Expert Systems and Belief Networks

1 Overview

This is an ill-defined and potentially vast subject. Its domain is areas of knowledge where the experts really are much better than others, and where there is not a large base of quantified examples to which pattern recognition methods could be applied. Some mainstream examples are

evaluation of exploration sites for mineral ores. This was one of the first successful examples — the program PROSPECTOR (Duda *et al.*, 1979) helped find in 1983 a deposit of molybdenum in Washington State, and then another in Alberta worth $100 million.

trouble-shooting. The first commercial expert system, originally named R1 and later XCON (McDermott, 1982, Bachant and McDermott, 1984) was used by Digital to check orders and product configurations of VAX computers before shipping. By 1986 a $40 million per year saving was being claimed for XCON.

A modern example (Heckerman and Wellman, 1995, Heckerman *et al.*, 1995) is the EPTS[1] system Microsoft ship with Windows 95.

interpreting mass spectrographs. The DENDRAL expert system (Buchanan *et al.*, 1969, Feigenbaum *et al.*, 1971, Lindsay *et al.*, 1980) helped interpret molecular information from mass spectrographs of organic chemicals.

medical diagnosis. This is the main application area.

The MYCIN system (Shortliffe, 1976; Buchanan and Shortliffe, 1984), was applied to the diagnosis and treatment of around 100 bacterial infections of blood. It had about 450 rules obtained by interviewing experts, and performed "better than junior doctors, as well as some experts." Winston (1992, pp. 130–1) gives an example session.

De Dombal (1980) studied diagnosis of acute abdominal pain — unlike many of the other examples he did have a reasonably large (several hundred) database of cases.

The CHILD (or 'blue baby') system (Spiegelhalter *et al.*, 1993) is a diagnosis system for telephone referrals of newborn babies with (possible) congenital heart disease.

The INTERNIST system (Miller *et al.*, 1982; Pople, 1985) covers about 80% of general medicine, with descriptions of 750 disorders, compiled from the medical literature and interviews with specialists.

One vision of the future (Russell and Norvig, 1995, p. 848) is that it will become essential for doctors to follow the advice expert systems if these become more reliably accurate than humans, since they may otherwise be legally liable. At least in the US, physicians are responsible for understanding the reasoning behind any decision and using their own judgement in deciding whether to use the system's recommendations.

The key features are that there is a knowledge base of information *elicited* from experts and an *inference engine* which uses that knowledge base to answer questions. Some people would require the two to be separate, that is that there be a general-purpose inference engine which can be applied to many different knowledge bases. It is also often essential that the expert system have a power of *explanation*, as a physician has to understand the reasoning used.

This should be contrasted with *pattern recognition*, where we have a database of examples of symptoms and tests on patients, and eventual outcomes (confirmed diagnoses). But the boundaries are slippery, as some expert systems (e.g. CHILD) were trained both by experts and on data, and some pattern recognition databases were produced by watching or interrogating experts. (An example is Claude Sammet's work on systems learning to fly by collecting data from experts on a flight simulator.)

In some domains you do *not* want an expert system: pattern recognition systems for consumer credit are much better than the so-called expert bank managers. Note too that we may not want to mimic the decision-making of experts once we have elicited their expert knowledge, for humans fail to combine information in the optimal way. Expert systems whose aim is rational decision making are called *normative*.

1.1 Some definitions

> *A computer system that operates by applying an inference mechanism to a body of specialist expertise represented in the form of 'knowledge'. [Goodall (1985)]*
>
> *A program intended to make reasoned judgements or give assistance in a complex area in which human skills are fallible or scarce. [Lauritzen and Spiegelhalter (1988, p. 157)]*
>
> *A program designed to solve problems at a level comparable to that of a human expert in a given domain. [Cooper (1989)]*
>
> *To summarize, an expert system has two parts. The first one is the so-called knowledge base. It usually makes up most of the system. In its simplest form it is a list of IF…THEN rules: each one specifies what to do, or what conclusions to draw, under a set of well-defined circumstances. The second part of the expert system often goes under the name of the "shell". As the name implies, it acts as a receptacle for the knowledge base and contains instruments for making efficient use of it. These include*
>
> - *A short-term memory that contains specific data about the actual problem under study.*
> - *Tree-searching as such.*
> - *User interface, which can range from simple, menu-driven interaction with the computer, to quasi natural-language dialog.*
>
> *[Crevier (1993)]*
>
> *When you have finished this chapter, you will understand the key ideas that support many of the useful applications of artificial intelligence. Such applications are often mislabeled* ***expert systems****, even though their problem-solving behavior seems more like that of human novices, rather than of human experts. [Winston (1992)]*

These are rather contradictory: it is perhaps not surprising that Russell and Norvig (1995) offer no definition despite frequent use of the term.

We will ignore the elicitation of the knowledge base, and concentrate on the inference engine, in particular how *uncertainty* might (or should) be handled.

2 Rule-based systems

Rule-based systems have a series of rules, that is a set of conditions (called *antecedents*) which will lead to a conclusion. This is a common way to think in AI, and naturally expressed in an AI programming language such as PROLOG. A toy rule base might be

has feathers AND lays eggs $\implies$ is a bird
has scales AND lives on land AND lays eggs $\implies$ is a reptile
has scales AND lives on water AND lays eggs $\implies$ is a fish
has fur AND drinks milk $\implies$ is a mammal
is viviparous AND drinks milk $\implies$ is a mammal
is a bird AND is flightless AND swims $\implies$ is a penguin
is a bird AND is flightless AND is big $\implies$ is an ostrich
is a mammal AND lives in water AND is big $\implies$ is a whale
is a fish AND is big $\implies$ is a shark
is a reptile AND has no legs $\implies$ is a snake

(Such rules are sometimes called *production rules*.) If we are told that an animal is big, flightless, has feathers and lays eggs we will easily deduce that it is an ostrich. How is this done? Two distinct ways of reasoning:

(a) *backward chaining* or 'goal-driven'. We set out to prove this is an ostrich. We try the rules which imply this (in our case just one rule) and try to prove all the facts in its premise; here that it is a bird, flightless and big. We have still to prove that it is a bird, so we repeat the process for that fact. The literature claims physicians work this way.

Note that we have to guess the answer or try all possible answers, and that we may have to try to an exponentially increasing tree of explanations.

(b) *forward chaining* or 'data-driven', in which we start with what we know and try to prove facts to add to the knowledge base. Here we first prove it is a bird, then that it is an ostrich, by scanning rules to find one that is satisfied.

This is an ancient way of representing knowledge: Crevier (1993, pp. 145f) quotes a medical example from 3000 BC, and most botanical keys are of this form. A real example from DENDRAL (Crevier, 1993, p. 149):

IF the spectrum has two peaks at x_1 and x_2 such that

a. $x_1 + x_2 =$ molecular weight $+ 28$
b. $x_1 - 28$ is a high peak
c. $x_2 - 28$ is a high peak
d. at least one of x_1 or x_2 is high.

THEN the molecule contains a ketone group.

There are a lot of problems with rule-based systems. There are no exceptions, so rules can interact in surprising ways. When XCON reached 10,000 rules it was a full-time job for 150 people to maintain. Rule-based expert systems use monotonic logic, so cannot retract conclusions. A famous example (Minsky, 1982 presidential address to the AAAI):

Human: "All ducks can fly. Charlie is a duck."

Expert System: "Then Charlie can fly."

Human: "But Charlie is dead."

Expert System: "Oh! Then Charlie can't fly!"

There are now a few systems built on non-monotonic logics. This is part of the issue of how to deal with conflict between rules. It is also related to uncertainty: we mean most (live) birds can fly, not that all birds can fly. Similarly, diseases usually but not invariably have particular symptoms. Rule-based systems cannot learn (easily) from their mistakes.

Rule-based systems do provide a powerful, deterministic, explanation. Chapters 7 and 8 of Winston (1992) give a clear introduction with worked examples. Tree-structured rule systems (in which there is only one way to combine rules to reach each conclusion) are widely used, and are often known as *diagnostic keys*. (They dominate botanical classification, for example.)

