Bayesian Networks for Logical Reasoning

Jon Williamson*

Draft of October 29, 2001

Abstract

By identifying and pursuing analogies between causal and logical influence I show how the Bayesian network formalism can be applied to reasoning about logical deductions.

Contents

Despite the fact that deductive logic is concerned with propositions that are certainly true or certainly false, logical reasoning takes place in a context of very little certainty. In fact the very search for a proof of a proposition

*Department of Philosophy, King's College, London. jon.williamson@kcl.ac.uk.

is usually a search for certainty: we are unsure about the proposition and want to become sure by finding a proof or a refutation. Even the search for a *better* proof takes place under uncertainty: we are sure of the conclusion but not of the alternative premises or lemmas.

Uncertainty is rife in mathematics, for instance. A good mathematician is one who can assess which conjectures are likely to be true, and from where a proof of a conjecture is likely to emerge — which hypotheses, intermediary steps and proof techniques are likely to be required and are most plausible in themselves. Mathematics is not a list of theorems but a web of beliefs, and mathematical propositions are constantly being evaluated on the basis of the mathematical and physical evidence available at the time.[1]

Of course logical reasoning has many other applications, notably throughout the field of artificial intelligence. Planning a decision, parsing a sentence, querying a database, checking a computer program, maintaining consistency of a knowledge base and deriving predictions from a model are only few of the tasks that can be considered theorem-proving problems. Finding a proof is rarely an easy matter, thus automated theorem proving and automated proof planning are important areas of active research.[2] However, current systems do not tackle uncertainty in any fundamental way.

I will argue in this paper that Bayesian networks are particularly suited as a formalism for logical reasoning under uncertainty, just as they are for causal reasoning under uncertainty, their more usual domain of application.

The plan is first to describe Bayesian networks in §1.1 and then influence relations in §1.2. Influence relations are important because they permit the application of Bayesian networks: the fact that causality, for example, is an influence relation explains why Bayesian networks can be applied to causal reasoning. In §2 I will argue that logical implication is also an influence relation, and so Bayesian networks can also be applied to logical reasoning. I will then (§3) highlight further analogies between logical and causal Bayesian networks, the presence of which ensure that Bayesian networks offer an *efficient* representation for logical, as well as causal, reasoning. I will go on to show how logical networks can be used to represent probability distributions over clauses in logic programs (§4) and then to generalise the defined notion of logical network (§5), so that logical networks may be applied to problems like proof planning. Finally in §6 I will give an example to indicate the power of the resulting formalism.

[1] This point is made very compellingly by [Corfield 2001].

[2] [Bundy 1999], [Bundy 2001], [Melis 1998], [Richardson & Bundy 1999].

§1
Preliminaries

1.1 Bayesian Networks

A *Bayesian network* consists of two components:

- a directed acyclic graph, or *dag*, $\mathcal{G}$ over variables $C_1, \ldots, C_N$, (which for our purposes we may assume binary-valued, $C_i = v_i \in \{0, 1\}$, and I will abbreviate literal $C_i = 1$ by $+c_i$ or just c_i and literal $C_i = 0$ by $-c_i$ or $\neg c_i$),
- a set of specifying probability values $\mathcal{S} = \{p(c_i|d_i) : d_i \text{ is a state } \pm c_{j_1} \wedge \ldots \wedge \pm c_{j_k} \text{ of the parents } D_i = \{C_{j_1}, \ldots, C_{j_k}\} \text{ of } C_i \text{ in } \mathcal{G},\ i = 1, \ldots, N\}$.[3]

Now, under an independence assumption,[4] namely that given its parents D_i, each node C_i in $\mathcal{G}$ is probabilistically independent of any set S of other nodes not containing its descendants, $p(c_i|d_i \wedge s) = p(c_i|d_i)$, a Bayesian network suffices to determine a probability function p over the variables. This can be determined from the probabilities of the atomic states, which are given by the formula $p(\pm c_1 \wedge \ldots \wedge \pm c_N) = \prod_{i=1}^{N} p(\pm c_i|d_i)$ where the d_i are the parent states consistent with the atomic state. Furthermore, any probability function on the variables can be represented by some Bayesian network.[5]

In particular, if the C_i are *causal* variables, and the graph $\mathcal{G}$ represents the causal relations amongst them, with an arrow from C_i to C_j if C_i is a direct cause of C_j, then it is thought that $\mathcal{G}$ will necessarily be acyclic, and that the independence assumption will be a valid assumption.[6]

A probability function p may also be represented without appealing to a strong assumption like the independence assumption, by directly specifying all the probabilities of the atomic states $p(\pm c_1 \wedge \ldots \wedge \pm c_N)$. However a Bayesian network offers the following advantages over this direct representation. First, depending on the structure of the graph, the number of specifying probabilities in a Bayesian network may be relatively small. For example, if the number of parents of a node is bounded then the number of probabilities required to specify the measure is linear in N. In contrast the direct representation requires the specification of $2^N - 1$ probabilities (there are

[3] If C_i has no parents, $p(c_i|d_i)$ is just $p(c_i)$.

