11 Minimum Message Length and Generalized Bayesian Nets with Asymmetric Languages

Joshua W. Comley
School of Computer Science and Software Engineering
Monash University
Clayton Campus
Victoria 3800
Australia

David L. Dowe
School of Computer Science and Software Engineering
Monash University
Clayton Campus
Victoria 3800
Australia

This chapter describes the minimum message length (MML) principle, including its relationship to the subsequent minimum description length (MDL) principle. A brief discussion of the history and development of MML is given, including 'strict MML' (SMML) and some of its approximations. After addressing some common misconceptions about MML, we present a novel application of MML to the inference of generalized Bayesian networks, using decision trees to supply conditional probability distributions. Unlike many Bayesian network applications, the new generalized Bayesian networks presented in this chapter are capable of modeling a combination of discrete and continuous attributes. This demonstrates the power of information-theoretic approaches, such as MML, which are able to function over both discrete probability distributions and continuous probability densities. Furthermore, we give examples of asymmetric languages in which the desired target attribute is best modeled implicitly rather than as an explicit output attribute. Last, we provide some preliminary results and suggest several possible directions for further research.

11.1 Introduction

Minimum message length (MML) is an information-theoretic Bayesian principle of inductive inference, machine learning, statistical inference, econometrics, and "data mining" which was developed by Chris Wallace and David Boulton in a series of six journal papers from 1968 to 1975, including several explicit articulations of the MML principle (see, e.g., [Wallace and Boulton 1968, p. 185, sec. 2]; [Boulton and Wallace 1970, p. 64, col. 1]; [Boulton and Wallace 1973, sec. 1, col. 1]; [Boulton and Wallace 1975, sec. 1, col. 1]; [Wallace and Boulton 1975, sec. 3]). (David Boulton then published his Ph.D. thesis [Boulton 1975] in this area.)

Given a data set, D, we wish to find the most probable hypothesis, H — that is, that which maximizes $\mathrm{P}(H|D)$. By Bayes' theorem, the posterior probability of H is the product of the prior probability of H and the likelihood function of D given H divided by the marginal probability of the observed data, D — that is, $\mathrm{P}(H|D) = (1/\mathrm{P}(D)) \times \mathrm{P}(H) \times \mathrm{P}(D|H)$, where the marginal probability $\mathrm{P}(D)$ is given by $\mathrm{P}(D) = \sum_H \mathrm{P}(H) \cdot \mathrm{P}(D|H)$ or $\mathrm{P}(D) = \int_H \mathrm{P}(H) \cdot \mathrm{P}(D|H)\, dH$. Recall from elementary information theory that an event of probability p_i can be optimally encoded by a code word of length $l_i = -\log p_i$. Because $\mathrm{P}(D)$ is a function of D independent of the hypothesis, H, maximising $\mathrm{P}(H|D)$ is equivalent to maximising the product of the two probabilities $\mathrm{P}(H) \times \mathrm{P}(D|H)$, which is in turn equivalent to minimising $-\log \mathrm{P}(H) - \log \mathrm{P}(D|H)$, the length of a two-part message transmitting, first, H and then D given H (see, e.g., [Wallace and Boulton 1968, p. 185, sec. 2]; [Boulton and Wallace 1970, p. 64, col. 1]; [Boulton and Wallace 1973, sec. 1, col. 1]; [Boulton and Wallace 1975, sec. 1, col. 1]; [Wallace and Boulton 1975, sec. 3]).

In the remainder of this chapter, we define strict MML (SMML) and then deal with several issues pertaining to MML. These include dealing with some common (unfortunate) misconceptions in the literature about MML, Kolmogorov complexity, Bayesianism, statistical invariance, and statistical consistency. We also present in Section 11.3.1 a conjecture [Dowe, Baxter, Oliver, and Wallace 1998, p. 93]; [Wallace and Dowe 1999a, p. 282]; [Wallace and Dowe 2000, p. 78] of David Dowe's relating some of these concepts. In Section 11.4, we mention the issue of inference vs. prediction and the merits of logarithmic scoring in probabilistic prediction. We tersely (due to space constraints) survey some MML literature, relate it to our understanding of current minimum description length (MDL) writings and raise the issue of MML as a universal principle of Bayesian inference. Given the many (unfortunate) misconceptions some authors seem to have about the extensive MML literature and its original concepts, and given the above-mentioned historical precedence of MML over MDL, we have cited several instances where — at least at the time of writing — MML is apparently state-of-the-art.

Subsequently and in Section 11.4.4, we then discuss comparatively new work on the second author's notion of inverse learning (or implicit learning) [Dowe and Wallace 1998; Comley and Dowe 2003] and the first author's refinements thereof [Comley and Dowe 2003], including setting the asymmetric languages in

a framework of generalized Bayesian networks and investigating search algorithms. We believe that this is an advance of much substance and potential.

11.2 The Strict Minimum Message Length Principle

The strict minimum message length (strict MML, or SMML) principle was first introduced in [Wallace and Boulton 1975], from which we largely borrow in this section. The relationship of strict MML with algorithmic information theory is given in [Wallace and Dowe 1999a], and various other descriptions and applications of it are given in [Wallace and Freeman 1987; Wallace 1996; Wallace and Dowe 1999b; Farr and Wallace 2002; Fitzgibbon, Dowe, and Allison 2002a].

A point estimation problem is a quadruple $\{H, X, h, f\}$ such that H is a parameter space (assumed to be endowed with a field of subsets), X is a set of possible observations, and h is a given prior probability density function with respect to a measure, $d\theta$, on the parameter space H such that $\int_H h(\theta)\ d\theta = 1$.

f is the known conditional probability function $f : (X, H) \to [0, 1] : f(x, \theta) = f(x|\theta)$, where $\sum_i f(x_i|\theta) = 1$ for all $\theta \in H$.

A solution to a point estimation problem is a function $m : X \to H$ with $m(x) = \theta$. Recalling from Section 11.1, that $r(x) = \int_H h(\theta) f(x|\theta)\ d\theta$ is the marginal probability of a datum, x, we note that $\sum_{x \in X} r(x) = 1$ and that the posterior distribution, $g(\cdot|\cdot)$, is given by $g(\theta|x) = h(\theta) \cdot f(x|\theta) / \int_H h(\theta) \cdot f(x|\theta) d\theta = h(\theta) f(x|\theta)/r(x)$.

We assume that the set, X, of possible observations is countable. (We suspect the even stronger result(s) that it is probably even recursively enumerable and perhaps even recursive.) Given that X is countable, so, too, is $H^* = \{m(x) : x \in X\}$, that is, we can say $H^* = \{\theta_j : j \in N\}$. We can then define $c_j = \{i : m(x_i) = \theta_j\}$ for each $\theta_j \in H^*$, and $C = \{c_j : \theta_j \in H^*\}$. Given some fixed H^* as just defined, we assign finite prior probabilities $q_j = \sum_{i \in c_j} r(x_i) = \sum_{i:m(x_i)=\theta_j} r(x_i)$ to the members θ_j of H^* and then, for each $x \in X$, we choose $m(x)$ to be that $\theta \in H^*$ which maximize $p(x|h)$ and in turn minimizes the expected length of the codebook (given this H^*).