3 Uncertainty

To us it may be axiomatic that uncertainty should be handled by probability theory, but this has been far from obvious to workers in expert systems, and not just out of ignorance. Thus Pearl (1988) devotes two chapters to the case for probabilities. Only recently with the demonstration that probabilistic expert systems can be made to deliver workable systems (largely by Pearl, Lauritzen & Spiegelhalter, Heckerman and co-workers) has the case for probabilistic reasoning been widely (but not universally) accepted. To see some of the debate browse the collection of papers edited by Shafer and Pearl (1990).

3.1 Certainty factors

One system (Shortliffe and Buchanan, 1975) is so famous that we need to discuss it, even though it is only of historical interest and no longer recommended by one of its inventors (according to Russell and Norvig, 1995, p. 462). Each rule in MYCIN had a conclusion like *there is suggestive evidence that* ... with a *certainty factor* such as 0.6 attached. A CF of $+1$ indicated logical implication, and -1 logical exclusion. The original interpretation was in terms of $P(H \mid E)$ and $P(H)$, where H is the hypothesis 'the conclusion is true' and E is the evidence 'the premise is true'. Then

$$\begin{aligned} CF &= \frac{\mathsf{P}(H \mid E) - \mathsf{P}(H)}{1 - \mathsf{P}(H)} \qquad \text{if} \quad \mathsf{P}(H \mid E) \geq \mathsf{P}(H) \\ &= \frac{\mathsf{P}(H \mid E) - \mathsf{P}(H)}{\mathsf{P}(H)} \qquad \text{if} \quad \mathsf{P}(H \mid E) \leq \mathsf{P}(H) \end{aligned}$$

Note that this is the *interpretation*: the numbers were elicited directly on a score of 1 to 10 (or -1 to -10). If $\mathsf{P}(H \mid E) < \mathsf{P}(H)$ the evidence is against H, and is regarded as for H^c. Thus the experts were allowed to give certainty scores for or against the conclusion (but not both).

The current belief in an assertion is also a number in $[-1, 1]$. The belief β in the premise of a rule with multiple terms is the minimum of the beliefs in each term. If there is negative belief β in an antecedent the rule has no effect: if $\beta > 0$ then the change α in belief in the conclusion becomes $\max(0, \beta) \times CF$. Finally, if two rules share a conclusion then the combined change in belief is

$$\begin{array}{ll} \alpha_1 + \alpha_2(1 - |\alpha_1|) & \text{if} \quad \alpha_1\alpha_2 \geq 0 \\ \dfrac{\alpha_1 + \alpha_2}{1 - \min(|\alpha_1|, |\alpha_2|)} & \text{if} \quad \alpha_1\alpha_2 < 0 \end{array}$$

Backward chaining is used with an initial belief of zero. The rules in MYCIN are diagnostic, not causal, that is symptoms imply diseases. For example

IF
1) the site of the culture is blood, and
2) the Gram stain of the organism is negative, and
3) the morphology of the organism is rod, and
4) the patient is a compromised host

THEN there is suggestive evidence (0.6) that the identity of the organism is *pseudomonas aeruginosa*.

3.2 Fuzzy logic

A *very* controversial subject. Russell and Norvig (1995, p. 463) assert

> 'most authors say that fuzzy set theory is not a method for uncertainty reasoning at all.'

Perhaps (they offer no survey data), but there are vociferous exceptions. For example the FAQ of the newsgroup `comp.ai.fuzzy` says

> 'Fuzzy sets and logic must be viewed as a formal mathematical theory for the representation of uncertainty.'
>
> 'A fuzzy expert system is an expert system that uses a collection of fuzzy membership functions and rules, instead of Boolean logic, to reason about data.'

and that is a collective statement by the fuzzists to outsiders of what their subject is.

Fuzzy set theory assigns degrees of memberships in $[0, 1]$ of objects to sets, which may be unrelated to belief in membership, but are not given an operationally verifiable meaning. Fuzzy sets are then combined by $\min$ rather than intersection, $\max$ rather than union, and membership of $A^c = 1 -$ membership of A. Note then that membership of A or $A^c = \max[m(A), 1 - m(A)] \neq 1$! The reference gives a relatively mild discussion of the grandiose claims for fuzzy logic (which may be implemented in your washing machine or even automatic gearbox). Laviolette *et al.* (1995) give a more detailed critique, with discussion contributions from both sides of the argument.

3.3 Problems with rule-based methods

There are fundamental problems common to all ways to add uncertainty to rule-based systems, including certainty factors and fuzzy logic.

- **locality** In logical systems, if $A \implies B$ then if A holds we know B holds. With uncertainty we need to consider all of the evidence and how it interacts. If we are told a rumour twice, is it greater evidence? It is in the CF theory and is not in fuzzy logic. Hearing a rumour twice is treated in the same way as a careful confirmation of a scientific experiment.
- **Detachment** Once we prove that we have an ostrich, we can use that fact regardless of how we proved it. When handling uncertainties we have to know if we have already used the evidence we have to hand in establishing the antecedent(s), to avoid over-counting.
- **Truth-functionality** We cannot establish the uncertainty in a a complex sentence from the uncertainty of its parts, but we can if we replace 'uncertainty' with 'truth'. Thus rule-based systems must be making stringent independence assumptions.

Consider a much-used example. We notice that the grass is wet. This has two possible causes, that it has rained or that the sprinkler system has been turned on. We can also observe the sky to see if there are rain-clouds around. We will need rules 'wet-grass $\implies$ rain' for diagnosis, and 'rain $\implies$ wet-grass' to reason causally. If we are not careful we get a feedback loop in which each reinforces the other. We also need 'sprinkler $\implies$ wet-grass'. Then if we see that the sprinkler is on, we will increase the certainty in wet-grass and hence that it rained. Thus in a truth-functionality system we cannot avoid deriving the rule 'sprinkler $\implies$ rain' (and its converse). Of course, this is ridiculous — finding that the sprinkler is on should reduce our belief that it has rained.

A similar critique is given by Cooke (1991, pp. 55–62).

3.4 Dempster-Shafer belief functions

This theory is claimed to distinguish between uncertainty and ignorance. There is a difference between knowing we have a fair coin with probability 0.5 of showing heads, and knowing nothing about the coin. Effectively Dempster-Shafer theory assigns an interval in which a probability might lie (here presumably $[0, 1]$) and shows how such probability intervals might be manipulated. The difficulty is relating such intervals to actions. If we have probability intervals on each of a large range of diagnoses, how should we act?

Notwithstanding these operational difficulties, belief functions are still an active subject of research. For some of the debate, see Shafer (1987); Lindley (1987) and their discussion, Pearl (1988, section 9.1) and many of the articles in Shafer and Pearl (1990).

3.5 From trees to rules

Applying C4.5 to the shuttle autolander dataset gave

```
Visibility = no: auto (128.0)
Visibility = yes:
|   Error = XL: noauto (32.0)
|   Error = LX: noauto (32.0)
|   Error = MM:
|   |   Stability = xstab: noauto (16.0)
|   |   Stability = stab:
|   |   |   Sign = nn: noauto (5.0)
|   |   |   Sign = pp:
```

```
|   |   |   |   Magnitude = Light: auto (2.0)
|   |   |   |   Magnitude = Medium: auto (2.0)
|   |   |   |   Magnitude = Strong: noauto (2.0/1.0)
|   |   |   |   Magnitude = Out: noauto (2.0)
|   Error = SS:
|   |   Stability = xstab: noauto (16.0)
|   |   Stability = stab:
|   |   |   Magnitude = Light: auto (4.0)
|   |   |   Magnitude = Medium: auto (4.0)
|   |   |   Magnitude = Strong: auto (4.0)
|   |   |   Magnitude = Out: noauto (4.0)
```

This is converted by the program c4.5rules to the set of rules

```
Rule 8:
        Stability = xstab
        Visibility = yes
        ->  class noauto  [97.9%]
Rule 1:
        Error = XL
        Visibility = yes
        ->  class noauto  [95.8%]
Rule 2:
        Error = LX
        Visibility = yes
        ->  class noauto  [95.8%]
Rule 6:
        Magnitude = Out
        Visibility = yes
        ->  class noauto  [95.8%]
Rule 7:
        Error = MM
        Sign = nn
        Visibility = yes
        ->  class noauto  [89.9%]

Rule 12:
        Visibility = no
        ->  class auto  [98.9%]
Rule 9:
        Stability = stab
        Error = SS
        Magnitude = Light
        ->  class auto  [84.1%]
Rule 10:
        Stability = stab
        Error = SS
        Magnitude = Medium
        ->  class auto  [84.1%]
Rule 11:
        Stability = stab
        Error = SS
        Magnitude = Strong
        ->  class auto  [84.1%]
```

```
Rule 3:
        Stability = stab
        Error = MM
        Sign = pp
        Magnitude = Light
        ->  class auto  [70.7%]
Rule 4:
        Stability = stab
        Error = MM
        Sign = pp
        Magnitude = Medium
        ->  class auto  [70.7%]
```

The process is first to write each route though the (full) tree as a rule. Then for each of the antecedents, a test is done to see if that condition is actually needed. We have a table of the form

	class correct	other classes
satisfies condition	Y_1	E_1
does not satisfy condition	Y_2	E_2

of cases which satisfy the rest of the antecedent. Thus the top row is the number of successes and failures amongst cases satisfying the full antecedent, and the whole table for those satisfying the reduced antecedent. The significance of the table is tested (e.g. by Fisher's exact test) for all antecedent conditions, and the least significant one dropped if any have p-value greater than 0.25. This is repeated until each rule is simplified as far as possible, and so generalized.