[4] The Bayesian network independence assumption is often called the *Markov* or *causal Markov* condition.

[5] See [Pearl 1988] or [Neapolitan 1990] for more on the formal properties of Bayesian networks.

[6] [Pearl 1988], [Neapolitan 1990].

2^N atomic states and the probability of one of these is determined from the others by additivity). Second, also depending on the structure of the graph, propagation techniques[7] can be employed which allow the quick calculation of conditional probabilities of the form $p(c_i|s)$, where s is a state of other nodes. For example, if the graph is singly-connected (there is at most one path between two nodes) then propagation can be carried out in time linear in N. Thus while in the worst case (which occurs when the graph is *complete*, that is when there is an arrow between any two nodes) there is nothing to be gained by using a Bayesian network, if the graph is of a suitable structure then both the space and time complexity will be dramatically reduced. It is generally presumed that causal graphs are simple enough to offer satisfactory reductions in complexity.

1.2 Influence Relations

The components $\mathcal{G}, \mathcal{S}$, of a Bayesian network may often be thought of as the background knowledge of some rational agent, X say. For example the graph may consist of X's causal knowledge, and the probability specification consist of her corresponding probabilistic knowledge, the probabilities of effects given their direct causes.[8] In this light, the probability function determined by the Bayesian network may be viewed as X's rational belief function, given this background knowledge. Specifically, if the Bayesian network determines that $p(s) = x$ for some atomic state s of $C_1, \ldots, C_N$, then X's degree of belief that s is true ought to be x, given her background knowledge.

But this interpretation only works if the Bayesian network independence assumption holds for rational belief. Furthermore, the question of whether the independence assumption holds depends on the type of knowledge expressed by the graph in the Bayesian network. If the graph is a causal graph, for instance, then it is generally thought that the independence assumption will hold. There is some evidence for this as we shall now see.

Of all the probability functions that X might adopt, some are more rational than others. Objective Bayesianism holds that there is a *most rational* function μ that X should adopt,[9] and that μ is the probability function that (i) is consistent with X's background knowledge $K = (\mathcal{G}, \mathcal{S})$, and (ii) max-

[7] [Neapolitan 1990].

[8] I leave it open as to whether the specified probabilities are objective or degrees of rational belief. If they are objective, I assume that they directly constrain rational belief: if the objective probability of c_i given state d_i of its parents is x then X should believe c_i to degree x, given d_i.

[9] In some circumstances there may be more than one most rational function, in which case X may adopt any one of them.

imises the entropy function $-\sum p(s)\log p(s)$ where the sum is taken over all atomic states s.[10] The reason for this second condition is that the maximum entropy function is the most cautious — it is the function that commits to the background knowledge but as little as possible beyond the background knowledge. For example, if a coin is to be tossed, c_1 signifies a head as outcome and $\neg c_1$ a tail, and X has no background knowledge pertaining to the outcome, then $p(c_1) = p(\neg c_1) = 1/2$ is the belief function that maximises entropy, whereas a least cautious function will commit degree of belief 1 to one of the outcomes and 0 to the other.

According to the first condition the background knowledge $K = (\mathcal{G}, \mathcal{S})$ somehow constrains μ. It is clear how the probabilistic information $\mathcal{S}$ should constrain rational belief: μ should yield the probabilities expressed in $\mathcal{S}$. We shall consider one way in which the graph $\mathcal{G}$ can also constrain μ. Suppose the graph $\mathcal{G}$ expresses X's knowledge about an asymmetric relation R: there is an arrow from C_i to C_j in $\mathcal{G}$ if X knows that $C_i R C_j$. For instance, R might be the relation of direct causality, and $C_i R C_j$ iff C_i is a direct cause of C_j. Define R to be an *influence relation* if the following condition holds:

IRRELEVANCE if $K_1 = (\mathcal{G}_1, \mathcal{S}_1)$ consists of the components of a Bayesian network on nodes $C_1, \ldots, C_n$, and $K_2 = (\mathcal{G}_2, \mathcal{S}_2)$ is the Bayesian network formed by adding an extra node C_{n+1} that is not an ancestor of any of $C_1, \ldots, C_n$, then $\mu_{K_2}^{C_1,\ldots,C_n} = \mu_{K_1}^{C_1,\ldots,C_n}$, i.e. the restriction of the new most rational belief function to the old domain is the same as the old most rational belief function. The new knowledge is irrelevant to beliefs on the old domain.

