College
Strand
London, WC2R 2LS, UK
Email: dg@dcs.kcl.ac.uk

and

Jon Williamson
Department of Philosophy
King’s College
Strand
London, WC2R 2LS, UK
Email: jon.williamson@kcl.ac.uk