This gives a set of rules which overlap: what do we do if more than one 'fires' on a particular case? We choose the one with the highest 'CF', that is the highest expected success rate on cases it covers. Using $Y/(Y+E)$, the apparent error rate, would be optimistic. Quinlan uses a lower 25% confidence limit for the binomial parameter θ which gave Y successes out of $Y+E$ trials.

This is not quite the whole story! There are lots of *ad hoc* schemes for dropping rules.

4 Probabilistic Reasoning Systems

The key to making use of probability theory to represent uncertainty has been to separate the actual numbers from the structure. In a probabilistic theory the saturated model is one in which everything is related to everything else. The 'blue baby' problem has six possible diseases, and six inputs, three binary, two with three categories and one with five. Thus in the saturated model $2^3 \times 3^2 \times 5 \times 6 = 2160$ probabilities would be needed. This is a small diagnostic problem, in a very restricted domain.

The most extreme simplified structure is known as *naïve* or *idiot's Bayes*. (Don't treat these terms as pejorative). This assumes that the inputs X_i give independent information about the diagnosis D, more precisely that the X_i are conditionally independent given D. Thus

$$\mathsf{P}(X_1, \ldots, X_r \,|\, D) = \prod_i \mathsf{P}(X_i \,|\, D)$$

and hence

$$\mathsf{P}(D \,|\, X_1, \ldots, X_r) \propto \mathsf{P}(D) \prod_i \mathsf{P}(X_i \,|\, D)$$

This is intuitive, simple to explain, but not comparable in complexity with domains with chains of dependent rules.

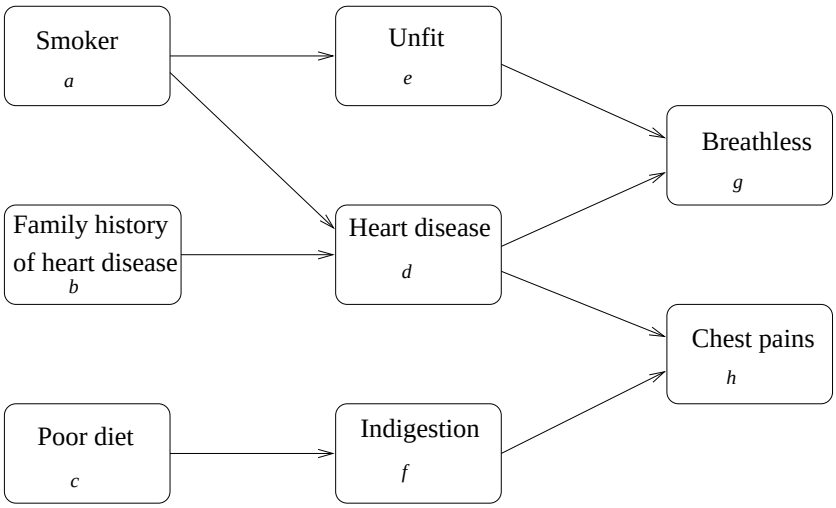

Figure 1: The DAG for an artificial medical diagnosis problem.

The key idea in specifying structure has been the use of a DAG (directed acyclic graph) to specify conditional independence relationships between (observed and unobserved variables). Figure 1 (from Ripley, 1996) shows an artificial medical example; the CHILD network (figure 2) is given in Spiegelhalter *et al.* (1993) and the net for EPTS (figure 3) is from Heckerman and Wellman (1995). The absence of links on such graphs expresses the structure through conditional independences. Some of them are obvious, for example that 'breathlessness' is conditionally independent of 'indigestion' given the status of 'unfit' and 'heart disease', but there are subtler statements too, with a calculus (in fact two) to read them from the graph. (These are statements that are made irrespective of the numbers that are assigned; there may be others encoded in the numbers.)

A DAG has a set of nodes or vertices representing variables X_v, directed edges between vertices and no directed cycles[2]. We will assume that all the variables take a finite set of values. It is obvious that the vertices in a DAG can be numbered so that the parents of a node have a lower number than the node itself. Then for any joint distribution

$$\mathsf{P}(X_1, \ldots, X_n) = \mathsf{P}(X_1) \prod_{i>1} \mathsf{P}(X_i \,|\, X_1, \ldots, X_{i-1})$$

For a distribution consistent with the DAG (called a *recursive model*) we have

$$\mathsf{P}(X_i \,|\, X_1, \ldots, X_{i-1}) = \mathsf{P}(X_i \,|\, \text{parents of } i) \tag{1}$$

so

$$\mathsf{P}(X_1, \ldots, X_n) = \mathsf{P}(X_1) \prod_{i>1} \mathsf{P}(X_i \,|\, \text{parents of } i) \tag{2}$$

Thus to specify the whole joint distribution we only have to give the distribution of the *root* X_1 and the conditional distribution of each variable given those at the parents of its nodes, a specification by *conditional probability tables* or CPTs. If the graph is sparse enough, the CPTs will be small and (fairly) easy to elicit from experts. We may use simplifying models (such as a logistic regression) for the CPTs.

[2] A directed cycle is a series of arrows from node to node that return to the first node.

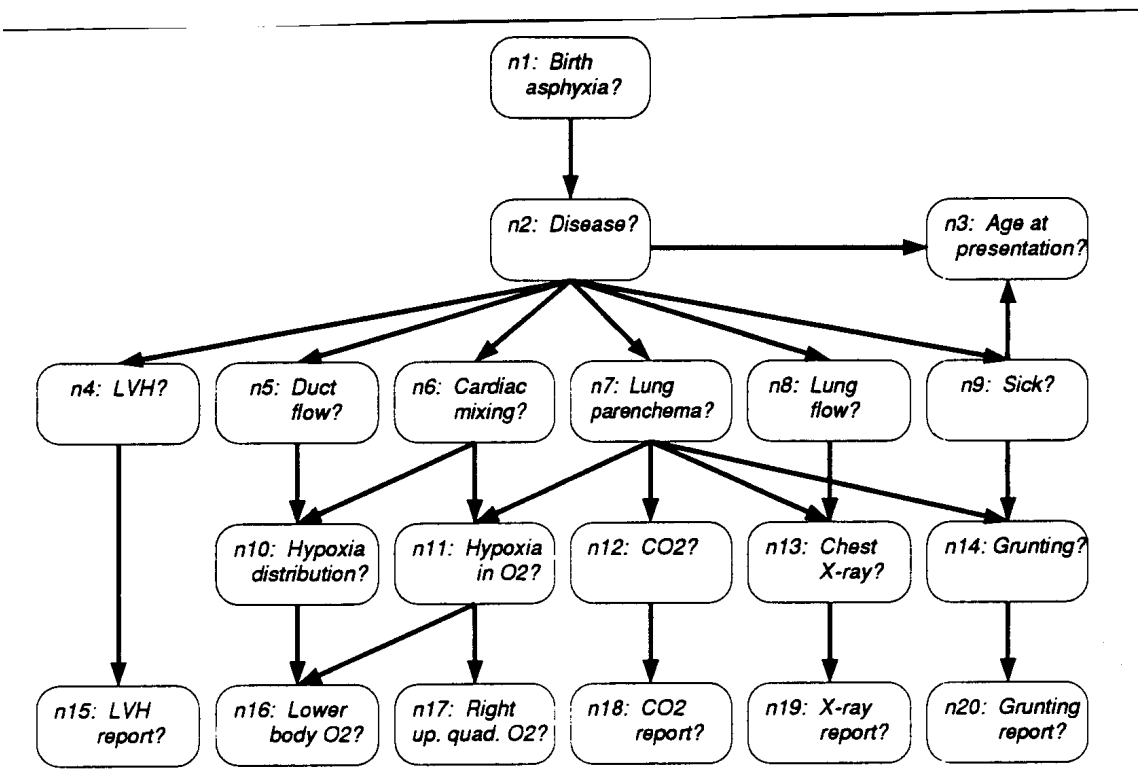

Figure 2: The DAG of the CHILD network (from Spiegelhalter *et al.*, 1993).