This defines $m^* : x \to H^*$, which we can then take to be our solution to $\{H, X, h, f\}$, provided that we have indeed selected the correct H^*.

For each c_j we choose the point estimate θ_j to minimize $-\sum_{x_i : i \in c_j} r(x_i) \cdot \log f(x_i|\theta_j)$. For each H^* the average two-part message length is

$$\Big(-\sum_j (q_j \cdot \log q_j) \Big) + \Big(-\sum_j \sum_{i \in c_j} \big(q_j \cdot \frac{r(x_i)}{q_j} \cdot \log f(x_i|\theta_j) \big) \Big).$$

In essence the larger the data groups the shorter the average length of the first part of the message but the longer the average length of the second part. We choose the c_j to minimize the expected two-part message length of the codebook. Having thus chosen the codebook given datum x, the SMML estimate is the θ_j representing the code block including x.

11.3 Invariance and Consistency of MML, and Some Common Misconceptions

11.3.1 Maximum a Posteriori (MAP) and MML

One common misconception among some authors is that the MML estimate is supposedly the same as the posterior mode – or maximum a posteriori (MAP) – estimate. To the contrary, when dealing with continuous distributions, MAP maximizes the posterior *density* (not a probability) [Wallace and Boulton 1975, p. 12]; [Wallace and Dowe 1999a, p. 279 sec. 6.1.1]; [Wallace and Dowe 1999c, p. 346]; [Wallace and Dowe 2000, secs. 2 and 6.1] and is typically not invariant, whereas MML maximizes the posterior *probability* and is invariant[Wallace and Boulton 1975] [Wallace and Freeman 1987, p, 243]; [Wallace 1996, sec. 3.5 and elsewhere]; [Dowe, Baxter, Oliver, and Wallace 1998, secs. 4.2 and 6]; [Wallace and Dowe 1999a, secs. 6.1 and 9]; [Wallace and Dowe 1999c, secs. 1 and 2]; [Wallace and Dowe 2000, p. 75, sec. 2 and p. 78–79]. A method of parameter estimation is said to be (statistically) invariant if for all one-to-one transformations t, $t(\hat{\theta}) = t(\hat{\theta})$, that is, the point estimate in the transformed parameter space is equal to the transformation of the original point estimate.

For further cases highlighting the difference between MML and MAP which also show MML outperforming MAP, see, for example, [Dowe, Oliver, Baxter, and Wallace 1995; Dowe, Oliver, and Wallace 1996], (for polar and cartesian coordinates on the circle and sphere respectively), and [Wallace and Dowe 1999b, secs. 1.2 and 1.3].

MAP and MML when all attributes are discrete On some occasions, all attributes are discrete – such as if we were interested only in the topology of a decision tree and the attributes which were split on (and possibly also the most likely class in each leaf [Quinlan and Rivest 1989]) without being interested in the additional inference of the class probabilities [Wallace and Patrick 1993; Tan and Dowe 2002; Comley and Dowe 2003; Tan and Dowe 2003] in each leaf. In these cases, where all attributes are discrete, like MML, MAP will also maximize a probability rather than merely a density. For many MML approximations, in these cases, both MAP and MML will optimize the same objective function and obtain the same answer. It is a subtle point, but *even in these* cases, MAP will generally be different from the strict MML inference scheme [Wallace and Boulton 1975; Wallace and Freeman 1987; Wallace and Dowe 1999a] of Sections 11.2 and 11.3.3 (which partitions in data space) and to the 'fairly strict MML' scheme (which is similar to strict MML but instead partitions in parameter space). The subtle point centers on the fact that the construction of the strict (or fairly strict) codebook is done in such a way as to minimize the expected message length [Wallace and Dowe 1999a, sec. 6.1]. Consider two distinct hypotheses available as MAP inferences which happen to be very similar (e.g., in terms of Kullback-Leibler distance) and suppose – for the sake of argument – that they have almost identical prior probability. If these are

merged into one, the prior probability of the resultant hypothesis will have about twice the prior probability – resulting in its being about 1 bit cheaper in the first part of the message – than either of the unmerged alternatives. If the expected additional cost to the second part of the message from the merging is more than compensated for by the expected saving from the first part, then such a merging would take place in the construction of the MML codebook. So, we recall from Section 11.3.1 that MAP is different from MML when continuous-valued attributes and probability densities are involved. But – as we have just explained – even in the rare case when all attributes are discrete and only probabilities (and no densities) are involved, then – although some approximations to MML would yield MAP – we *still* find that MML is generally different from MAP. Whereas the MAP, maximum likelihood (ML), and Akaike's information criterion (AIC) estimates are statistically inconsistent for a variety of parameter estimation problems (e.g., Neyman-Scott [Dowe and Wallace 1997] and linear factor analysis [Wallace 1995; Wallace and Freeman 1992]), the two-part structure of MML messages leads to MML's general statistical consistency results [Barron and Cover 1991; Wallace 1996]; [Wallace and Freeman 1987, Sec. 2, p 241]. We note in passing Dowe's related question [Dowe, Baxter, Oliver, and Wallace 1998, p. 93]; [Wallace and Dowe 1999a, p. 282]; [Wallace and Dowe 2000, p. 78] as to whether only (strict) MML and possibly also closely related Bayesian techniques (such as minimising the expected Kullback-Leibler distance [Dowe, Baxter, Oliver, and Wallace 1998]) can generally be both statistically invariant and statistically consistent.

11.3.2 "The" Universal Distribution and Terms "of Order One"

In its most general sense (of being akin to Kolmogorov complexity or algorithmic information-theoretic complexity), MML uses the priors implicit in the particular choice of universal Turing machine [Wallace and Dowe 1999a, sec. 7 and elsewhere]. We agree with other authors [Rissanen 1978, p. 465]; [Barron and Cover 1991, sec. IV, pp. 1038–1039]; [Vitanyi and Li 1996][Li and Vitanyi 1997, secs 5.5 and 5.2]; [Vitanyi and Li 2000] about the relevance of algorithmic information theory (or Kolmogorov complexity) to MDL and MML. However, a second common misconception is either an implicit assumption that there is one unique universal distribution, or at least something of a cavalier disregard for quantifying the (translation) terms "of order one" and their relevance to inference and prediction. We note that there are countably infinitely many distinct universal Turing machines and corresponding universal (prior) distributions. As such, the relevance of the Bayesian choice of prior or Turing machine or both should be properly understood [Wallace and Dowe 1999a, secs. 2.4 and 7].

