Bayesian networks basics

Finn V. Jensen
Department of Mathematics and Computer Science
Aalborg University
Fredrik Bajers Vej 7
DK-9220 Aalborg Ø
Denmark

Abstract

This article is intended as an introduction to the theoretical background for Bayesian networks. First some principles for reasoning under uncertainty in causal structures are presented, and the basic probability calculus behind Bayesian networks is introduced. Then some hints are given on how probabilities are updated, and at the end there are some references to systems based on Bayesian networks.

1 Normative systems

Let us start with observing an expert at work. Her domain of expertise is a well-defined part of the world. She may be a physician examining a patient, she may be a banker questioning an applicant for a loan, or she may be a pilot monitoring the aircraft.

A simple view of the tasks of the expert is the triangle in Figure 1. First of all she observes her part of the world to establish the state of it.

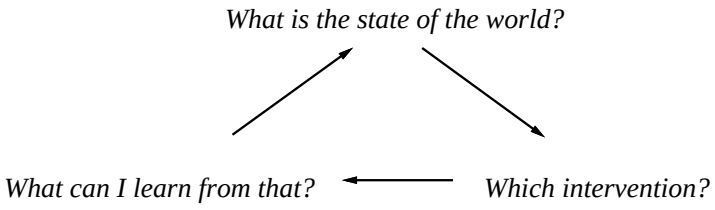

Figure 1: The task-triangle of an expert.

Based on her interpretation of the state of the world, the expert decides on an action. For any action the expert has some expectations. Sometimes they come true and sometimes they do not; but in any case she will learn from the results of the actions. This may help her interpreting the world in future.

The first *expert systems* were constructed in the late 1960s. Their scope is decision making characterized by repeatedly deciding cases with a common structure. The systems are constructed as computer models of the expert.

The long-term scope of the technology was that experts could be replaced by computer systems which modelled the best experts in the world. The building blocks for modelling the expert were *production rules.* A production rule is of the form

if *condition* **then** (*fact or action*)

where the condition is a logical expression. Though the language is very simple, it turned out to be rather powerful when modelling expert's reasoning, and several impressive rule based expert systems were constructed (for example MYCIN (Shortliffe 1976) and R1 (McDermott 1984).)

Rather soon after their first successes it became clear that rule based systems have their shortcomings. One of the major problems was how to treat uncertainty, and it was demonstrated that rule based systems are not able to treat uncertainty coherently.

Normative expert systems are an alternative to rule based expert systems. Both types of systems deal with repeated decision making, but the design principles for normative systems differ from rule based systems in three ways

- instead of modelling the expert, model the domain
- instead of using a non-coherent uncertainty calculus tailored for rules, use classical probability calculus and decision theory
- instead of replacing the expert, support her

Historically, the principles above are not new. In the 1960s attempts were already being made to use probability theory in expert systems (Gorry & Barnett 1968). However, due to the very heavy calculation load required the attempt was given up and considered an intractable task (Gorry 1973).

In the mid 1980's the principles were revived. Work by Pearl (1986b) introduced Bayesian networks to expert systems, work by Lauritzen & Spiegelhalter (1988) and Jensen, Olesen & Andersen (1990) gave very efficient calculation methods, and with the MUNIN system (Andreassen, Jensen, Andersen, Falck, Kjærulff, Woldbye, Sørensen, Rosenfalck & Jensen 1989) it was demonstrated that the necessary calculations for very large networks can indeed be tractable.

In the present paper we focus on the upper corner of Figure 1: how can observations change our belief of non-observed events?

For the other tasks, the reader is referred to the litterature. For example Shachter (1986), Shenoy (1992) and Jensen, Jensen & Dittmer (1994) on interventions, and Spiegelhalter & Lauritzen (1990), Buntine (1994) and Heckerman, Geiger & Chickering (1994) on learning.