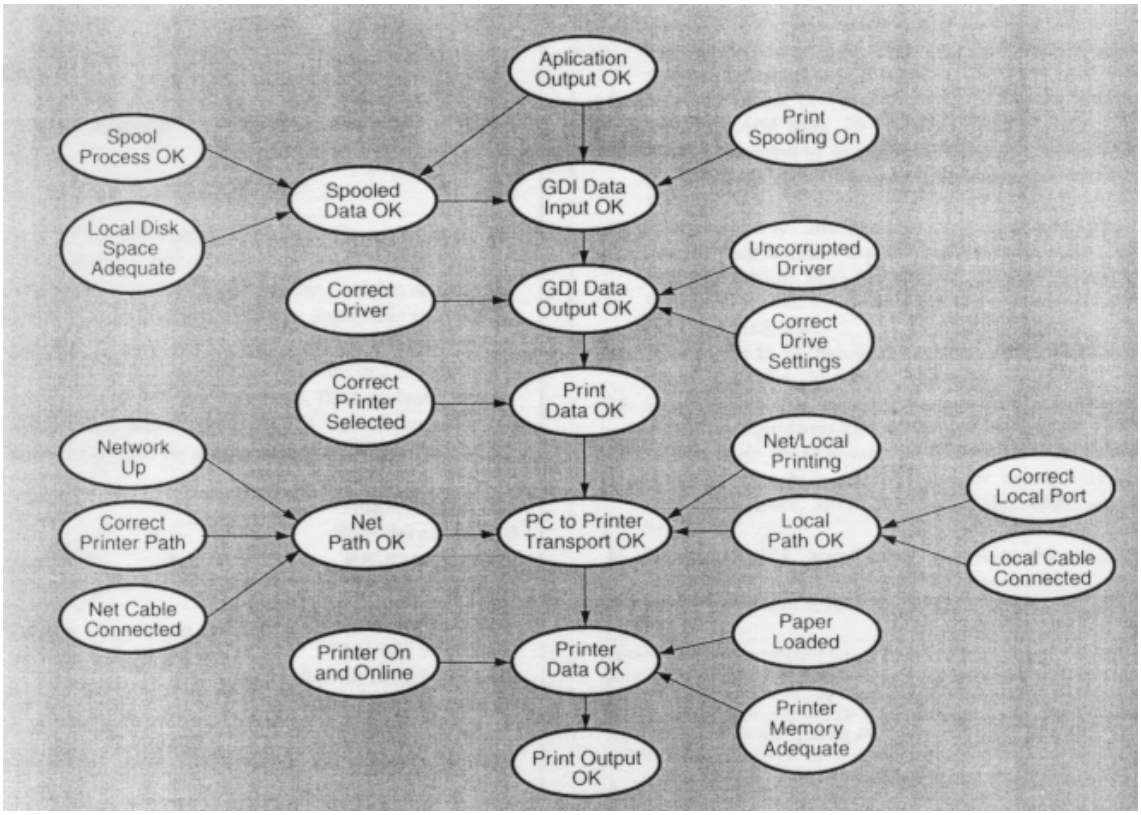

Figure 3: The DAG of the EPTS network (from Heckerman and Wellman, 1995).

$\mathsf{P}\{A\} = 0.5$	$\mathsf{P}\{B\} = 0.2$	$\mathsf{P}\{C\} = 0.3$
$\mathsf{P}\{D \mid A = \mathtt{F}, B = \mathtt{F}\} = 0.05$	$\mathsf{P}\{D \mid A = \mathtt{F}, B = \mathtt{T}\} = 0.3$	
$\mathsf{P}\{D \mid A = \mathtt{T}, B = \mathtt{F}\} = 0.2$	$\mathsf{P}\{D \mid A = \mathtt{T}, B = \mathtt{T}\} = 0.5$	
$\mathsf{P}\{E \mid A = \mathtt{F}\} = 0.3$	$\mathsf{P}\{E \mid A = \mathtt{T}\} = 0.5$	
$\mathsf{P}\{F \mid C = \mathtt{F}\} = 0.1$	$\mathsf{P}\{F \mid C = \mathtt{T}\} = 0.4$	
$\mathsf{P}\{G \mid D = \mathtt{F}, E = \mathtt{F}\} = 0.01$	$\mathsf{P}\{G \mid D = \mathtt{F}, E = \mathtt{T}\} = 0.5$	$\mathsf{P}\{G \mid D = \mathtt{T}\} = 1$
$\mathsf{P}\{H \mid D = \mathtt{F}, F = \mathtt{F}\} = 0$	$\mathsf{P}\{H \mid D = \mathtt{T}, F = \mathtt{F}\} = 0.5$	$\mathsf{P}\{H \mid F = \mathtt{T}\} = 1$

Table 1: The conditional probability tables (CPTs) for the DAG of figure 1.

Thus the knowledge base of a rule-based expert system is replaced by the structure of the DAG and the set of CPTs.

As with naïve Bayes we need to be able to turn the probability calculations around to calculate the probability of diseases (which are often at low-numbered nodes of the DAG) given evidence on variables (which are often at high numbers). There are now algorithms to do this for general DAGs and CPTs, so playing the rôle of a 'shell' in a rule-based expert system. (We will use one such shell, BAIES, in the practical class, and a demo version of HUGIN is supplied with Jensen's book.) Although these are general solutions, they are not guaranteed to be computationally feasible (the problem is NP-hard) and they are restricted to discrete variables.

The rest of this course will explore these concepts further. In full generality they are hard and much of the literature is erroneous. (This is one reason why Ripley (1996, Chapter 8) has complete proofs.) But the difficulties are in showing that the algorithms will *always* work: it is much easier to show how they work in small examples.

These systems are variously called belief networks, Bayes(ian) net(work)s, causal (probabilistic) networks, probabilistic expert systems, ...

As the interpretation of the DAG is in terms of conditional independence, which is not directed, we can ask if the directional arrows are really necessary. In fact they are not, and often we have a choice of direction, in particular of working in the causal direction (as in our examples) or in the diagnostic direction (symptoms imply possible causes). Causally defined systems are often simpler, and physicians find it easier to express causal CPTs (Tversky and Kahneman, 1982).

There is another way to think about the construction of a belief network. We could just number all the variables in some order, and declare as parents of node i a minimal set of variables for which (1) holds: for this DAG (2) holds and we have a recursive model. Note though that the simplicity of the DAG will be heavily dependent on the ordering.

4.1 Medical examples

PATHFINDER is a diagnostic expert system for lymph-node diseases, built by the Stanford Medical Computer Science programme during the 1980s (Heckerman, 1991; Heckerman *et al.*, 1992). It deals with over 60 diseases and over 100 inputs (symptoms and test results). It was built in four phases. PATHFINDER I was rule-based, without uncertainty reasoning. PATHFINDER II experimented with certainty factors and Dempster-Shafer belief functions, but showed that naïve Bayes outperformed all other methods considered, whilst being poor at excluding events the experts were sure could not happen. PATHFINDER III again used

naïve Bayes with a more careful elicitation of the probabilities. PATHFINDER IV used a belief network. Deciding on the set of nodes took 8 hours of interviews, the DAG took 35 hours, and making the 14,000 probability assessments took another 40 hours. A recent comparison showed that PATHFINDER IV is now outperforming the experts who were consulted—those experts 'being some of the world's leading pathologists'.