Causality, for example, is an influence relation: coming to learn that two variables have a common effect should not change one's beliefs in the original variables. In contrast, if one finds out that two variables have a common cause then this alone may be reason to believe the two variables are more dependent than previously thought. Thus it is possible to reason via common causes in a way that is not possible for common effects. By way of example, consider the following situation. Agent X is concerned with two variables L and B signifying lung cancer and bronchitis respectively and initially she has no causal knowledge concerning these variables. Then she learns that smoking S causes each of lung cancer and bronchitis, yielding causal graph Figure 1. One can argue that learning of the existence of common cause S should impact on X's degrees of belief concerning L and B, making them more dependent. The reasoning is as follows: if bronchitis is present, then this may be because the individual is a smoker, and smoking may also have

[10] See [Jaynes 1998].

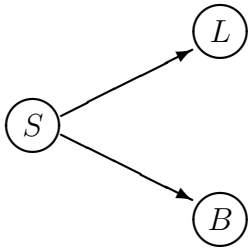

Figure 1: Smoking, lung cancer and bronchitis.

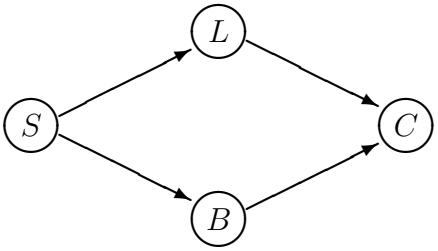

Figure 2: Smoking, lung cancer, bronchitis and chest pains.

caused lung cancer, so X should believe the individual has lung cancer given bronchitis to a greater extent than before — the two variables become more dependent.

Next X learns that both lung cancer and bronchitis cause chest pains C, giving Figure 2. But in this case one can *not* argue that L and B should be rendered more dependent. If the individual has bronchitis then he may well have chest pains, but this does not render lung cancer any more probable because there is already a perfectly good explanation for the chest pains. One cannot reason via a common effect in the same way that one can via a common cause, and the irrelevance condition picks up on this central feature of causality.

It turns out that if $\mathcal{G}$ is knowledge of an influence relation, then the probability function that maximises entropy subject to the constraints imposed by $(\mathcal{G}, \mathcal{S})$ does satisfy the Bayesian network independence assumption. The most rational function is just the probability function determined by the Bayesian network on $(\mathcal{G}, \mathcal{S})$. See [Williamson 2001] for a proof.

This identity validates the rational belief interpretation of Bayesian networks that was mooted at the beginning of this section, and allows us to use Bayesian networks to represent, and perform calculations on, rational belief functions constrained by causal and probabilistic knowledge. But there are other influence relations apart from causality, and it is the aim of this paper

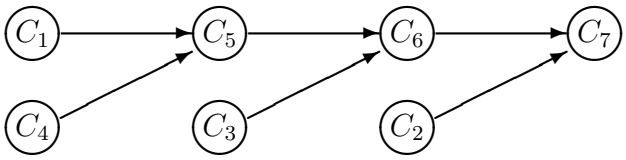

Figure 3: A logical dag.

to show that direct logical implication is an influence relation, and therefore that Bayesian networks can be used to reason about logical as well as causal knowledge.

§2
Logical Networks

A logical proof of a sentence takes the form of an ordered list. Consider a propositional language with sentences $s, t, u, \ldots$ and the following proof of $s \to t, t \to u \vdash s \to u$, using the axiom system of [Mendelson 1964] section 1.4:
1: $t \to u$ [hypothesis]
2: $s \to t$ [hypothesis]
3: $(s \to (t \to u)) \to ((s \to t) \to (s \to u))$ [axiom]
4: $(t \to u) \to (s \to (t \to u))$ [axiom]
5: $s \to (t \to u)$ [by 1, 4]
6: $(s \to t) \to (s \to u)$ [3, 5]
7: $s \to u$ [2, 6]

The important thing to note is that the ordering in a proof defines a directed acyclic graph. If we let C_i signify the sentence on line i, for $i = 1, \ldots, 7$, and deem C_i to be a parent of C_j if C_i is required in the step leading to C_j, we get the dag in Figure 3.

By specifying probabilities of root nodes and conditional probabilities of other nodes given states of their parents, we can form the components of a Bayesian network. These probabilities will depend on the meaning of the sentences rather than simply their syntactic structure, and in this example a specification might start like this: $\mathcal{S} = \{p(c_1) = \frac{3}{4}, p(c_2) = \frac{1}{3}, p(c_3) = 1, p(c_4) = 1, p(c_5|c_1 \wedge c_4) = 1, p(c_5|\neg c_1 \wedge c_4) = \frac{1}{2}, \ldots\}$. Where do these probabilities come from? From the objective Bayesian point of view they should be a rational agent's degrees of belief in the truth of sentences conditional on states of their parents, influenced where possible by objective considerations like known facts, known logical relations and known objective probabilities,

and as cautious as possible about questions not resolved by this background knowledge. In this example the logical axioms have probability 1, but not so the hypotheses.