11.3.3 Strict MML Codebook, Expected Length and Actual Lengths

A third, related, common misconception concerns the construction of the MML codebook in SMML. Given the likelihood function(s) and the Bayesian prior(s),

without having seen any data, we construct the MML codebook as in Section 11.2 so as to minimize the expected length of the two-part message [Wallace and Boulton 1975, secs. 3.1 - 3.3]; [Wallace and Freeman 1987, sec. 3]; [Wallace 1996]; [Wallace and Dowe 1999a, secs. 5 and 6.1]; [Wallace and Dowe 1999b, secs. 1.2 and 1.3]. This typically results in coding blocks which are partitions of the *data* space. (One can only record countably different measurement values, and it is reasonable to assume that any continuous-valued measurement is made to some accuracy, ϵ. As such, its value can be encoded with a finite code length.) With the MML codebook now thus chosen, given data D, we choose hypothesis, H, so as to minimize the length of the two-part message. This should clarify that common misconception about SMML and the MML codebook. The strict MML principle has been applied to problems of binomial distributions [Wallace and Boulton 1975, sec. 5]; [Wallace and Freeman 1987, sec. 3]; [Farr and Wallace 2002] and a restricted cut-point segmentation problem [Fitzgibbon et al. 2002a] (which, like the Student T distribution, would appear to have no sufficient statistics other than the data themselves), but is generally computationally intractable. In practice, we consider approximations to a partitioning of the *parameter* space, such as the invariant point estimator of [Wallace and Freeman 1987, sec. 5, a usable estimator]; [Wallace and Dowe 1999a, sec. 6.1.2, practical MML]; [Wallace and Dowe 1999c, p. 346, col. 2] – and sometimes others, as discussed below.

Tractable Approximations to Strict MML The invariant point estimator of [Wallace and Freeman 1987, sec. 5, a usable estimator]; [Wallace and Dowe 1999a, sec. 6.1.2, practical MML]; [Wallace and Dowe 1999c, p. 346, col. 2] is based on a quadratic approximation to the Taylor expansion of the log-likelihood function and the assumption of the prior being approximately locally uniform. Despite the many and vast successes (e.g. [Wallace and Dowe 1993; Dowe, Oliver, and Wallace 1996; Dowe and Wallace 1997; Oliver and Wallace 1991; Oliver 1993; Tan and Dowe 2002; Tan and Dowe 2003; Wallace and Dowe 2000; Edgoose and Allison 1999; Wallace and Korb 1999; Baxter and Dowe 1996; Wallace 1997; Vahid 1999; Viswanathan and Wallace 1999; Fitzgibbon, Dowe, and Allison 2002b; Comley and Dowe 2003]) of the Wallace and Freeman [1987] approximation [Wallace and Freeman 1987, sec. 5]; [Wallace and Dowe 1999a, sec. 6.1.2], it *is* an approximation, and its underlying assumptions are sometimes strained [Wallace and Dowe 1999c, p. 346, col. 2] (or at least appear to be [Grünwald, Kontkanen, Myllymaki, Silander, and Tirri 1998]) – leading us to new approximations. These include Dowe's invariant MMLD approximation and Fitzgibbon's modification(s) [Fitzgibbon et al. 2000a,b], Wallace's numerical thermodynamic entropy approximation [Wallace 1998], and others (e.g., [Wallace and Freeman 1992; Wallace 1995]).

We note that the standard deviation σ of measurements of accuracy ϵ (as in Section 11.2 and 11.3.3) is generally assumed ([Wallace and Dowe 1994, sec. 2.1]; [Comley and Dowe 2003, sec. 9]) to be bounded below by 0.3ϵ or $\epsilon/\sqrt{12}$.

11.4 MML as a Universal Principle, Prediction and MDL

11.4.1 Brief History of Early MML Papers, 1968–1975

These first six MML papers [Wallace and Boulton 1968; Boulton and Wallace 1969, 1970, 1973, 1975; Wallace and Boulton 1975] from Section 11.1 and David Boulton's 1975 Ph.D. thesis [Boulton 1975] were variously concerned with univariate or multivariate multinomial distributions [Boulton and Wallace 1969; Wallace and Boulton 1975], multivariate Gaussian distributions, mixture modeling (or clustering or cluster analysis or intrinsic classification or unsupervised learning) of such distributions [Wallace and Boulton 1968; Boulton and Wallace 1970; Boulton and Wallace 1975; Boulton 1975] and even hierarchical mixture modeling of such distributions [Boulton and Wallace 1973]. We also see the introduction [Boulton and Wallace 1970, p. 63] of the (term) *nit*, where 1 nit = $\log_2 e$ bits. The nit has also been referred to as a *nat* in subsequent MDL literature, and was known to Alan M. Turing as a *natural ban* (see, e.g., [Hodges 1983, pp. 196–197] for *ban*, *deciban*, and *natural ban*). These units can be used not only to measure message length and description length but also to score probabilistic predictions.

11.4.2 Inference, Prediction, Probabilistic Prediction and Logarithmic Scoring

Two equivalent motivations of MML are, as given in Section 11.1, (1) to maximize the posterior *probability* (*not* a density – see Section 11.3.1) and, equivalently, (2) to minimize the length of a two-part message.

The second interpretation (or motivation) of MML can also be thought of in terms of Occam's razor [Needham and Dowe 2001]. Both these interpretations can be thought of as inference to the best (single) explanation. Prediction [Solomonoff 1964, 1996, 1999] is different from inductive inference (to the best explanation) in that prediction entails a weighted Bayesian averaging of all theories, not just the best theory [Solomonoff 1996; Dowe, Baxter, Oliver, and Wallace 1998]; [Wallace and Dowe 1999a, sec. 8]; [Wallace and Dowe 1999c, sec. 4]. Thus, despite the many successes described in this chapter, MML is not directly concerned with prediction.

Probabilistic Prediction and Logarithmic Scoring A prediction which gives only a predicted class (e.g., class 2 is more probable than class 1) or mean (e.g., $\hat{\mu} = 5.2$) conveys less information than one giving a probability distribution — such as $(\hat{p}_1 = 0.3, \hat{p}_2 = 0.7)$ or $N(\hat{\mu} = 5.2, \hat{\sigma}^2 = 2.1^2)$ — whereas a probabilistic prediction, such as $(0.3, 0.7)$, also gives us the predictively preferred class (class 2) as a byproduct. To paraphrase it more bluntly, the current literature could be said to contain all too many methods endeavoring to tune their "right"/"wrong" predictive accuracy with scant regard to any probabilistic predictions. A first obvious shortcoming of such an approach will be its willingness to "find" (or "mine")

spurious patterns in data which is nothing more than 50% : 50% random noise.

One criterion of scoring functions for probabilistic predictions is that the optimal expected long-term return should be gained by using the true probabilities, if known. The logarithmic scoring function achieves this, and has been advocated and used for binomial [Good 1952; Good 1968; Dowe, Farr, Hurst, and Lentin 1996]; [Vovk and Gammerman 1999, sec. 3], multinomial [Dowe and Krusel 1993]; [Tan and Dowe 2002, sec. 4]; [Tan and Dowe 2003, sec. 5.1], and other distributions (e.g., Gaussian [Dowe et al. 1996]). Interestingly, Deakin has noted several cases of scoring functions other than logarithmic achieving this criterion for multinomial distributions [Deakin 2001]. Nonetheless, we prefer the logarithmic scoring function for the added reason of its relation to log-likelihood, the sum of the logarithms of the probabilities being the logarithm of the product of the probabilities, in turn being the logarithm of the joint probability.

The predictive estimator which minimizes the expected (negative) log-likelihood function is known as the minimum expected Kullback-Leibler distance (MEKLD) estimator [Dowe et al. 1998]. Theoretical arguments [Solomonoff 1964; Dowe et al. 1998] and intuition suggest that the SMML estimator (recall Sections 11.2 and 11.3.3) will come very close to minimising the expected Kullback-Leibler distance.