The ALARM network (Beinlich *et al.*, 1989) for the domain of anaesthesia in the operating theatre has 37 nodes and 46 links. It was derived by Beinlich from reading the literature and personal experience as an 'anesthesiologist'; fixing on the structure took about 10 hours and filling in the CPTs took about 20. It has been widely used to simulate data, in order to test techniques to learn the network from data (Cooper and Herskovits, 1992; Spirtes *et al.*, 1993; Heckerman, 1996).

The QMR-DT project (Shwe *et al.*, 1991; Pradhan *et al.*, 1994) constructed a belief network for internal medicine with 448 nodes, 906 links and 8,254 values in the CPTs, at first based on the INTERNIST-1 knowledge base. They use a simulation approach and report about half an hour of CPU is needed for each 'consultation'. This small number of CPT values is obtained only by assuming specific parametric forms for the tables; there would otherwise have been 133,931,430 values. The parametric forms used are known as 'noisy OR' and 'noisy MAX' gates. In a 'noisy OR' gate the boolean inputs are assumed to inhibit independently the output, with probability one if they are off and probability p_r if they are on. The output is on unless all inputs are inhibiting it.

5 Interpreting a Belief Network

The most convenient way to read conditional independence relationships from a DAG is by Pearl's notion of d-separation. Consider sets A, B and C of variables (and nodes). The sets of variables A and B are conditionally independent given C when the corresponding nodes are d-separated by C.

To define d-separation, consider a *trail* between A and B, which is a path from a node in A to one in B following links in either direction. A trail is *blocked* at an intermediate node v by C if the two edges at v either

(i) do not have converging arrows and $v \in C$, or

(ii) form a *collider* and neither v nor any of its descendants are in C.

Then A and B are d-separated by C if every trail between them is blocked at an intermediate node.

Consider the nodes 'Print Spooling On' and 'Correct Local Port' in figure 3. These are unconditionally independent as the node 'PC to Printer Transport OK' blocks the only path, for an empty C. In fact they will be conditionally independent given any set of variables which does not include 'Printer Data OK' or 'Print Output OK', or does include either of the GDI nodes or 'Print Data OK'.

Figure 3 is a relatively simple network, as only 'Aplication Output OK' (their spelling) has more than one effect. In the CHILD network many nodes have multiple children, but few have more than one parent. Here 'Cardiac mixing?' and 'Lung flow?' are conditionally independent given 'Disease?' but *not* given 'Disease?' and 'X-ray report?'.

6 Computing Probabilities on a Belief Network

There are two families of algorithms for computing probabilities on a belief network and updating them when evidence becomes available. One due to Pearl (1982) is described in Pearl (1988) and Russell and Norvig (1995). It applies only to *polytrees*, which are DAGs without undirected cycles, so there is precisely one trail between each pair of nodes.

The other family follows Lauritzen and Spiegelhalter (1988) and applies to general DAGs. However, it does so by converting them into trees of clusters of variables, so the process is sometimes known as *clustering*. We need a short excursion into graph theory.

6.1 Forming the join tree

The first step is to convert the DAG into an undirected graph. It does not quite suffice to drop the directional arrows, as the notion of d-separation gives a special place to colliders. We form the *moral graph* of a DAG by

1. replace all the directed edges by undirected ones and
2. add edges joining the parents (in the DAG) of each vertex if necessary.

(The distribution is then a Markov random field on the moral graph, for those who know about MRFs.)

A *clique* of a graph is a maximal complete subgraph. That is, it is a collection of vertices which are all connected to each other, and to which no other vertex is completely connected. The second step is to replace the moral graph by the set of cliques, which are connected to form a tree.

Unfortunately, not all moral graphs can be connected to form a tree of cliques. We need the graph to be *triangulated* (also called *chordal* or *decomposable*), that is have no cycle of four or more vertices without a chord (an edge joining two non-consecutive vertices). We make the moral graph triangulated if necessary by adding edges. The following procedure (Tarjan and Yannakakis, 1984) both triangulates the moral graph and forms the tree of cliques.

1. Order the vertices by *maximal cardinality search*. Start anywhere, and number next a vertex with the largest number of already numbered neighbours.
2. Starting with the highest numbered vertex, check for each vertex that all its lower-numbered neighbours are themselves neighbours. (By adding missing edges here, a triangulated graph will be produced.)
3. Identify all the cliques, and order them by the highest numbered vertex in the clique.
4. Form the *join tree* (also known as the junction tree) by connecting each clique to a predecessor (in the ordering of step 3) which shares the most vertices.

The variables that two neighbouring cliques have in common are called the *separator*.

Let's see how this procedure works on figure 1.

To form the moral graph (figure 4) we have to join a to b, d to e and d to f; this is already a triangulated graph. The cliques are abd, ade, cf, deg and dfh. One ordering by maximum cardinality search, starting from a, for the vertices is $abdegfhc$ and for the cliques is $C_1 = abd, C_2 = ade, C_3 = deg, C_4 = dfh, C_5 = cf$. The separators are then $S_2 = ad, S_3 =$

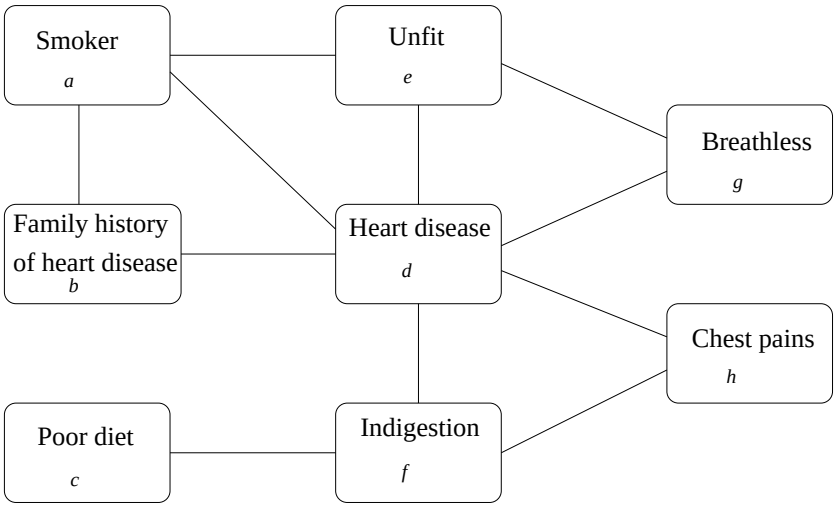

Figure 4: The moral graph associated with Figure 1.

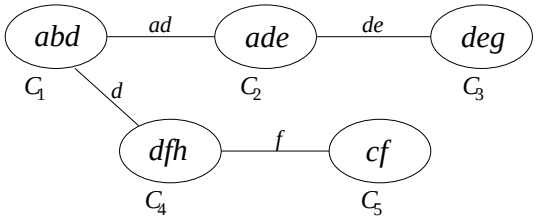

Figure 5: The clique tree associated with figure 4. The separators S_i are marked on the edges.

$de, S_4 = d, S_5 = f$. In forming the join tree we have considerable freedom, as C_4 can be linked to any of its predecessors. One choice (joining C_4 to C_3) would make the join tree into a chain; we chose the tree shown in figure 5.

A *join tree* is an undirected tree of collections C_i of random variables with the property that if a variable v is in collections C and C', then it is also in all the collections on the (unique) path on the tree joining C and C'. Since this applies to all $v \in C \cap C'$, we can conclude that $C \cap C' \subset D$ for all collections D on the path between C and C', and hence also for all separators S on that path.

Check this out for figure 5: $d \in C_3, C_4$ and hence d must be in C_1 and C_2: it is.

6.2 The marginal representation

There are many ways to represent the joint probability distribution of the random variables on the join tree. The most useful one is guaranteed to exist (this is the hard part and is only true if the graph is triangulated) and is given by

> A Markov distribution with respect to $\mathcal{G}$ can be written as a product of the distributions on the cliques of $\mathcal{G}$ divided by the product of the distributions on their intersections.