A causal Bayesian network is sometimes just called a *causal network*. The above example may be called a logical Bayesian network or simply a *logical network*. As in the causal case, if logical implication is an influence relation and the components $\mathcal{G}$ and $\mathcal{S}$ are thought of as background knowledge, then one can use the Bayesian network to represent the most rational belief function that incorporates this knowledge, and to calculate desired probabilities. If for example one learns that C_2, $s \rightarrow t$, is true, then one can propagate probabilities in the Bayesian network to update one's degree of belief $p(c_7|c_2)$ in the conclusion C_7.

In order for logical implication to be an influence relation, learning of a new node that is not a logical influence of any of the other nodes in the network should not change beliefs over the old nodes — the new node must be irrelevant to the old. But this is rather plausible, for a similar reason to the causal case. Suppose X has a logical graph involving two nodes $h(s)$, signifying that Socrates was human, and $\forall x(h(x) \rightarrow m(x))$, signifying that all humans are mortal, and that these nodes have no arrows between them. She has beliefs $p(h(s)) = 0.3$ and $p(\forall x(h(x) \rightarrow m(x))) = 0.9 = p(\forall x(h(x) \rightarrow m(x))|h(s)) = p(\forall x(h(x) \rightarrow m(x))|\neg h(s))$. Later she learns that these two nodes imply $m(s)$, that Socrates is mortal. This new information would not change X's beliefs on the original two nodes: there would be no reason to give a new value to $p(h(s))$, nor to $p(\forall x(h(x) \rightarrow m(x)))$, nor to render the two nodes dependent in any way.[11] On the other hand, if X were to learn that the two original nodes had a new common parent $\forall x(h(x) \wedge m(x))$, then she may well find reason to change her original beliefs. She might render the two original nodes more dependent, for example, by reasoning that if Socrates were human then this might be because all things under consideration are human and mortal, in which case it must be the case that $\forall x(h(x) \rightarrow m(x))$.

Thus direct logical implication is an influence relation. This means that a Bayesian network can be used to represent a rational probability function over the nodes in a logical proof graph, just as it can be used to represent a rational probability function over the nodes in a causal graph. Just as a parent in a causal graph may be called a *causal influence* of its child, so too a parent in a logical graph may be called a *logical influence*. A logical influence need not by itself logically imply its child, but only in conjunction

[11]It is important to note that X learns only of the new node and that it is a child of the two original nodes — she does not learn of the truth or falsity of $m(s)$, which would provide such a reason.

with the child's other parents: one state of the parent nodes logically implies a literal involving the child node. Causal influence and logical influence are both influence relations, but they are not the only influence relations: subsumption of meaning provides another example, where A influences B if a B is a type of A. These influence relations are different concepts in part because they are relations over different types of domains: causality relates physical events, logical influence relates sentences and subsumption of meaning relates words.[12]

I have maintained that the methodology of Bayesian networks may be applied to logical influence, because, like direct causality, direct logical implication is an influence relation. But there are other reasons why this application is effective, and I shall turn to these now.

§3
Effectiveness of the Formalism

A causal Bayesian network offers an efficient representation of a probability function, in the sense that it contains little redundant information. This is partly due to the independence assumption, which only requires that probabilities of nodes given their parent states be provided. But what stops these specified probabilities from being redundant themselves? Might it be possible that further independencies obtain and that a smaller graph in the Bayesian network would yield a more compact representation of the same probability function? No, because causality seems to exclude further independencies: a direct cause C_j increases (or, in the case of prevention, lowers) the probability of its effect C_i, conditional on its other direct causes, $p(c_i|c_j \wedge s) > p(c_i|\neg c_j \wedge s)$ for some state s of the other direct causes of C_i (with the reverse inequality in the case of prevention). We may call this the *dependence principle*: it says that no parent is probabilistically independent of its child, conditional on the other parents.[13] Now an arrow is required in the causal graph from C_j to C_i in order to prevent the Bayesian network from implying an independency where there is this dependency. Thus the fact that causality satisfies the dependence principle explains why the arrows in a causal network (and the corresponding probability specifiers) are not redundant.

[12]Some terminology: when we are dealing with an influence relation a child of an influence may be called an *effluence* (generalising the causal notion of effect), a common effluence of two influences is a *confluence* (generalising common effect) and a common influence of two effluences is a *disfluence* (generalising common cause).

[13]See [Williamson 1999] for a justification of the dependence principle.