2 Basic principles for reasoning under uncertainty

The basic knowledge when reasoning under uncertainty is whether information on some event influences your belief of other events. The reason why rule based systems can not capture reasoning under uncertainty is that dependence between events changes with knowledge of other events. The following example illustrates this.

2.1 Wet grass

Mr Homes leaves his house in the morning and notices that his grass is wet. He reasons: "I think it has been raining tonight. Then my neighbour, dr. Watson's grass is most probably wet also". That is, the information that Holmes' grass is wet has an influence on his belief of the status of Watson's grass. Now, assume that Holmes has checked the rain meter, and it was dry. Then he will not reason as above, and information on Holmes' grass has no influence on his belief about Watson's grass.

Next, consider two possible causes for wet grass. Besides rain, Holmes may have forgot to turn off the sprinkler. Assume that Mr. Holmes the next morning again notices that his grass is wet. Holmes' belief of both rain and sprinkler increases. Now he observes that Watson's grass is wet, and he concludes, that it has rained tonight. The next step in the reasoning is virtually impossible through rules, but natural for human beings, namely explaining away: *Holmes' wet grass has been explained by the rain, and thus there is no longer any reason to believe that the sprinkler has been on. Hence Holmes' belief of sprinkler is reduced to (almost) its initial size.*

Explaining away is another example of dependence changing with the information available. In the intial state, when nothing is known, rain and sprinkler are independent. However, when we have information on Holmes' grass, then rain and sprinkler become dependent.

2.2 Causal networks

The situations above can be described by a graph. The events are nodes, and two nodes A and B are connected by a directed link from A to B if A has a causal impact on B. Figure 2 is a graphical model for Holmes' small world of wet grass.

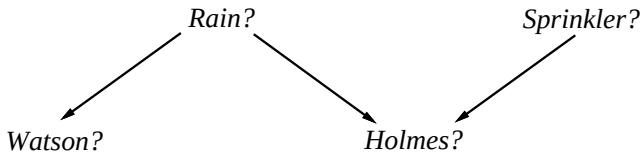

Figure 2: A network model for the wet grass example. Rain and sprinkler are causes of Holmes' grass being wet. Only rain can cause Watson's grass to be wet.

Figure 2 is an example of a causal network. A causal network consists of a set of *variables* and a set of *directed links* between variables. Mathematically the structure is called a *directed graph*. When talking about the relations in a directed graph we use the wording of family relations: if there is a link from A to B we say that B is a *child* of A, and A is a *parent* of B.

The variables represent events (propositions). In Figure 2 each variable has the states *yes* and *no* reflecting whether a certain event had taken place or not. In general a variable can have any number of mutually exclusive states. A variable may for example be the colour of a car (states $blue, green, red, brown$), the number of children in a family (states $0, 1, 2, 3, 4, 5, 6 > 6$) or a disease (states *bronchitis, tuberculosis, lung cancer*).

From the graph in Figure 2 you can read off the dependencies and independencies in the small world of wet grass. For example, we can see that if we know that it has not rained tonight, then

information on Watson's grass has no influence on our belief on Holmes' grass.

Pearl (1986a) and Verma (1987) have analysed the ways in which influence may run between variables in a causal network. We say that two variables are *separated* if new evidence on one of them has no impact on our belief of the other. If the state of a variable is known we call it *instantiated*. The three cases in Figures 3, 4, and 5 cover all types of connections in a causal network.

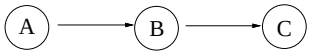

Figure 3: Serial connection. Influence may run from A to C and vica verse unless B is unstantiated.

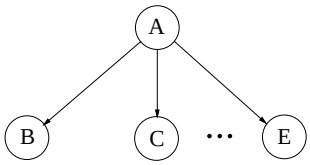

Figure 4: Diverging connection. Influence may run between A's children unless A is instantiated.