In symbols,

$$\mathsf{P}\{X_V\} = \prod_i \mathsf{P}\{X_{C_i}\} \;/\; \mathsf{P}\{X_{S_i}\} \tag{3}$$

where X_A refers to the variables on a set $A \subset V$ of nodes. If we have the marginal representation, we can find the marginal distribution of any variable from the marginal distribution of a clique C_j that contains it, by summing over the other variables.

A recursive model on a DAG $\mathcal{D}$ is a Markov distribution the moral graph $\mathcal{G}$ and the triangulated moral graph $\mathcal{G}^{\Delta}$. We will construct a marginal representation for a recursive model later.

A *potential representation* is of the form

$$\mathsf{P}\{X_V = x_v\} = \prod_C \phi_C(x_C) \Big/ \prod_S \psi_S(x_S) \tag{4}$$

where $1/0$ is taken as 0 (and so if $\psi_S(x_S) = 0$ we can adjust $\phi_C(x_C)$ to be zero). The marginal representation is a special case, and the key step in the Lauritzen-Spiegelhalter algorithm is to convert a general potential representation into the marginal representation.

6.3 Message passing

Calculations on potential representations are done by message-passing. If C and D are neighbours in the join tree with separator $S = C \cap D$, the message we pass from C to D is the margin of S in ϕ_C, that is

$$\psi_S^*(X_S) = \sum_{C \setminus S} \phi_C(X_C), \qquad \phi_D^*(X_D) = \phi_D(X_D) \times \frac{\psi_S^*(X_S)}{\psi_S(X_S)}$$

become the new terms in the potential representation. (Note that $\psi_S^*(X_S) = 0$ only if $\phi_C(X_C) = 0$ and hence $P(X_V) = 0$.)

Now consider a message passing scheme in which one clique is selected as the root of the tree, and first messages are passed in towards the root starting with the cliques furthest (in the number of links) from the nodes, and then messages are passed out from the node (so when a clique receives a message, it passes a message on to all its other neighbours). Suppose C and D are neighbouring cliques, with C farther from the root than D. In the inwards pass, C passes a message to D, so we have

$$\psi_S^*(X_S) = \sum_{C \setminus S} \phi_C(X_C)$$

In the outward pass, D passes a message to C. As D has received a message, ϕ_D may have changed, to ϕ_D^{**} say, but ϕ_C is unchanged. After the message passes from D to C we have

$$\psi_S^{**}(X_S) = \sum_{D \setminus S} \phi_D^{**}(X_D), \qquad \phi_C^{**}(X_D) = \phi_C(X_C) \times \frac{\psi_S^{**}(X_S)}{\psi_S^*(X_S)}$$

and so

$$\sum_{C \setminus S} \phi_C^{**}(X_C) = \sum_{C \setminus S} \phi_C(X_C) \frac{\psi_S^{**}(X_S)}{\psi_S^*(X_S)} = \left[\sum_{C \setminus S} \phi_C(X_C) \right] \frac{\psi_S^{**}(X_S)}{\psi_S^*(X_S)} = \psi_S^*(X_S) \frac{\psi_S^{**}(X_S)}{\psi_S^*(X_S)}$$

Thus after message passing is completed,

$$\sum_{C \setminus S} \phi_C(X_C) = \sum_{D \setminus S} \phi_D(X_D) = \psi_S(X_S)$$

If A is any collection of random variables contained in two or more cliques, it is contained in all cliques and separators on the path(s) between the cliques, and after message passing $\sum_{C\setminus A}\phi_C(X_C)$ and $\sum_{S\setminus A}\psi(X_S)$ will be the same for every clique and every separator containing A. (Just follow the paths to prove this.)

Fix a clique C_0. Choose another clique C' which is a leaf of the tree, and hence has only one link with separator S'. Let $R' = C' \setminus S'$. No vertex $v \in R'$ belongs any other clique or separator, as by the join-tree property we would have $v \in S'$. Thus after message passing

$$\begin{aligned} P(X_v, v \in V \setminus R') &= \sum_{R'} P(X_V) = \sum_{R'}\Big[\prod_C \phi_C(X_C) \Big/ \prod_S \psi_S(X_S)\Big] \\ &= \sum_{R'} \phi_{C'}(X_{C'}) \times \prod_{C\neq C'} \phi_C(X_C) \Big/ \prod_S \psi_S(X_S) \\ &= \psi_{S'}(X_{S'}) \times \prod_{C\neq C'} \phi_C(X_C) \Big/ \prod_S \psi_S(X_S) \\ &= \prod_{C\neq C'} \phi_C(X_C) \Big/ \prod_{S\neq S'} \psi_S(X_S) \end{aligned}$$

so the potential representation still holds for the smaller tree with clique C and its link removed.

By repeating this process for a remaining leaf clique other than C_0, we eventually find

$$P(X_v, v \in C_0) = \phi_{C_0}(X_{C_0})$$

and since C_0 was arbitrary, $P(X_C) = \psi_C(X_C)$ must hold for all cliques C. It then follows that $\psi_S(X_S) = P(X_S)$ for every separator, and so

$$P(X_V) = \prod_C P(X_C) \Big/ \prod_S P(X_S)$$

is the marginal representation formed after message passing on any potential representation.

Getting started

Given a recursive model on a DAG, we have from equation (2)

$$P(X_1, \ldots, X_n) = P(X_1) \prod_{i>1} P(X_i \,|\, \text{parents of } i)$$

At the moralization stage all the parents of v were 'married', and so the subgraph of v and its parents is completely connected and hence contained in some clique. Set $\phi_C \equiv 1$, $\psi_S \equiv 1$ and then for each term in (2) choose an appropriate clique C and multiply ϕ_C by $P(X_i \,|\, \text{parents of } i)$. This gives an initial potential representation. After message-passing we have a marginal representation, so we have proved that one exists.

Artificial medical example

In our example the specification of the distribution via the CPTs is

$$p(A)\, p(B)\, p(C)\, p(D|A,B)\, p(E|A)\, p(F|C)\, p(G|D,E)\, p(H|D,F).$$

The separators are then $S_2 = ad, S_3 = de, S_4 = d, S_5 = f$ and so the marginal representation is

$$\frac{p(A,B,D)\,p(A,D,E)\,p(D,E,G)\,p(D,F,H)\,p(C,F)}{p(A,D)\,p(D,E)\,p(D)\,p(F)}.$$

The marginal probabilities may be found from $p(C_1) = p(D \mid A,B)p(A)p(B)$, $p(C_2) = p(E \mid A)p(S_2)$, $p(C_3) = p(G \mid D,E)p(S_3)$, $p(C_5) = p(F \mid C)p(C)$ and $p(C_4) = p(H \mid F,D)p(S_5)p(S_4)$. We find, writing tables in lexicographic order (false before true, last index varies fastest).

ABD	0.38	0.02	0.07	0.03	0.32	0.08	0.05	0.05
AD	0.45	0.05	0.37	0.13				
ADE	0.315	0.135	0.035	0.015	0.185	0.185	0.065	0.065
DE	0.50	0.32	0.10	0.08				
DEG	0.495	0.005	0.16	0.16	0	0.1	0	0.08
D	0.82	0.18						
CF	0.63	0.07	0.18	0.12				
F	0.81	0.19						
DFH	0.6642	0	0	0.1558	0.0729	0.0729	0	0.0342

Note the treatment of clique $C_5 = cf$; we need the marginal for f, and this demands that we process C_5 before C_4. It is natural to think of C_4 depending on C_5, but to produce a tree we have to label the edge in the opposite direction.

We can also find the initial marginal probabilities via a potential representation and message-passing. Suppose we take initial potentials as $p(A)p(B)p(D \mid A,B)$, $p(E \mid A)$, $p(G \mid D,E)$, $p(H \mid F,D)$ and $p(F \mid C)p(C)$. Message-passing then multiplies ϕ_{C_2} by $p(A,D)$, ϕ_{C_3} by $p(D,E)$ and ϕ_{C_4} by $p(D)$ and by $p(F)$ to form the marginal representation.