We have seen that logical influence is analogous to causal influence because they are both influence relations, and that this fact explains why the Bayesian network independence assumption holds. But the analogy extends further because the dependence principle also carries over to logical influence. Since one state d_i of the parents of C_i logically implies one literal involving C_i, c_i say, each of the parent nodes is an influence in this implication and raises the probability of c_i, as follows. For parent node C_j, either c_j or $\neg c_j$ is in d_i. If the former, then decomposing d_i as $c_j \wedge s$ we have that $p(c_i|c_j \wedge s) = 1$. If $p(c_i|\neg c_j \wedge s) = 1$ too, then C_j is redundant in the proof of C_i from its parents and is not after all a logical influence, contrary to our assumption.[14] Hence $p(c_i|c_j \wedge s) > p(c_i|\neg c_j \wedge s)$. The same is true but with the reverse inequality if $\neg c_j$ occurs in d_i. Therefore the dependence principle carries over to logical networks: each node C_i is probabilistically dependent on each of its logical influences C_j, conditional on the other logical influences of C_i.

The dependence principle explains why information in a logical network is not redundant, but we require more, namely that logical networks be computationally tractable.

Recall that both the space complexity of a Bayesian network representation and the time complexity of propagation algorithms depend on the structure of the graph in the Bayesian network. Sparse graphs lead to lower complexity in the sense that, roughly speaking, fewer parents lead to lower space complexity and fewer connections between nodes lead to lower time complexity. Bayesian networks are thought to be useful for causal reasoning just because, it is thought, causal graphs are normally sparse.

But logical graphs are sparse too. The maximum number of parents is dictated by the maximum number of premises utilised by a rule of inference of the logic in question, and this is usually small. In Mendelson's propositional logic for example, the only rule of inference is modus ponens, which accepts two premises, and so a node in such a logical graph will either have no parents (if it is an axiom or hypothesis) or two parents (if it is the result of applying modus ponens). Likewise, the connectivity in a logical graph tends to be low. A graph will be multiply connected only to the extent that a sentence is used more than once in the derivation of another sentence. This may happen, but occasionally rather than pathologically.[15]

[14]This assumes that *only* the logical truths have probability 1. This is a common assumption for objective Bayesians to make: once a sentence is awarded probability 1, this probability cannot be changed by Bayesian conditionalisation, and so to be cautious and undogmatic a rational agent should only award probability 1 to those sentences that could not be false — the logical truths.

[15]One can of course contest this claim by dreaming up a single-axiom logic which requires an application of the axiom for each inference: this logic will yield highly connected graphs.

In sum, while the fact that logical influence is an influence relation explains why Bayesian networks are applicable at all in this context, the dependence principle and the sparsity of proofs explain why Bayesian networks provide an efficient formalism for logical reasoning under uncertainty.

§4
Logic Programming and Logical Networks

Logic programming offers one domain of application. A logical network can be used to represent a probability distribution over clauses in a logic program: the graph in the network can be constructed from proof trees involving the clauses of interest,[16] and one then specifies the probability of each clause conditional on each state of its logical influences. By way of example, consider the following definite logic program:[17]

C_1: `proud(X) <- parent(X,Y), newborn(Y).`
C_2: `parent(X,Y) <- father(X,Y).`
C_3: `parent(X,Y) <- mother(X,Y).`
C_4: `father(adam,mary).`
C_5: `newborn(mary).`

If we then query `<- proud(Z)` and use Prolog to find a refutation we get the chain of reasoning depicted in Figure 4, where C_6 and C_7 are the sentences `parent(adam,mary)` and `proud(adam)` respectively. By adding a probability specification we can form a logical network and use this network to calculate probabilities of interest, such as $p(c_6|c_7 \wedge \neg c_5)$, the probability that Adam is a parent of Mary given that he is proud but Mary is not newborn.

Thus given a logic program we can use a logical Bayesian network to define a probability distribution over the clauses in the program. This might be useful, for example, where inductive logic programming yields a logic program from a database and one wants to predict the truth value of an atomic sentence on the basis of known facts which do not themselves prove the sentence or its negation. One might even replace the negation-as-failure of Prolog by some negation-as-low-probability in situations where a closed world assumption is inappropriate.

On the other hand, given a set of sentences that can be written in clausal form, we can construct a logic program representing those sentences and use

In the same way one can dream up awkward highly connected causal scenarios which will not be amenable to Bayesian network treatment. Thus the pathological cases can occur, but there is no indication that they are anything but rare in practice.

[16] [Nilsson & Maluszyński 1990] §9.2 shows how to collect proof trees in Prolog.

[17] This is the example of [Nilsson & Maluszyński 1990] §3.1.

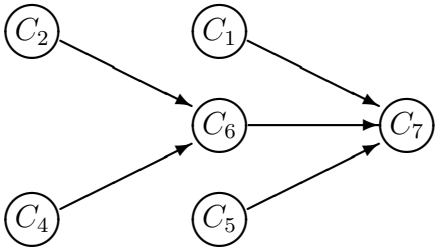

Figure 4: Proof graph from a logic program.

this program to construct a logical network on the original sentences: logic programming can be used as a general tool for finding proof graphs for logical networks.