11.4.3 MML, MDL, and Algorithmic Information Theory

The relation between MML [Wallace and Boulton 1968; Wallace and Freeman 1987; Wallace and Dowe 1999a] and MDL [Rissanen 1978, 1987, 1999b] has been discussed in [Wallace and Freeman 1987; Rissanen 1987] and related articles in a 1987 special issue of the *Journal of the Royal Statistical Society*, in [Wallace and Dowe 1999a,b,c; Rissanen 1999a,b,c] and other articles [Dawid 1999; Clarke 1999; Shen 1999; Vovk and Gammerman 1999; Solomonoff 1999], in a 1999 special issue of the *Computer Journal*, and elsewhere. For a discussion of the relationship between strict MML (SMML) (see Sections 11.2 and 11.3.3) and the work of Solomonoff [1964], Kolmogorov [1965], and Chaitin [1966], see [Wallace and Dowe 1999a].

We reiterate the sentiment [Wallace and Dowe 1999c] that, in our opinion, MDL and MML agree on many, many points. We also acknowledge that from the perspective of someone who knew relatively little about MDL and MML, the disagreements between MDL and MML would appear to be both infrequent and minor [Wallace and Dowe 1999c]. Having said that, we now venture to put forward some concerns about some MDL coding schemes.

Efficiency and Reliability of Coding Schemes and of Results Recalling Section 11.4.2, the predictive reliability of MDL and MML will depend very much upon the coding schemes used. In [Quinlan and Rivest 1989; Wallace and Patrick 1993] and [Kearns, Mansour, Ng, and Ron 1997; Viswanathan, Wallace, Dowe, and Korb 1999], we respectively see a decision tree inference problem and a problem of segmenting a binary process in which the original coding schemes [Kearns et al. 1997; Quinlan and Rivest 1989] had their results improved upon by corresponding

improvements in the relevant coding schemes [Viswanathan et al. 1999; Wallace and Patrick 1993]. Reinterpreting the Occam's razor measure of decision tree simplicity from the node count in [Murphy and Pazzani 1994] to a message length measure in [Needham and Dowe 2001] likewise gives improved results.

While the principle and spirit of the 1978 MDL coding scheme [Rissanen 1978] live on, it is generally acknowledged in more recent MDL writings and elsewhere (see, e.g., [Wallace and Dowe 1999a, p. 280, col. 2]) to have been substantially improved upon.

In conclusion, we ask the reader wishing to use an MDL or MML coding scheme to note that the reliability of the results will be highly dependent upon the reliability of the coding scheme.

Further Comments on Some Other MDL Coding Schemes While MML is openly and subjectively Bayesian and known to be so, many often either assert that MDL is Bayesian or ask whether or not it is (see, e.g., [Vitanyi and Li 1996]; [Dawid 1999, p. 323, col. 2, sec. 4, sec. 5]; [Clarke 1999, sec. 2]; [Vitanyi and Li 2000]). Some would contend that a parameter space restriction [Rissanen 1999b, p. 262, col. 2] was also invoking a prior — namely, that the values of the parameters cannot lie in the prohibited area (cf. [Dawid 1999, p. 325, col. 2]).

The Jeffreys 'prior' [Jeffreys 1946] uses the Fisher information as though it were a Bayesian prior, thus depending upon the sensitivity of the measuring instruments and observational protocol used to obtain the data [Lindley 1972; Bernardo and Smith 1994]; [Dowe, Oliver, and Wallace 1996, p. 217]. This would appear to be able to lead to situations where the prior beliefs one uses in modeling the data depend upon the strength or location of the measuring instrument [Dowe, Oliver, and Wallace 1996, p. 217]; [Wallace and Dowe 1999a, sec. 2.3.1] (see also [Wallace and Freeman 1987, Sec. 1, p241]; [Wallace and Dowe 1999a, sec. 5, p. 277, col. 2] for other concerns). The Jeffreys 'prior' has been used in comparatively recent MDL work [Rissanen 1996a,b], raising some of the above concerns. The Jeffreys 'prior' also does not always normalise (e.g., [Wallace and Dowe 1999b, secs. 2.3.2 – 2.3.4]). We understand that Liang and Barron [2005] and Lanterman [2005] partially address this problem. The notion of complete coding in MDL [Rissanen 1996a; Dom 1996]; [Grünwald, Kontkanen, Myllymaki, Silander, and Tirri 1998, sec. 4] would appear to be in danger of contravening the convergence conditions of the two-part message form from [Barron and Cover 1991].

Some other comments on some MDL coding schemes and suggested possible remedies for some of the above concerns are given in [Wallace and Dowe 1999c, sec. 2] and [Wallace and Dowe 1999b, sec. 3].

11.4.4 MML as a Universal Principle

The second author founded and (in 1996) chaired the Information, Statistics and Induction in Science (ISIS) conference (see, e.g., [Rissanen 1996b; Solomonoff 1996; Wallace 1996; Vitanyi and Li 1996; Dowe and Korb 1996]) because of a belief in the

universal relevance of MML to problems in induction and the philosophy of science. Recalling Section 11.4.3, we suspect that the editors and perhaps also many of the other authors in this book might well have similar beliefs.

The relevance of MML to inductive inference is clear, but let us summarize. MML has been used for parameter estimation for a variety of distributions [Wallace and Boulton 1968; Boulton and Wallace 1969; Wallace and Dowe 1993; Dowe, Oliver, and Wallace 1996; Dowe and Wallace 1997; Wallace and Dowe 2000], and for supervised learning [Wallace and Patrick 1993; Oliver and Wallace 1991; Oliver 1993; Vahid 1999; Tan and Dowe 2002; Tan and Dowe 2003] and unsupervised learning (clustering or mixture modeling) [Wallace and Boulton 1968; Wallace 1986; Wallace and Dowe 1994; Wallace and Dowe 2000; Edgoose and Allison 1999], hierarchical clustering [Boulton and Wallace 1973], inference of probabilistic finite state automata (PFSAs) or hidden Markov models (HMMs) [Wallace and Georgeff 1983], Markov models of clustering [Edgoose and Allison 1999], linear and polynomial regression [Baxter and Dowe 1996; Wallace 1997; Viswanathan and Wallace 1999; Vahid 1999; Fitzgibbon et al. 2002b], segmentation problems [Viswanathan et al. 1999; Fitzgibbon et al. 2002a], factor analysis [Wallace and Freeman 1992; Wallace 1995], clustering with factor analysis [Edwards and Dowe 1998], and so on. It should be added that the success of MML in some of the above problems is emphatic. We also recall the statistical invariance and statistical consistency of MML from Section 11.3.

MML or closely related work has also been applied to genome analysis [Allison, Wallace, and Yee 1990; Dowe, Oliver, Dix, Allison, and Wallace 1993; Dowe, Allison, Dix, Hunter, Wallace, and Edgoose 1996; Edgoose, Allison, and Dowe 1996], psychology [Kissane, Bloch, Dowe, R.D. Snyder andP. Onghena, and Wallace 1996], causal networks [Wallace and Korb 1999], Bayesian networks (sec. 11.4.4 in this book and [Comley and Dowe 2003]), Goodman's "Grue" paradox [Solomonoff 1996], financial market (in)efficiency [Dowe and Korb 1996], and cognitive science and IQ tests [Dowe and Hajek 1998; Hernandez-Orallo and Minaya-Collado 1998; Dowe and Oppy 2001]; [Sanghi and Dowe 2003, sec. 5.2].