6.4 Conditioning on evidence

For evidence $\mathcal{E}$ of the form $\bigcup\{X_{e_i} \in E_i\}$, given any potential representation of $P(X_V)$, for each i select a clique $C \ni e_i$ and set $\phi^*_C = \phi_C \times I(x_{e_i} \in E_i)$. We then have the potential representation

$$P(X_V, \mathcal{E}) = P(X_V) \prod_i I(X_{e_i} \in E_i) \prod_C \phi^*_C(X_C) \Big/ \prod_S \psi_S(X_S)$$

Nothing in the message-passing argument depends on the probabilities summing to one, so after message passing we have the marginal representation

$$P(X_V, \mathcal{E}) = \prod_C P(X_C, \mathcal{E}) \Big/ \prod_S P(X_S, \mathcal{E})$$

From this we can read off $P(\mathcal{E})$ (sum over a small separator) and hence find $P(X_A \mid \mathcal{E})$ for any collection A contained in some clique, in particular for any node of the DAG.

Artificial medical example

Suppose a patient presents symptoms of breathlessness and chest pains, and is a smoker. What is the probability of heart disease? We condition on the evidence $A = G = H = \mathtt{T}$. We

illustrate the message-passing approach by sending messages to $C_1 = abd$ at the root of the tree. We enter $G = H = \mathtt{T}$ at cliques C_3 and C_4. For $A = \mathtt{T}$ we have a choice of cliques, and choose C_1. We start at C_4 with a message over $S_4 = d$ to C_1, the new-to-old ratio of the separator distributions.

DFH	0	0	0	0.1558	0	0.0729	0	0.0342
D	0.1558				0.1071			
msg	$0.1558/0.82 = 0.19$				$0.1071/0.18 = 0.595$			

Clique C_3 sends a message over $S_3 = de$ to C_2:

DEG	0	0.005	0	0.16	0	0.1	0	0.08
DE	0.005		0.16		0.1		0.08	
msg	$0.005/0.5 = 0.01$		$0.16/0.32 = 0.5$		1		1	

This is then incorporated into clique C_2 and a message sent over $S_2 = ad$ to C_1:

ADE	0.00315	0.0675	0.035	0.015	0.00185	0.0925	0.065	0.065
AD	0.07065		0.05		0.09435		0.13	
msg	0.157		1		0.255		1	

Clique C_1 then incorporates two messages and its own constraint. We need only give the results for $A = \mathtt{T}$:

BD	$0.32 \cdot 0.19 \cdot 0.255$	$0.08 \cdot 0.595 \cdot 1$	$0.05 \cdot 0.19 \cdot 0.255$	$0.05 \cdot 0.595 \cdot 1$
=	0.015504	0.0476	0.002423	0.02975
D	0.017927	0.07735		

so $\mathsf{P}\{\text{Evidence}\} = 0.095277$ and $\mathsf{P}\{D \mid \text{Evidence}\} = 0.812$.

As C_1 is the root, we should then pass messages back down the tree, but our question has already been answered from the marginal in C_1. It will be convenient at this stage to normalize, that is to divide the marginal distribution C_1 by $\mathsf{P}\{\text{Evidence}\}$. The messages sent to C_2 and C_4 are then approximately

AD	0	0	$0.188/0.09435$	$0.812/0.13$
D	$0.188/0.1558$	$0.812/0.1071$		

This modifies the clique marginals to (approximately)

ADE	0	0	0	0	0.0037	0.1843	0.406	0.406
DFH	0	0	0	0.188	0	0.553	0	0.259

These send messages to C_3 and C_5 of

DE	$0.00037/0.005$	$0.1843/0.16$	$0.406/0.1$	$0.406/0.08$
F	$0.553/0.81$	$0.447/0.19$		

and those cliques can be updated to

DEG	0	0.00037	0	0.1843	0	0.406	0	0.406
CF	0.430	0.165	0.123	0.282				

After these calculations it is often easy to answer further questions. Suppose we discover that the patient's family has a history of heart disease. This amounts to new evidence that $B = \mathtt{T}$. Rather than go through the full procedure of propagating messages, we can just examine the marginal distribution of C_1 to see that the conditional probability of heart disease is now $0.02975/(0.002423 + 0.02975) \approx 0.925$. Similar calculations will often allow us to evaluate the value of 'buying' various items of new evidence, and so decide which to obtain (Lauritzen and Spiegelhalter, 1988).

These manipulations are automated in various packages, notably HUGIN.

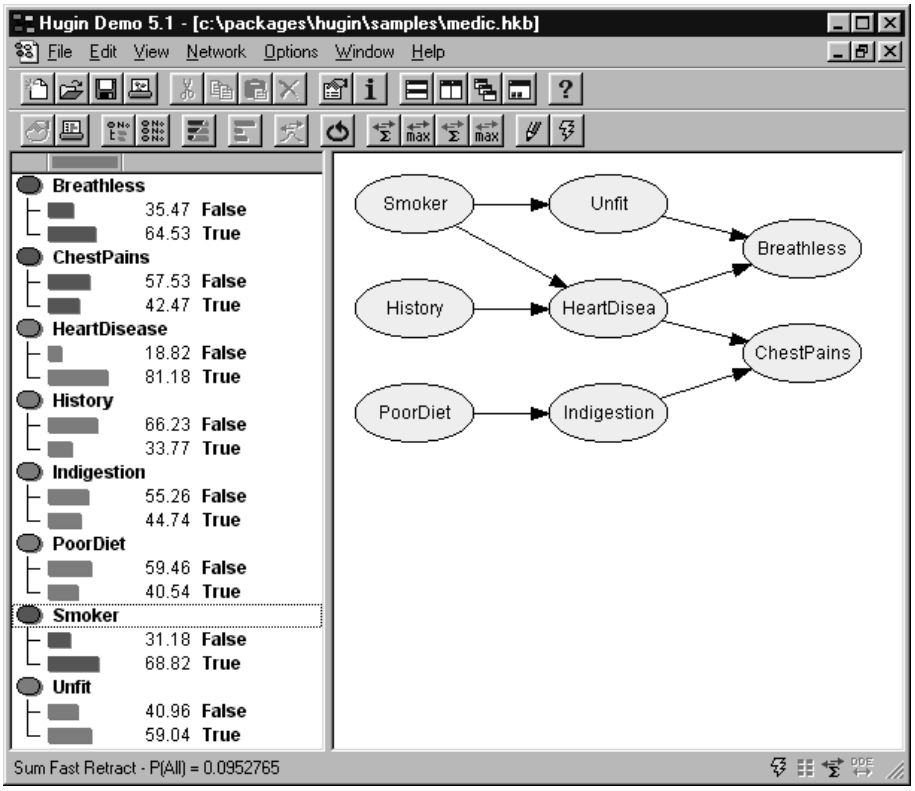

Figure 6: A screenshot of the HUGIN package for manipulating belief networks. The left-panel is showing the marginal probabilities for nodes without evidence, and the conditional probabilities given all the other evidence at nodes with evidence.

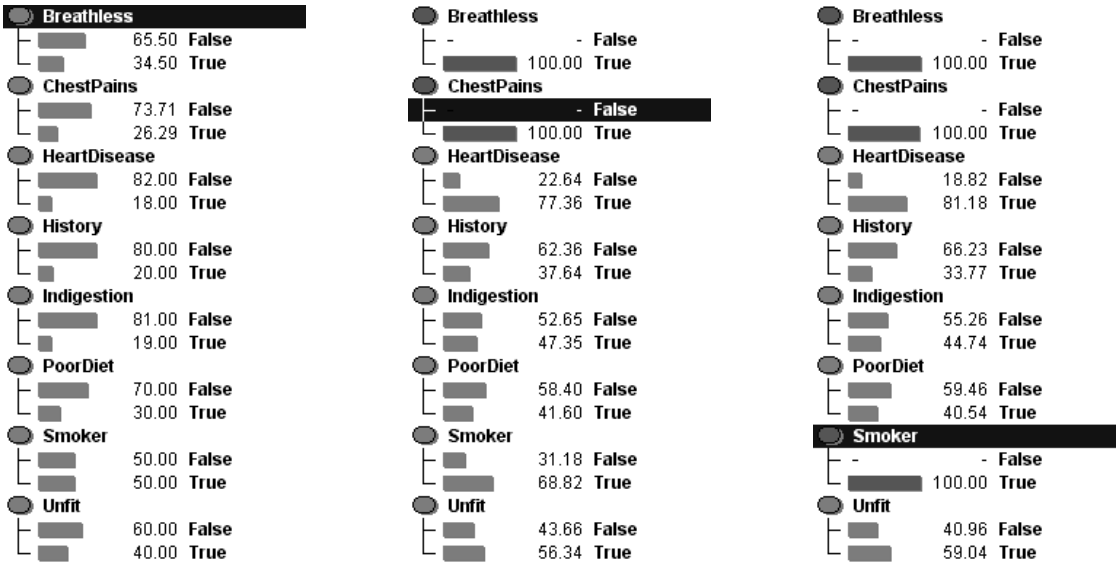

Figure 7: Graphical displays from HUGIN of the changes in marginal probabilities as evidence is introduced (the highlighted item).