Stochastic Logic Programming (SLP) also uses proofs of clauses to define a probability distribution over clauses in a logic program,[18] but does so in a rather different way. SLP works by assigning probabilistic labels to the arcs in the proof tree for a goal, multiplying these together to obtain the probability of each derivation in the proof tree, and then summing the probabilities of successful derivations to define the probability of the goal itself being instantiated. The probability of a particular instantiation of the goal is the sum of the probabilities of derivations that yield that instantiation divided by the probability of the goal being instantiated. Thus SLP can be used to define probability distributions that can be broken down as a sum of products (log-linear distributions). Logical networks, on the other hand, ascribe probabilities directly to clauses, and only use proof trees to determine the logical relations amongst the clauses and hence the graphical structure of the network. In SLP the probability of an atom is the proportion of derivations of a more general goal that yield that atom as its instantiation,[19] whereas in a logical network the probability of a clause is the probability that it is true, as a universally quantified sentence. SLP represents a Bayesian network *within* the logic, by means of clauses which describe the graphical structure and probability specification of the corresponding Markov network (formed by linking parents of each node, replacing arrows with undirected arcs, triangulating this graph, and then specifying the marginal probability distributions over cliques in the resulting graph).[20] In contrast a logical Bayesian network over the clauses in a logic program is *external* to the logic program which forms its domain: the probabilities are not part of the logic,

[18] [Cussens 2001] §2.2, [Muggleton 1995].
[19] [Cussens 2001] §2.4.
[20] [Cussens 2001] §2.3.

in the sense that they are not integrated into the logic program as with SLP.

§5
Uncertainty about Structure

Thus far logical networks have been proposed as a mechanism for evaluating sentences within the context of a particular proof. However, just as we are often uncertain as to causal structure, we may not have a perfect idea of what logically implies what. In some situations logic programming can help find a proof, but this is not always possible (if the sentences of interest cannot be written in clausal form) or successful (if the chains of reasoning are too long to be executed in available time, or if the incompleteness of the logic programming system fails to find all the required connections). It would be useful to be able to appeal to probabilistic considerations to help find a proof, and in this section we shall see how we might use logical networks to help plan a proof when faced with uncertainty about logical structure.

The graphs in our logical networks have, up to now, been isomorphic to logical proofs. Several levels of generalisation are possible.

Firstly, not every logical step need be included in a logical network. We may only have a sketch of the key steps of the proof, yet we may be able to form a logical network. Just as a causal graph may represent causality on the macro-scale as well as the micro-scale, so too a logical graph may represent an argument involving large logical steps. At this level of generalisation, some state of parents still logically implies some literal involving their child, but the parents need not be one rule of inference away from their child.

Second, we may not be aware even of all the key steps in the proof, and some of the logical influences on which the proof depends may be left out. Here it may no longer be true that a parent-state logically implies a child-literal. All that can be said is that each parent is involved in the derivation of its child.

In these first two generalisations, the logical graph is a subgraph of the correct proof graph (and therefore the logical graph is guaranteed to remain acyclic). But we can generalise in a third way by allowing logical influences which are not part of the correct proof. We may want to prove a conjecture c from some fixed premise pr, but have two alternative lemmas l_1, l_2, and not know which is involved in the correct proof, as in Figure 5. Here the premise is a genuine logical influence of both l_1 and l_2, which are each logical influences of the conclusion, but the only valid proof of the conclusion from the premise may involve just one of the lemmas, l_1 say. The logical network

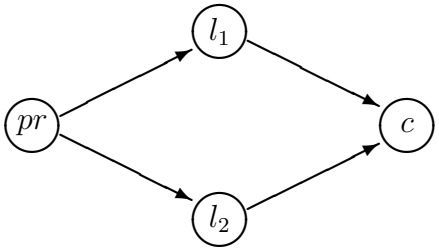

Figure 5: Alternative lemmas l_1 and l_2.

may be used to assess the lemmas by evaluating $p(l_i|pr), p(c|l_i), i = 1, 2$, for example. Hence in this third generalisation we remain in the context of a proof, and the logical graph makes correct assertions of logical influence, but the logical graph contains nodes not in a correct proof.

Note that in all these generalisations we remain in the context of a particular problem. We do not try to form a Bayesian network out of a large body of logical knowledge and then apply this network to individual problems as they arise. The reason for this is two-fold. Firstly, Bayesian networks are built on acyclic graphs and if too much information is included there is a danger of cycles being formed. Secondly, while Bayesian networks often allow an efficient representation and inference mechanism for probability functions, it is still important to keep Bayesian networks small, for practical reasons. A large body of logical knowledge will presumably be multiply-connected, and inference in a Bayesian network will be correspondingly slow.