We now proceed throughout the remainder of this chapter to discuss comparatively new work on the second author's notion of inverse learning (or implicit learning) [Dowe and Wallace 1998; Comley and Dowe 2003] and the first author's refinements thereof [Comley and Dowe 2003], including setting the asymmetric languages in a framework of generalized Bayesian networks and investigating search algorithms. sectionMML, Generalized Joint Distributions, and Implicit Learning

11.4.5 Generalized Bayesian Networks

We now describe an application of MML to the inference of generalized Bayesian networks. This is an extension of the idea of 'inverse learning' or 'implicit learning' proposed by Dowe in [Dowe and Wallace 1998], and further developed by Comley in [Comley and Dowe 2003]. In this domain we deal with multivariate data, where each item X (also known as a thing, case, or record) has k attributes (also referred to as

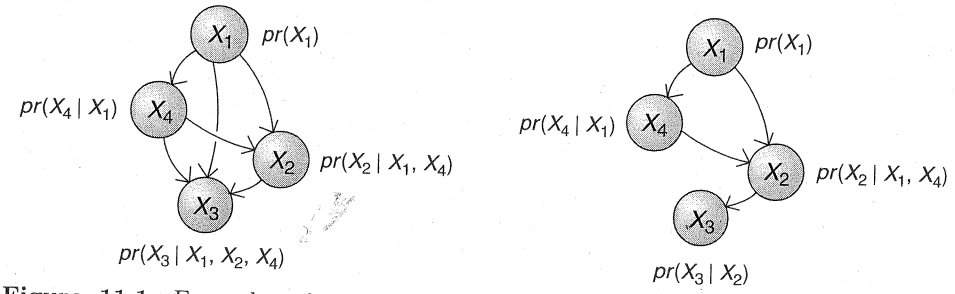

Figure 11.1 Examples of Bayesian network structures, illustrating the probability distributions supplied in each node. On the left is a fully connected network, while the network on the right is partially connected. Notice here that X_3 is conditionally independent of X_1 and X_4 given X_2.

variables or fields), denoted here as $X_1, \ldots, X_k$. We wish to model the statistical relationships between attributes when presented with a set of n such data. We may want to do this to be able to predict one of the attributes when given values for the others, or simply because we are interested in the interattribute correlations.

The graphical structure of Bayesian networks makes them an intuitive and easily interpreted representation of the relationships between attributes. A Bayesian network is a directed acyclic graph (DAG) with one node corresponding to each attribute. Each node provides a conditional probability distribution of its associated attribute given the attributes associated with its parent nodes. Figure 11.1 shows example network structures for $X = \{X_1, X_2, X_3, X_4\}$. For a general introduction to Bayesian network theory, see [Russell and Norvig 1995, chap. 15, sec. 5].

Bayesian networks model the joint distribution over all attributes, and express this as a product of the conditional distributions in each node. In the case of a fully connected Bayesian network (see Figure 11.1), the joint distribution $\mathrm{P}(X_1 \& X_2 \& \ldots \& X_k)$ is modeled as $\mathrm{P}(X_1) \cdot \mathrm{P}(X_2|X_1) \ldots \mathrm{P}(X_k|X_1, \ldots, X_{k-1})$. In practice, however, Bayesian networks are rarely fully connected, and make use of conditional independencies to simplify the representation of the joint distribution (see Figure 11.1).

Although — in the abstract sense — the conditional probability distribution of a node can take any form at all, many Bayesian network methods simply use conditional probability tables, and are limited by the restriction that all attributes must be discrete. Others [Scheines, Spirtes, Glymour, and Meek 1994; Wallace and Korb 1999] model only continuous attributes, describing an attribute as a linear combination of its parent attributes. Here we show how information-theoretic approaches like MML can be used, together with decision tree models, to build a general class of networks able to handle many kinds of attributes. We use decision trees to model the attribute in each node, as they tend to be a compact and powerful representation of conditional distributions, and are able to efficiently express context-specific independence [Boutilier, Friedman, Goldszmidt, and Koller 1996]. It should be noted that any conditional model class could be used, so long

as MML message lengths can be formulated for it. An MML coding scheme for basic decision trees is given in [Wallace and Patrick 1993], which refines an earlier coding scheme suggested in [Quinlan and Rivest 1989]. A variant of this scheme is summarized in Section 11.4.7.

11.4.6 Development and Motivation of Implicit Learning

The idea of implicit learning (or inverse learning) by MML presented here builds on material originally proposed by Dowe in [Dowe and Wallace 1998]. That work involved only two attributes, or at most two groups of attributes. Comley [Comley and Dowe 2003] later refined the implicit learning MML coding scheme (given in Section 11.4.7) and generalized the idea to handle more than two attribute groups, relating it to Bayesian networks.

Although the two-attribute case is a simple one, it provides informative examples of the benefits of implicit learning. The idea is that we have a class of conditional models that we are comfortable with and know how to use. We can use this to accurately model one attribute X_1 as a probabilistic function of the other attribute, X_2. But imagine it is actually X_2 that we wish to predict, given X_1. Using Bayes' rule, and coupling our model of $\mathrm{P}(X_1|X_2)$ with a 'prior' model of $\mathrm{P}(X_2)$, we can form a model of the joint distribution $\mathrm{P}(X_1\&X_2) = \mathrm{P}(X_2) \cdot \mathrm{P}(X_1|X_2)$. By taking a cross-sectional 'slice' from this composite joint model, we can then extract the conditional probability $\mathrm{P}(X_2|X_1)$.

For example, take the case where X_1 and X_2 are both continuous variables, where X_2 is generated from the Gaussian distribution $N(10, 1)$ and X_1 is in turn generated from $(X_2)^3 + N(0, 1)$. Suppose our model language is the class of univariate polynomials of the form

$X_2 = a_0 + a_1X_1 + a_2(X_1)^2 + a_3(X_1)^3 + \cdots + a_d(X_1)^d + N(0, \sigma^2)$ for some degree d

and we wish to predict X_2 given X_1. If such a technique were to model X_2 as an *explicit* probabilistic function of X_1, it could not express — let alone discover — the true conditional relationship $X_2 = \big(X_1 + N(0, 1)\big)^{\frac{1}{3}}$, as this is outside its model language. However we can use the same model language to *implicitly* state X_2's dependence on X_1 using the joint distribution and Bayes' rule as follows:

$$\mathrm{P}(X_2|X_1) = \frac{\mathrm{P}(X_1\&X_2)}{\mathrm{P}(X_1)} \tag{11.1}$$

$$= \frac{\big(\mathrm{P}(X_2) \cdot \mathrm{P}(X_1|X_2)\big)}{\int_{z \in X_2^*} \mathrm{P}(z) \cdot \mathrm{P}(X_1|z)}, \tag{11.2}$$

where $\mathrm{P}(X_2)$ is given by $X_2 = 10 + N(0, 1)$, and $\mathrm{P}(X_1|X_2)$ is given by $X_1 = (X_2)^3 + N(0, 1)$. The point here is that the given model language cannot explicitly express $\mathrm{P}(X_2|X_1)$. It can, however, express both $\mathrm{P}(X_2)$ and $\mathrm{P}(X_1|X_2)$, which can be used together to define $\mathrm{P}(X_2|X_1)$ implicitly.