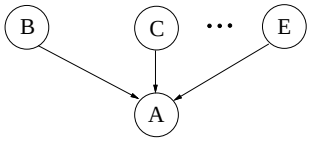

Figure 5: Converging connection. If nothing is known about A except what may be inferred from knowledge of its parents $B, \ldots, E$, then the parents are separated. If A changes certainty it opens for communication between its parents.

Definition (d-separation). Two variables A and B in a causal network are *d-separated* if for all trails between A and B there is an intermediate variable V such that either

- the connection is serial or diverging and the state of V is known.

or

- the connection is converging and neither V nor any of V's descendants have received evidence.

Note that in Figure 2 $Sprinkler?$ and $Watson?$ are d-separated because the connecting trail is converging around the variable $Holmes?$.

Note that d-separation is a property of human reasoning, and therefore any calculus for uncertainty in causal structures must obey the principle that whenever A and B are d-separated then new information on one of them does not change the certainty of the other.

2.3 Probability calculus

So far nothing has been said about the quantitative part of certainty assessment. Various certainty calculi exist, but we shall present the so called Bayesian calculus, which is *classical probability calculus*.

2.3.1 Basic calculus

The basic concept in the Bayesian treatment of certainties in causal networks is *conditional probability*.

A conditional probability statement is of the following kind:

"Given the event B (and everything else known is irrelevant for A), then the probability of the event A is x"

The notation for the statement above is $P(A \mid B) = x$.

The *Fundamental Rule* for probability calculus is the following:

$$P(A \mid B)P(B) = P(A, B), \tag{1}$$

where $P(A, B)$ is the probability of the joint event $A \wedge B$.

From (1) follows $P(A \mid B)P(B) = P(B \mid A)P(A)$ and this yields the well known *Bayes' Rule*:

$$P(B \mid A) = \frac{P(A \mid B)P(B)}{P(A)}. \tag{2}$$

If A is a variable with states $a_1, \ldots, a_n$, then $P(A)$ is a probability distribution over these states:

$$P(A) = (x_1, \ldots, x_n) \qquad x_i \geq 0 \qquad \sum_{i=1}^{n} x_i = 1$$

where x_i is the probability of A being in state a_i. Note that if A and B are variables, then $P(A, B)$ is a table of probabilities $P(a_i, b_j)$ for the possible pairs of states of A and B.

From a table $P(A, B)$ the probability distribution $P(A)$ can be calculated. Let a_i be a state of A. There are exactly m different events for which A is in state a_i, namely the mutually exclusive events $(a_i, b_1), \ldots, (a_i, b_m)$. Therefore

$$P(a_i) = P(a_i, b_1) + P(a_i, b_2) + \ldots + P(a_i, b_m).$$

This calculation is called *marginalization* and we say that the variable B is marginalized out of $P(A, B)$ (resulting in $P(A)$). The notation is

$$P(A) = \sum_{B} P(A, B) \tag{3}$$

2.3.2 Subjective probabilities

Probability calculus does not require that the probabilities are based on theoretical results or frequencies of repeated experiments. Probabilities may also be completely subjective estimates of the certainty of an event.

A subjective probability may e.g. be my personal assessment of the chances of Denmark winning the European Soccer Cup again in 1996.

A way to assess this probability could be the following. I am given the choice between two gambles:

1. If Denmark wins the Cup in 1996 I will receive $ 100.
2. I will by July 1996 be allowed to draw a ball from an urn with n red balls and $100 - n$ white balls. If my ball is red I will get $ 100.

Now, if all balls in the urn are red I will prefer 2, and if all balls are white I will prefer 1. There is a number n for which the two gambles are equally attractive, and for this n, $\frac{n}{100}$ is my estimate of the probability for Denmark to win the Cup. (I shall not disclose the n to the reader).

2.3.3 Conditional independence

The blocking of influence between variables as described in Section 2.2 is in the Bayesian calculus reflected in the concept of *conditional independence.* The variables A and C are *independent given the variable* B if

$$P(A \mid B) = P(A \mid B, C) \tag{4}$$

This means that if the state of B is known then no knowledge of C will alter the probability of A.