One other point. Expositions of the theory of rational belief often include a requirement of *logical omniscience*: if set A of sentences logically implies sentence b then $p(b|A) = 1$. Clearly this requirement does not allow for uncertainty of logical structure, and is too strong for practical purposes. A more sensible substitute is: if X's background knowledge contains the fact that A logically implies b then $p(b|A) = 1$. This issue is addressed in [Williamson 1999b].

§6
An Example

Consider the following example. Agent X conjectures that for natural numbers n and $m \neq 0$, there is a unique quotient q and remainder $r < m$ (also natural numbers) such that $n = qm + r$. Let h be this hypothesis, $\forall n, m \neq 0, \exists! q, r < m[n = qm + r]$. X is not very confident in h, giving it

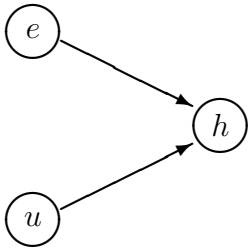

Figure 6: Existence and uniqueness together imply the hypothesis.

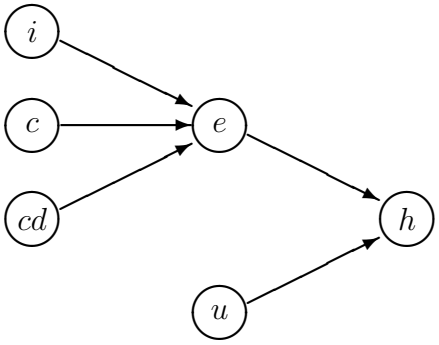

Figure 7: Induction, construction and contradiction added.

degree of belief 0.6. Her initial logical network contains as logical graph the single node h and this specified degree of belief.

X decides to try and prove h. She realises that a good strategy for dealing with a hypothesis of this form is to first prove the existence claim $e : \forall n, m \neq 0, \exists q, r < m[n = qm + r]$, and then prove uniqueness $u : \forall n, m \neq 0, q_1, r_1 < m, q_2, r_2 < m[(n = q_1 m + r_1) \wedge (n = q_2 m + r_2) \rightarrow (q_1 = q_2) \wedge (r_1 = r_2)]$. Figure 6 gives the picture. X decides to write computer programs to generate numbers n, m at random and test these claims and she finds that they are upheld with frequency 1, although X is only 90 percent certain that the uniqueness program is bug-free. X forms a new network with degrees of belief $p(e) = 1, p(u) = 0.9, p(h|e \wedge u) = 1, p(h|e \wedge \neg u) = 0 = p(h|\neg e \wedge u) = p(h|\neg e \wedge \neg u)$. X calculates that $p(h)$ is now 0.9 and so presses ahead with the proof.

X decides to tackle the existence e first, owing to its higher probability. She is familiar with three techniques: mathematical induction i, proof by construction c and proof by contradiction cd. While these are techniques, not sentences, each may be substituted for sentences and so can be included

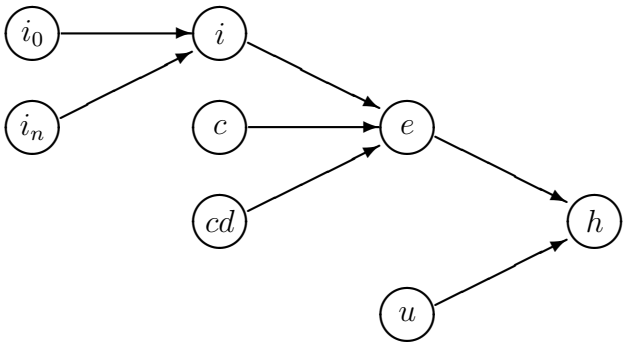

Figure 8: Base case and inductive step added.

as nodes in the graph, as in Figure 7.[21] X believes that induction is 70 percent likely to be successful, construction 30 percent, and contradiction 60 percent. Her new probability specifiers are $p(i) = 0.7, p(c) = 0.3, p(cd) = 0.6, p(e|\neg i \wedge \neg c \wedge \neg cd) = 0.6$, otherwise $p(e|\pm i \wedge \pm c \wedge \pm cd) = 1$ (*each* of the techniques may be used to imply the existence component, whereas the existence and uniqueness components *together* imply the hypothesis). Now the Bayesian net implies that $p(e) = 0.97$ and $p(h) = 0.87$.