Many other circumstances exist where our target attribute is not necessarily best modeled as an explicit probabilistic function of the remaining attributes. Consider two continuous attributes, X_1 and X_2, which come from a two-dimensional mixture model [Wallace and Dowe 2000; McLachlan and Peel 2000]. While one could attempt to do a linear or polynomial regression of the target attribute, X_1, as a function of X_2, one would do best to acknowledge the mixture model and then model X_1 as a cross section (given X_2) of the mixture distribution. (Indeed, in this example X_1 and X_2 could equally well be groups of attributes [Dowe and Wallace 1998]). The point is that with a restricted model language one cannot always accurately estimate the desired conditional probability distribution, and it may be beneficial to implicitly model the target attribute by estimating the joint distribution. The generality of MML makes it an ideal tool for doing this. The consistency results of MML [Barron and Cover 1991; Wallace 1996; Wallace and Dowe 1999a; Wallace and Dowe 1999c], [Wallace and Freeman 1987, sec. 2, p. 241] suggest strongly that — quite crucially — it will converge to the best possible representation of the joint distribution.

The idea of implicit modeling was in fact first inspired by the problem of protein secondary structure prediction based on a known amino acid sequence. Learning a conditional model of the secondary structure sequence given the amino acid sequence is difficult, but the secondary structure sequence is far from random and can be easily modeled by itself. This model can be paired with a conditional model of the amino acids given the secondary structures, forming a joint distribution from which secondary structure can be predicted.

11.4.7 MML Coding of a General Bayesian Network

Recall from Section 11.1 the two-part format of the MML message - first stating the hypothesis H, then data D in light of this hypothesis. These two parts reflect a Bayesian approach where the cost of stating H is $-\log\big(\mathrm{P}(H)\big)$, $\mathrm{P}(H)$ being our prior belief that H is the true hypothesis, and D is transmitted using some optimal code based on the probabilities supplied by H. The H corresponding to the shortest overall message is chosen, as it maximizes the joint probability $\mathrm{P}(H\&D)$. Since D is held constant and we are only choosing from competing Hs this also corresponds to choosing the H with the highest posterior probability $\mathrm{P}(H|D)$. This subsection details how one might construct such a message for the general Bayesian networks proposed here.

The first part of the message, our hypothesis, must include the structure of the network — that is, the (partial) node ordering and connectivity — and the parameters required for the conditional probability distribution in each node. There are many possible ways in which one could do this; we describe one below.

We start by asking how many possible network structures there are. For k attributes there are $k!$ different fully connected structures (or total node orderings). But this does not take into account the number of partially connected networks. A total ordering has $\binom{k}{2} = (k^2 - k)/2$ directed arcs (each of which may or may

not be present in a partially connected network). So there are $2^{(k^2-k)/2}$ possible arc configurations for each of the $k!$ total orderings, leaving us with $k!\ 2^{(k^2-k)/2}$ possible network structures. We can assign each of these an equal prior probability of $\left(k!\ 2^{(k^2-k)/2}\right)^{-1}$.

Note now, though, that many of the partially connected structures will actually correspond to the *same* network (Figure 11.2). As we wish to choose between distinct networks it is important to treat these equivalent network representations as a single hypothesis. If we were to ignore this, the hypothesis' prior probability will be split among its equivalent representations, each of which would be inappropriately expensive. So from each group G of equivalent structures we choose one representative and assign it the prior probability of $c_G\left(k!\ 2^{(k^2-k)/2}\right)^{-1}$ where c_G is the cardinality of group G. This means that network structures with many equivalent representations are assigned a higher prior probability. Note that the coding scheme was chosen primarily for its simplicity, rather than being motivated by any belief that these structures are really more likely. For some applications it may be worth using a less 'biased' scheme, even if this is computationally more difficult.

Let us now calculate $len(S)$, the number of bits required to encode a network structure S (remembering that we are yet to transmit the conditional probability distribution parameters for each node).

$$len(S) = -\log_2\left(\mathrm{P}(S)\right) \tag{11.3}$$

$$= -\log_2\left(\frac{c_G}{k!2^{\frac{k^2-k}{2}}}\right) \tag{11.4}$$

$$= \log_2(k!) + \log_2\left(2^{\frac{k^2-k}{2}}\right) - \log_2(c_G) \tag{11.5}$$

$$= \log_2(k!) + \frac{k^2-k}{2} - \log_2(c_G) \tag{11.6}$$

Now that we have stated the node ordering and connectivity, we can transmit the parameters for the conditional distribution in each node. Nodes can express their conditional probability distributions using any of a wide variety of model classes — for example, conditional probability tables, polynomial regressions, and so on; here we use a rather general class of decision tree, described below.

The leaves of the tree may model either continuous-valued attributes using Gaussian density functions, or discrete-valued attributes using multistate distributions. Branch (test) nodes are capable of performing a binary split on a continuous-valued attribute (using a cut point) or a multiway split on a discrete-valued attribute (one subtree per possible value). Once a discrete-valued attribute has been tested in a branch, no sub-tree may test this attribute again (as the outcome of such a test is already known). However, a continuous attribute may still be tested by a branch even if a parent branch has already tested it, as a different cut point can be used to further partition the data. The coding scheme used for these trees is similar to that presented in [Wallace and Patrick 1993].

We transmit the topology in a depth-first fashion as a string of code words —

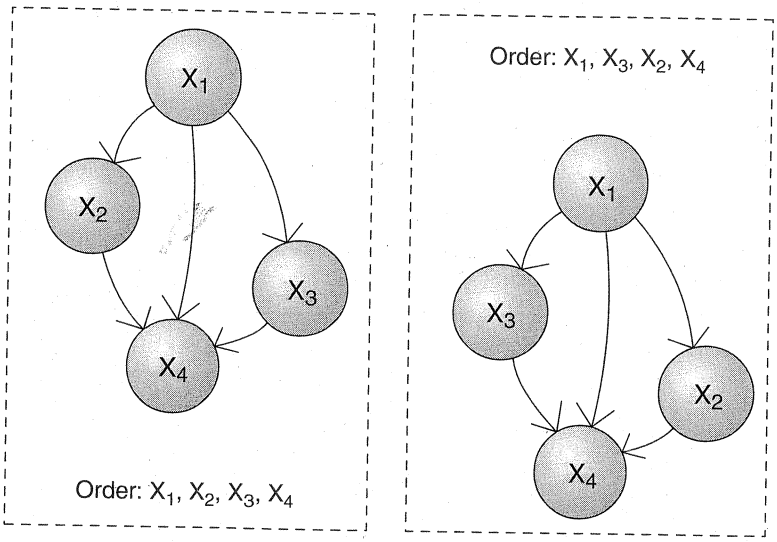

Figure 11.2 Two equivalent partially connected networks with different total node orderings.

each either 'branch' or 'leaf'. The probability of the root node being a branch is $n_A/(n_A+1)$ where n_A is the number of input attributes for the tree. The probability of any other node being a branch is taken to be $1/a$ where a is the 'arity' of the node's parent. The probability of a leaf is obviously one minus the probability of a branch. For a tree where all tests have a binary outcome, stating 'branch' or 'leaf' each cost one bit.[1] After each 'branch' code word, we state which of the input attributes is tested there. This costs $\log_2(n'_A)$ bits where n'_A is the number of input attributes that could be tested at that node. n'_A is equal to n_A at the root of the tree, but decreases by one in any path when a discrete attribute is tested (as further testing of the same discrete attribute is prohibited). If it is a continuous attribute we are testing, we also need to encode the associated cutpoint c. For this we use a scheme outlined in [Comley and Dowe 2003, sec. 3.1], and used prior to that in the software associated with [Wallace and Patrick 1993] and [Kornienko, Dowe, and Albrecht 2002, sec. 4.1].