Conditional independence appears in the cases of serial and diverging connections (see Figure 6).

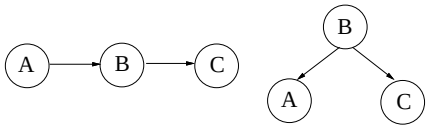

Figure 6: Examples where A and C are conditionally independent given B.

2.4 Bayesian networks

Let A be a parent of B in a causal network. Using probability calculus it would be natural to let $P(B \mid A)$ be the strength of the link. However, if also C is a parent of B, then the two conditional probabilities $P(B \mid A)$ and $P(B \mid C)$ alone do not give any clue on how the impacts from A and B interact. They may co-operate or counteract in various ways. So, we need a specification of $P(B \mid A, C)$.

A Bayesian network consists of the following

> A set of *variables* and a set of *directed edges* between variables.
>
> Each variable has a finite set of mutually exclusive states.
>
> The variables together with the directed edges form a *directed acyclic graph* (DAG).[1]
>
> To each variable A with parents $B_1, \ldots, B_n$ is attached a conditional probability table $P(A \mid B_1, \ldots, B_n)$.

Note that if A has no parents then the table reduces to unconditional probabilities $P(A)$ (see Figure 7).

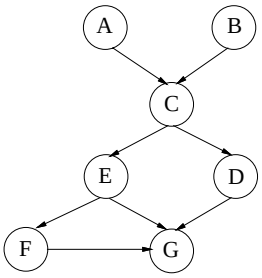

Figure 7: A directed acyclic graph (DAG). The probabilities to specify are $P(A)$, $P(B)$, $P(C \mid A, B)$, $P(E \mid C)$, $P(D \mid C)$, $P(F \mid E)$, and $P(G \mid D, E, F)$.

The requirement that the graph must be a DAG is a proper restriction. It may happen that the domain to be modelled contains feed-back cycles (see Figure 8).

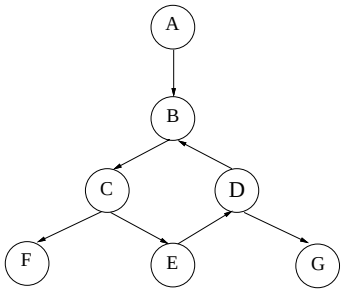

Figure 8: A directed graph with a feed-back cycle. This is not allowed in Bayesian networks.

Feed-back cycles are difficult to model quantitatively; for causal networks no calculus has been developed that can cope with feed-back cycles.

Bayesian networks admit d-separation

As argued in Section 2.2, any uncertainty calculus must obey the rules formulated in the definition of d-separation. This holds for Bayesian Networks: if A and B are d-separated in a Bayesian

[1] A directed graph is *acyclic* if there is no directed path $A_1 \to \cdots \to A_n$ such that $A_1 = A_n$.

network with evidence e entered, then $P(A \mid B, e) = P(A \mid e)$. This means that you can use d-separation to read-off conditional independencies.

2.4.1 The chain rule

Let U be a universe of variables. What we really would like to have is the table $P(U)$ giving the probabilities of all possible configurations of the universe. However, usually $P(U)$ is so large that it cannot be stored in any computer. For example, if U consists of 30 variables with 3 states each, then $P(U)$ consists of more than 10^{14} numbers.

Therefore, we look for a more compact *representation* of $P(U)$: a way of storing information from which $P(U)$ can be calculated if needed.

A Bayesian network over U is such a representation: if the conditional independencies in the Bayesian network hold for U, then $P(U)$ can be calculated from the conditional probabilities specified in the network.

The chain rule:

Let BN be a Bayesian network over $U = \{A_1, \ldots, A_n\}$. Then the joint probability distribution $P(U)$ is the product of all conditional probabilities specified in BN:

$$P(U) = \prod_i P(A_i \mid p(A_i))$$

where $p(A_i)$ is the parent set of A_i.