References

Bachant, J. and McDermott, J. (1984) R1 revisited: Four years in the trenches. *AI Magazine*, (Fall issue), 21. [1]

Beinlich, I., Suermondt, H., Chavez, R. and Cooper, G. (1989) The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks. In *Proceedings of the Second European Conference on Artificial Intelligence in Medicine*, pp. 247–256, Berlin. Springer. [12]

Buchanan, B. G. and Shortliffe, E. H. (eds) (1984) *Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project*. Reading, MA: Addison-Wesley. [1, 21]

Buchanan, B. G., Sutherland, G. L. and Feigenbaum, E. A. (1969) Heuristic DENDRAL: a program for generating explanatory hypotheses in organic chemistry. In *Machine Intelligence 4* (Eds B. Meltzer, D. Michie and M. Swann), pp. 209–254. Edinburgh: Edinburgh University Press. [1]

Cooke, R. M. (1991) *Experts in Uncertainty. Opinion and Subjective Probability in Science*. New York: Oxford University Press. [6]

Cooper, G. F. (1989) Current research directions in the development of expert systems based on belief networks. *Applied Stochastic Models and Data Analysis*, **5**, 39–52. [2]

Cooper, G. F. and Herskovits, E. (1992) A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, **9**, 309–347. [12]

Crevier, D. (1993) *AI. The Tumultuous History of the Search for Artificial Intelligence*. New York: Basic Books. [2, 3]

De Dombal, F. T. (1980) *Diagnosis of Acute Adominal Pain*. Edinburgh: Churchill Livingstone. [1]

Duda, R., Gaschnig, J. and Hart, P. (1979) Model design in the Prospector consultant system for mineral exploration. In *Expert Systems in the Microelectronic Age* (Ed. D. Michie), pp. 153–167. Edinburgh: Edinburgh University Press. [1]

Feigenbaum, E. A., Buchanan, B. G. and Lederberg, J. (1971) On generality and problem solving: A case study using the DENDRAL program. In *Machine Intelligence 6* (Eds B. Meltzer and D. Michie), pp. 165–190. Edinburgh: Edinburgh University Press. [1]

Goodall, A. (1985) *The Guide to Expert Systems*. Oxford, NJ: Learned Information. [2]

Heckerman, D. (1991) *Probabilistic Similarity Networks*. Cambridge, MA: The MIT Press. [11]

Heckerman, D. (1996) Bayesian networks of knowledge discovery. In *Advances in Knowledge Discovery and Data Mining* (Eds U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth and R. Uthurusamy), pp. 273–305. Menlo Park, CA: The AAAI Press. [12]

Heckerman, D. and Wellman, M. P. (1995) Bayesian networks. *Communications of the ACM*, **38**(3), 26–30. [1, 9, 10]

Heckerman, D., Horvitz, E. and Nathwani, B. (1992) Toward normative expert systems. 1: The PATHFINDER project. *Methods of Information in Medicine*, **31**, 90–105. [11]

Heckerman, D., Breese, J. S. and Rommelse, K. (1995) Decision-theoretic troubleshooting. *Communications of the ACM*, **38**(3), 49–57. [1]

Lauritzen, S. and Spiegelhalter, D. J. (1988) Local computations with probabilities on graphical structures and their application to expert systems (with discussion). *Journal of the*

Royal Statistical Society series B, **50**, 157–224. [Reprinted in Shafer and Pearl (1990)]. [2, 13, 18]

Laviolette, M., Seaman, Jr, J. E., Barrett, J. D. and Woodall, W. H. (1995) A probabilistic and statistical view of fuzzy methods (with discussion). *Technometrics*, **37**, 249–292. [5]

Lindley, D. V. (1987) The probability approach to the treatment of uncertainty in artificial intelligence and expert systems. *Statistical Science*, **2**(1), 17–24. [6]

Lindsay, R. K., Buchanan, B. G., Feigenbaum, E. A. and Lederberg, J. (1980) *Applications of Artificial Intelligence for Organic Chemistry: The DENDRAL Project*. New York: McGraw-Hill. [1]

McDermott, J. (1982) R1: A rule-based configurer of computer systems. *Artificial Intelligence*, **19**(1), 39–88. [1]

Miller, R. A., Pople, H. E. and Myers, J. D. (1982) INTERNIST-1, an experimental computer-based diagnostic consultant for general internal medicine. *New England Journal of Medicine*, **307**, 468–476. [1]

Pearl, J. (1982) Reverend Bayes on inference engines: a distributed hierarchical approach. In *Proceedings of the AAAI National Conference on Artificial Intelligence (Pittsburgh, 1982)* (Ed. D. Waltz), pp. 133–136, Menlo Park, CA. AAAI. [13]

Pearl, J. (1988) *Probabilistic Inference in Intelligent Systems. Networks of Plausible Inference*. San Mateo, CA: Morgan Kaufmann. [4, 6, 13]

Pople, H. E. (1985) Evolution of an expert system: from Internist to Caduceus. In *Artificial Intelligence in Medicine* (Eds I. De Lotto and M. Stefanelli), pp. 179–208. Amsterdam: North-Holland. [1]

Pradhan, M., Provan, G. M., Middleton, B. and Henrion, M. (1994) Knowledge engineering for large belief networks. In *Proceedings of Uncertainty in Artificial Intelligence*, Seattle, Washington. Morgan Kaufmann. [12]

Ripley, B. D. (1996) *Pattern Recognition and Neural Networks*. Cambridge: Cambridge University Press. [9, 11]

Russell, S. J. and Norvig, P. (1995) *Artificial Intelligence. A Modern Approach*. Englewood Cliffs, NJ: Prentice-Hall. [1, 3, 4, 5, 13]

Shafer, G. (1987) Probability judgment in artificial intelligence and expert systems. *Statistical Science*, **2**(1), 3–16. [6]

Shafer, G. and Pearl, J. (eds) (1990) *Readings in Uncertainty Reasoning*. San Mateo, CA: Morgan Kaufmann. [4, 6, 20, 21]

Shortliffe, E. H. (1976) *Computer-Based Medical Consultations: MYCIN*. Amsterdam: Elsevier/North-Holland. [1]

Shortliffe, E. H. and Buchanan, B. G. (1975) A model of inexact reasoning in medicine. *Mathematical Biosciences*, **23**, 351–379. [Reprinted as Chapter 11 of Buchanan and Shortliffe (1984) and in Shafer and Pearl (1990).]. [4]

Shwe, M., Middleton, B., Heckerman, D., Henrion, M., Horvitz, E. and Lehmann, H. (1991) Probabilistic diagnosis using a reformulation of the INTERNIST-1/QMR knowledge base. 1: The probabilistic model and inference algorithms. *Methods of Information in Medicine*, **30**, 241–255. [12]

Spiegelhalter, D. J., Dawid, A. P., Lauritzen, S. L. and Cowell, R. G. (1993) Bayesian analysis in expert systems (with discussion). *Statistical Science*, **8**, 219–283. [1, 9, 10]

Spirtes, P., Glymour, C. and Scheines, R. (1993) *Causality, Prediction, and Search*, volume 81 of *Lecture Notes in Statistics*. New York: Springer. [12]

Tarjan, R. E. and Yannakakis, M. (1984) Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs. *SIAM Journal of Computing*, **13**, 566–579. [13]

Tversky, A. and Kahneman, D. (1982) Causal schemata in judgements under uncertainty. In *Judgement Under Uncertainty: Heuristics and Biases* (Eds D. Kahneman, P. Slovic and A. Tversky). Cambridge: Cambridge University Press. [11]

Winston, P. H. (1992) *Artificial Intelligence*. Reading, MA: Addison-Wesley, third edition. [1, 2, 4]