X tries induction on n, splitting it into the base case $i_0 : \forall m \neq 0, \exists q, r < m[0 = qm + r]$ and the inductive step $i_n : (\forall m \neq 0, \exists q_1, r_1 < m[n = q_1 m + r_1]) \rightarrow (\forall m \neq 0, \exists q_2, r_2 < m[n + 1 = q_2 m + r_2])$, as in Figure 8, and adds degrees of belief $p(i_0) = .9, p(i_n) = .7, p(i|i_0 \wedge i_1) = 1$.

This gives the general gist of the process. In the next step X would prove i_0 by construction with $q = r = 0$, and go on to tackle the inductive step (the idea here is that $n + 1 = q_1 m + (r_1 + 1)$, and if $r_1 + 1 = m$ then we can write $n + 1 = (q_1 + 1)m + 0$), and then the uniqueness claim.[22]

The key points of the example are that logical structure does not need to be known in advance, and that whether and how to proceed in the proof can be decided on the basis of probabilities. In the example, probability specifiers

[21] In this context induction i may be taken to denote the sentence

$$(\forall m \neq 0, \exists q, r < m[0 = qm + r]) \wedge$$
$$((\forall n, m \neq 0, \exists q_1, r_1 < m[n = q_1 m + r_1]) \rightarrow$$
$$(\exists q_2, r_2 < m[n + 1 = q_2 m + r_2])),$$

and construction and contradiction can be treated similarly.

[22] A final proof would look like that of [Mendelson 1964] proposition 3.11. If the constructive route were taken, the division algorithm might be arrived at — see [Nathanson 2000] theorem 1.1.

were determined from degrees of belief, frequencies in a random experiment, and estimates of frequency of success. Nodes corresponded to both sentences and techniques. This flexibility and expressive power are crucial to the logical network formalism if it is to be applied to realistic problems.

§7
Conclusion

Thus far metalogic has overlooked an account of the way uncertainty guides logical inference. Yet uncertainty must be confronted in any comprehensive approach to practical theorem proving. One aim of this paper has been to introduce probability into the metalanguage and put uncertainty on the agenda. Another aim has been to give a practical formalism for reasoning about deductions. Logical Bayesian networks provide a very general framework which can be applied to important tasks such as conjecture evaluation and proof planning.[23]

References

[Bundy 1999] Alan Bundy: 'A survey of automated deduction', Edinburgh Artificial Intelligence Research Paper 950.

[Bundy 2001] Alan Bundy: 'A critique of proof planning', in [Kakas & Sadri 2001].

[Corfield 2001] David Corfield: 'Bayesianism in mathematics', in [Corfield & Williamson 2001].

[Corfield & Williamson 2001] David Corfield & Jon Williamson(eds.): 'Foundations of Bayesianism', Kluwer Applied Logic Series.

[Cussens 2001] James Cussens: 'Integrating probabilistic and logical reasoning', in [Corfield & Williamson 2001].

[Jaynes 1998] E.T. Jaynes: 'Probability theory: the logic of science', http://bayes.wustl.edu/etj/prob.html.

[Kakas & Sadri 2001] A. Kakas & F. Sadri: 'Essays in honour of Robert Kowalski', to appear.

[23]Thanks to David Chart, David Corfield and Donald Gillies for useful comments, and to the UK Arts and Humanities Research Board for funding this research.

[Melis 1998] Erica Melis: 'AI techniques in proof planning', Proceedings of the 13th European Conference on Artificial Intelligence, John Wiley.

[Mendelson 1964] Elliott Mendelson: 'Introduction to mathematical logic', Chapman & Hall, fourth edition 1997.

[Muggleton 1995] Stephen Muggleton: 'Stochastic logic programs', in L. De Raedt(ed.)[1995]: 'Advances in inductive logic programming', IOS Press.

[Nathanson 2000] Melvyn B. Nathanson: 'Elementary methods in number theory', Springer.

[Neapolitan 1990] Richard E. Neapolitan: 'Probabilistic reasoning in expert systems: theory and algorithms', New York: Wiley.

[Nilsson & Maluszyński 1990] Ulf Nilsson & Jan Maluszyński: 'Logic, programming and prolog', Chichester: John Wiley & Sons, second edition, 1995.

[Pearl 1988] Judea Pearl: 'Probabilistic reasoning in intelligent systems: networks of plausible inference', Morgan Kaufmann.

[Richardson & Bundy 1999] Julian Richardson & Alan Bundy: 'Proof planning methods as schemas', Journal of Symbolic Computation 11.

[Williamson 1999] Jon Williamson: 'Does a cause increase the probability of its effects?', philosophy.ai report pai_jw_99_d, http://www.kcl.ac.uk/philosophy.ai.

[Williamson 1999b] Jon Williamson: 'Logical omniscience and rational belief', philosophy.ai report pai_jw_99_e, http://www.kcl.ac.uk/philosophy.ai.

[Williamson 2001] Jon Williamson: 'Foundations for Bayesian networks', in [Corfield & Williamson 2001].

Index