3 Belief updating in Bayesian networks

In this section we shall give some hints on how observations are used in Bayesian networks to change beliefs. The first break through was due to Kim & Pearl (1983), where a method for loop-free graphs was presented. A general method was presented by Lauritzen & Spiegelhalter (1988). We shall outline the HUGIN method proposed by Jensen et al. (1990), which is a modification of the Lauritzen-Spiegelhalter method.

Let BN be a Bayesian network over the variables U , and let e be a set of statements of the kind "*the variable A is in state a*". That is, e is the statement "*the joint configuration of $A, \ldots B$ is $(a, \ldots, b)$*". We now want the posterior probability distribution $P(X \mid e)$ for all variables X in U.

Mathematically the problem is solved in the following way:

- use the chain rule to calculate $P(U)$
- look up $P(U, e)$, the part of $P(U)$ corresponding to the configuration $(a, \ldots, b)$
- marginalize $P(U, e)$ down to $P(X, e)$ for each X in U (for each state x of X, sum up all entries in $P(U, e)$ with X in state x)

- $P(X \mid e)$ is the result of normalizing $P(X, e)$. That is, divide $P(X, e)$ by the sum of all its entries

However, usually $P(U)$ is so large that it cannot be stored in any computer, and even if it is possible the calculations involved may be overwhelming.

Inference in trees

Consider the tiny network in Figure 9. If we get a new distribution $P^*(A)$ of A, then the fundamental rule and marginalization can be used to calculate the new distribution of B.

$$P^*(B) = \sum_A P(B \mid A)P^*(A)$$

So, inference along the direction of the links is rather straightforward. You may think of the inference as a message passed from A to B. The message passed is a distribution of A, and in B the distribution of B is updated.

Figure 9: A tiny Bayesian network.

Belief updating also works in the opposite direction: information on B can be used to change our belief of A. For Bayesian networks the tool for inferring in the opposite direction is Bayes' rule. We shall not give the details, but you may also in this situation think of the inference as a distribution on B passed to A, and in A the message is used to update the belief of A.

In Figure 10 we show a Bayesian network which happens to be a tree. Messages can be passed along the links in both directions. That is, a node X can send a message to any neighbour Y. The message sent is the current distribution of X, and the node Y uses the message to update its own distribution.

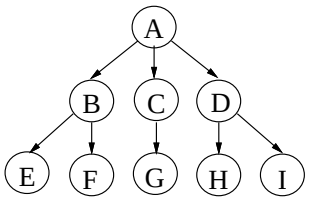

Figure 10: A Bayesian tree. Messages may be sent along each link in both directions.

First, let the nodes send messages anarchisticly, and wait for a while. It can proved that there is a stage, *the stable stage*, where no further message will change any distribution. Furthermore, the stable stage can be reached after a finite number of message passings, and in this stage each node holds the correct probability distribution.

The message passing can be performed more organized according to the following rule:

A node X can send a message to a neighbour Y if X has recieved a message from all its other neighbours.

Then all leaves can start sending messages, and there will be at least one node capable of sending until a message has been sent over each link in both directions. It has been proved that, when this has happened, then the tree is in the stable stage.

Junction trees

If the network is not a tree, the message passing scheme above is not satisfactory. Examples are found where a stable situation cannot be reached after a finite number of message passings. Also examples are found, where the message passing scheme converges, but the distributions in the limit are grossly incorrect.

The main reason why message passing works for trees is the independence properties: Take any node X. If X is instantiated then all its neighbours are independent. This does not hold for networks in general.

The HUGIN updating method is similar to the tree propagation above. Only, the nodes in the tree are sets of variables rather than single variables. Through graph theoretical methods, the independence properties in the network are analysed to establish a set of sets of variables (clusters) and to construct a tree over the clusters. This resulting tree has the *junction tree property*: for each pair V, W of nodes, all nodes on the path between V and W contains their intersection $V \cap W$. In Figure 11 we give an example of a network and a corresponding junction tree for probability updating.