Each 'leaf' code word is followed by the parameters for the model in that leaf — either μ and σ for a Gaussian distribution, or $\mathrm{P}(v_1), \ldots, \mathrm{P}(v_{m-1})$ for an m-state distribution (where the target attribute can take the values $v_1, \ldots, v_m$). Wallace and Boulton [1968] and Boulton and Wallace [1969] give well-behaved priors and coding schemes for both of the Gaussian and multistate models respectively.

This completes the transmission of H. We now transmit the data, D, one attribute at a time according to the node ordering of the network specified in H. For

1. Except at the root of the tree where $\mathrm{P}(branch) = n_A/(n_A+1)$

each attribute X_i, we can build an optimal code book based on the conditional probability distribution in the relevant node. We use this code book in conjunction with the attributes already sent to encode X_i. We thus achieve our two-part MML message.

If our H is a complicated network with high connectivity and large decision trees it will be expensive to transmit, but can achieve high compression of the training data, allowing us to state D very efficiently. At the other extreme oversimplified networks can be encoded cheaply, but may not fully exploit the correlations that exist in the data, making the transmission of D expensive. Minimising our two-part MML message corresponds to our intuitive wish to find a tradeoff between unjustifiably complicated models that overfit the data, and overly simplistic models that fail to recognize important patterns. The level of complexity we can accept in our models increases with the size of our (training) data.

11.4.8 Symmetric (Invertible) Languages

It is interesting to note that some families of conditional distribution are symmetric with respect to node ordering — that is, any probabilistic relationship $\mathrm{P}(X_i) = f(X_j, X_k)$ can also be expressed as $\mathrm{P}(X_j) = g(X_i, X_k)$, or $\mathrm{P}(X_k) = h(X_i, X_j)$, where f, g, and h are all in the family of conditional distributions. Put another way, the inverse of any model in the language is also in the language.

For Bayesian networks using such distributions, the node ordering has no effect on the family of joint distributions able to be expressed, provided that the connectivity of the network remains the same. In other words, reversing the direction of one or more arcs in a network will have no impact on the distributions it is able to represent. The choice of node ordering for such a network is somewhat arbitrary in the sense that it should not alter the joint distribution inferred.[2] This is in fact the case for the typical Bayesian network where all attributes are discrete and modeled by conditional probability tables. Another example of a model language able to be inverted without altering the joint distribution is that where all attributes are continuous and modeled as a linear combination of their parents, plus some Gaussian noise term. This is shown below:

$$X_i = a_1 P_1 + a_2 P_2 + \cdots + a_p P_p + N(\mu, \sigma^2),$$

where $P_1, \ldots P_p$ are the p parent attributes of X_i.

Although one can still do implicit learning with such languages, if the aim is simply to extract the conditional distribution, say $\mathrm{P}(X_i | X \setminus \{X_i\})$, from the inferred network, then one will do just as well to simply learn this conditional distribution outright rather than go to the trouble of inferring an entire Bayesian network. This idea is investigated by Ng and Jordan in [Ng and Jordan 2002], which is con-

2. In the case of causal networks the node ordering is often dictated by the user's notion of causality, or extra temporal information.

cerned with generative-discriminative model pairs. That work concerns two equivalent representations of a conditional probability distribution: one modeled explicitly (discriminative), and the other modeled implicitly via a joint distribution (generative). Ng and Jordan compare the performance of the generative and discriminative models, focusing on the efficiency of each and the asymptotic error rates. In this chapter we are interested in *asymmetric* languages — that is, situations where we are unable to express (or work with) the discriminative equivalent of a generative model. Thus the discriminative and generative models compared here do not really qualify as 'pairs' — the generative model is a more general case that can describe distributions unavailable to the discriminative model.

11.4.9 Inferring the Node Order

As mentioned in Section 11.4.8, some networks use conditional distribution languages that are symmetric with regard to node order. Altering the node order of such a network will not change the family of joint distributions able to be expressed.

If, however, we use asymmetric conditional models — for example, the class of decision trees described in Section 11.4.7 — then the order of the nodes can have a significant impact on the nature of the joint distribution.

Consider the simple case where we have only two attributes — a binary-valued attribute X_b and a continuous-valued attribute X_c. Using the decision tree language just mentioned, there are two ways to build a joint distribution over $(X_b \& X_c)$ — one using the ordering X_b, X_c and the other using the ordering X_c, X_b. These are illustrated in Figure 11.3. When we construct our MML message (using the coding scheme in Section 11.4.7), one of these networks will be cheaper than the other. So, in the case of such an asymmetric model language, MML provides us with a natural way of inferring node ordering. The node ordering in this example is not to be interpreted causally. We are simply choosing the ordering which provides us with the best family of joint distributions. For research pertaining to MML and causal networks, see, for example, [Wallace and Korb 1999].

11.4.10 An Efficient Search for Network Structure

Section 11.4.9 explained that, when using asymmetric conditional models, node ordering and connectivity can have a significant impact on the nature of the joint distribution. We show here how we can use this to our advantage when searching for the best network structure.

We begin by searching over the space of total node orderings. As mentioned in Section 11.4.7, there are $k!$ possible total orderings, where k is the number of attributes. Clearly we would like to avoid learning all the corresponding networks. First, we use the MML decision tree scheme discussed in Section 11.4.7 to build k decision tree models, $DT_1, \ldots, DT_k$, where DT_i models X_i and treats the other attributes as input. Note that just because X_j is an input attribute to DT_i does not necessarily mean that it is tested at any branches.