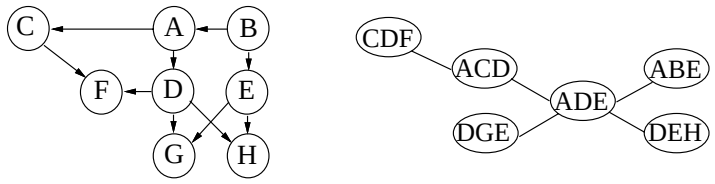

Figure 11: A Bayesian network and a corresponding junction tree. Note that if the nodes A, D and E are instantiated, then (C, F), (B), (G), and (H) are d-separated for all evidence scenaria.

Software

Readers interested in building Bayesian networks and experiencing with belief updating can do this without caring about the mathematics behind. Software exists for editing and running Bayesian networks. In *http://bayes.stat.washington.edu/almond/belief.html* a list of available software can be found, and a demo-version of HUGIN is available through *http://www.hugin.dk*.

4 Examples of systems based on Bayesian networks

Below we have listed references for normative systems. The list is not an attempt to be complete, it merely should give an impression of the variety of domains.

Agriculture

BOBLO is a system which helps in verification of the parentage of Jersey cattle through blood type identification (Rasmussen 1995b) and (Rasmussen 1995a).

A system for mildew control in winter wheat has been constructed by Jensen (1995).

Computer vision

Binford and colleagues made an early use Bayesian networks for interpretation of images (Binford, Levitt & Mann 1988) and (Levitt, Binford & Ettinger 1989). Jensen, Christensen & Nielsen (1992) and Rimey & Brown (1994) use Bayesian networks for control of computer resources in the interpretation process, and Munck-Fairwood (1992) uses a Bayesian network for 3D inference form 2D data. The Bayesian network approach to image interpretation has been used by Levitt, Hedgcock, Dye, Johnston, Shadle & Vosky (1993) for segmentation of computed radiographs of the hand.

Computing

The PC-operating system *Windows 95* incorporates a normative system for troubleshooting printing problems. The principles behind it are described in (Heckerman, Breese & Rommelse 1995).

DAACS is a system for software debugging (Burnell & Horvitz 1995).

Information processing

VISTA is a system used by NASA when launching space shuttles. It is used for filtering and displaying information on the propulsion system (Horvitz & Barry 1995).

(Bruza & van der Gaag 1993) developed a language for constructing Bayesian networks for information retrieval, and Fung & Favero (1995) describe another system for information retrieval.

Medicine

Child helps in diagnosing congenital heart diseases, (Franklin, Spiegelhalter, Macartney & Bull 1989), and (Lauritzen, Thiesson & Spiegelhalter 1994).

MUNIN is a system for obtaining a preliminary diagnosis of neuromuscular diseases on the basis of electromyografic findings (Andreassen et al. 1989).

Painulim diagnoses neuromuscular diseases (Xiang, Pant, Eisen, Beddoes & Poole 1993).

Pathfinder assists community pathologists with the diagnosis of lymph-node pathology (Heckerman, Horvitz & Nathwani 1992), (Heckerman & Nathwani 1992b), and (Heckerman & Nathwani 1992a). Pathfinder has been integrated with videodiscs to the commercial system *Intellipath* (Nathwani, Heckerman, Horvitz & Lincoln 1990).

SWAN is a system for insulin dose adjustment of diabetes patients (Andreassen, Hovorka, Benn, Olesen & Carson 1991), and (Hejlesen, Andreassen & Andersen 1993).

Miscellaneous

Hailfinder was developed for forecasting severe weather in the plane of northeastern Colorado (Abramson, Brown, Edwards, Murphy & Winkler 1996).

FRAIL is an automatic Bayesian network system (Goldman & Charniak 1993). It has been developed for building Bayesian networks for interpretation of written prose (Charniak & Goldman 1991).

References