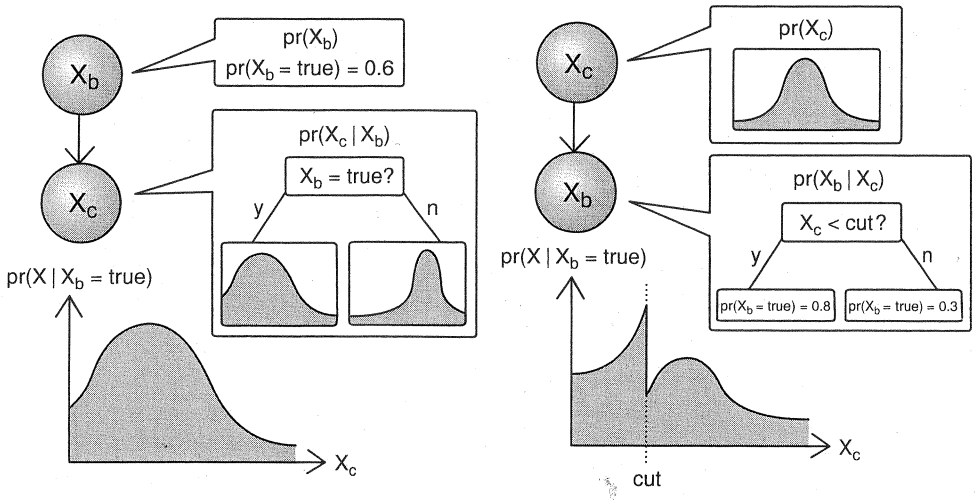

Figure 11.3 Two networks, each representing a different joint distribution over $X_b \& X_c$. This figure shows the difference that node order can make to the nature of the joint distribution when dealing with asymmetric Bayesian networks. Two networks are depicted - one on the left with the ordering (X_b, X_c), and one on the right with the ordering (X_c, X_b). To the right of each node we depict the conditional probability distributions it contains. Below each network is a (rough) graph showing how, when $X_b = \mathit{true}$, P(X) varies with X_c. NOTE: This figure is not drawn accurately or to scale — it is intended only to give an idea of the behavior of our class of asymmetric networks.

We can now establish a list of independencies, and one- and two-way dependencies. If DT_i does not test X_j, and DT_j does not test X_i then we can conclude that X_i and X_j are independent (at least in the presence of the other attributes) and there is not likely to be much benefit in directly connecting the corresponding nodes.

If DT_i *does* test X_j, but DT_j does *not* test X_i, then we establish a one-way dependency. This is particularly useful in formulating partial ordering constraints. Here we assert that there is little use in placing a connection from X_i to X_j in the network, as we are not able to express X_j's dependency on X_i. There is, however, some benefit in a connection directed from X_j to X_i, because we can see from examining DT_i that we *can* express some dependency of X_i on X_j. Given these considerations, it makes sense to try to place X_i after X_j in the total node ordering.

If DT_i tests X_j, and DT_j also tests X_i, then we conclude that there is a two-way dependency between X_i and X_j. This tells us that a connection between the corresponding nodes will probably be useful, but does not tell us which way this link should be directed, and hence does not shed any light on sensible total node orderings.

We now build a list L of useful directed links. For each one-way dependency from X_i to X_j, we add $X_i \rightarrow X_j$ to the list. For each two-way dependency between X_g and X_h, we add both $X_g \rightarrow X_h$ and $X_h \rightarrow X_g$ to the list.

Now we give each possible fully connected network structure a score equal to the

number of directed links in L that it exhibits. We keep only those structures with an equal highest score. For each of these structures, we remove any links that do not feature in L, creating a set of partially-connected structures, many of which will now be equivalent. For each group of equivalent structures we record the cardinality, and keep only one representative. We can now build a list of the conditional probability distributions required. Many of these will be used in more than one network, and there is no need to learn them more than once. For example, two networks may both model X_j as a probabilistic function of the same set of parent attributes, $P(X_j)$. The corresponding decision tree need only be learned once.

After learning all decision trees required (using the MML approach outlined in Section 11.4.7), we cost each network according to the scheme presented in Section 11.4.7. The cheapest network is chosen to represent our joint distribution.

While this method generally works well, it is not guaranteed to produce the optimal network structure. The two paragraphs below outline two potential downfalls.

Falsely Detecting Dependencies Consider an attribute X_a depending on a Boolean attribute X_b, and, *if* X_b *is true*, also depending on X_c. We conclude from this that X_a depends on both X_b and X_c, and that the corresponding directed links are worthwhile. Imagine now that we go with the ordering X_c, X_a, X_b. Suddenly the link $X_c \rightarrow X_a$ is useless — we cannot detect any dependency of X_a on X_c without the presence of X_b. It would be better to have removed this link, but it is too late because the structure (and connectivity) is decided before the trees are inferred, and it is only when we infer the trees that we discover $DT_{a|c}$ does not test X_c.

Failing to Detect a Dependency If some attributes are highly correlated, they may 'overshadow' each other. For example, X_a has a strong dependency on X_b, and a weaker (but still important) dependency on X_c. The decision tree DT_a tests X_b at the root node, nicely partitioning the classes. Each leaf now decides not to bother testing X_c, due to fragmentation of data, and the minimal extra purity gained. So we conclude that X_a does not depend on X_c, but in fact this independence is conditional on X_b being present. Imagine an ordering X_c, X_a, X_b where we would decide to remove the link $X_c \rightarrow X_a$. Now our encoding of X_a will not benefit from any correlations.

Another cause of this error is that in the presence of many input attributes, stating which attribute is to be tested at any branch becomes expensive. A 'borderline' branch may be rejected on this basis whereas in the actual network (where there are fewer input attributes) it will be cheaper to state that branch and it may be accepted.

11.4.11 A Coding Scheme for 'Supervised' Networks

We present in this subsection an alternative to the MML costing scheme given in Section 11.4.7. This alternative scheme can be used when we know, before inferring

the network, which attribute it is that we wish to predict. This is often the case in practical classification situations, where there is usually a particular attribute of interest which is difficult to measure, that we want to predict based on the rest of the (more easily) observed attributes. In this subsection we will refer to such an attribute as the 'target' attribute, and label it as X_t.

This scheme focuses on learning an accurate *conditional* distribution of X_t given $X \setminus \{X_t\}$, as opposed to learning an accurate joint distribution over all of X. Wettig, Grünwald, Roos, Myllymäki, and Tirri [2003] refer to networks that result from such schemes as 'supervised' networks, and to networks that have attempted instead to optimize the joint distribution as 'unsupervised' networks. We adopt this terminology, as it draws attention to the role of networks and their distributions in classification tasks.

If we had a universal language for our conditional probability distributions (CPDs), able to represent any conditional distribution at all, then we could do no better than to optimize the joint distribution over X. In other words, if one is able to perfectly model the joint distribution, then this will also yield (by taking the appropriate 'cross section') the best conditional distribution for any attribute. In practical situations, however, we cannot usually find such a perfect representation of the joint distribution, and the best joint distribution able to be expressed may not in fact correspond to the best conditional distribution for X_t.

For the asymmetric networks presented here, the structure, connectivity, and parameters required to represent the best joint distribution may differ significantly from those required to represent the best conditional distribution of X_t. We expect, when the task is to predict or classify X_t that the supervised network will produce better results.

Our proposed MML scheme for supervised networks differs only slightly from that for unsupervised networks presented in Section 11.4.7. The major difference is that the supervised scheme assumes that the values for $X \setminus \{X_t\}$ are common knowledge, and need not be included in the message. We transmit the network structure and the decision tree parameters in exactly the same manner. In the supervised scheme, though, we do not transmit the data values of $X \setminus \{X_t\}$. After decoding the network the receiver may use it, together with the values for $X \setminus \{X_t\}$, to derive a CPD $pr(X_t|X \setminus \{X_t\})$. It is by using this distribution that the values of our target attribute, X_t, are transmitted.

11.4.12 An Example Network

Figure 11.4 shows a network and one of the CPDs learned from the well-known iris data set.

Figure 11.5 summarizes the performance of various classifiers on the iris data set. The classifiers are: