distributions. **Hybrid Bayesian networks**, which include both discrete and continuous variables, use a variety of canonical distributions.

- Inference in Bayesian networks means computing the probability distribution of a set of query variables, given a set of evidence variables. Exact inference algorithms, such as **variable elimination**, evaluate sums of products of conditional probabilities as efficiently as possible.
- In **polytrees** (singly connected networks), exact inference takes time linear in the size of the network. In the general case, the problem is intractable.
- Stochastic approximation techniques such as **likelihood weighting** and **Markov chain Monte Carlo** can give reasonable estimates of the true posterior probabilities in a network and can cope with much larger networks than can exact algorithms.
- Probability theory can be combined with representational ideas from first-order logic to produce very powerful systems for reasoning under uncertainty. **Relational probability models** (RPMs) include representational restrictions that guarantee a well-defined probability distribution that can be expressed as an equivalent Bayesian network. **Open-universe probability models** handle **existence** and **identity uncertainty**, defining probabilty distributions over the infinite space of first-order possible worlds.
- Various alternative systems for reasoning under uncertainty have been suggested. Generally speaking, **truth-functional** systems are not well suited for such reasoning.

BIBLIOGRAPHICAL AND HISTORICAL NOTES

The use of networks to represent probabilistic information began early in the 20th century, with the work of Sewall Wright on the probabilistic analysis of genetic inheritance and animal growth factors (Wright, 1921, 1934). I. J. Good (1961), in collaboration with Alan Turing, developed probabilistic representations and Bayesian inference methods that could be regarded as a forerunner of modern Bayesian networks—although the paper is not often cited in this context.[10] The same paper is the original source for the noisy-OR model.

The **influence diagram** representation for decision problems, which incorporated a DAG representation for random variables, was used in decision analysis in the late 1970s (see Chapter 16), but only enumeration was used for evaluation. Judea Pearl developed the message-passing method for carrying out inference in tree networks (Pearl, 1982a) and polytree networks (Kim and Pearl, 1983) and explained the importance of causal rather than diagnostic probability models, in contrast to the certainty-factor systems then in vogue.

The first expert system using Bayesian networks was CONVINCE (Kim, 1983). Early applications in medicine included the MUNIN system for diagnosing neuromuscular disorders (Andersen *et al.*, 1989) and the PATHFINDER system for pathology (Heckerman, 1991). The CPCS system (Pradhan *et al.*, 1994) is a Bayesian network for internal medicine consisting

[10] I. J. Good was chief statistician for Turing's code-breaking team in World War II. In *2001: A Space Odyssey* (Clarke, 1968a), Good and Minsky are credited with making the breakthrough that led to the development of the HAL 9000 computer.

of 448 nodes, 906 links and 8,254 conditional probability values. (The front cover shows a portion of the network.)

Applications in engineering include the Electric Power Research Institute's work on monitoring power generators (Morjaria *et al.*, 1995), NASA's work on displaying time-critical information at Mission Control in Houston (Horvitz and Barry, 1995), and the general field of **network tomography**, which aims to infer unobserved local properties of nodes and links in the Internet from observations of end-to-end message performance (Castro *et al.*, 2004). Perhaps the most widely used Bayesian network systems have been the diagnosis-and-repair modules (e.g., the Printer Wizard) in Microsoft Windows (Breese and Heckerman, 1996) and the Office Assistant in Microsoft Office (Horvitz *et al.*, 1998). Another important application area is biology: Bayesian networks have been used for identifying human genes by reference to mouse genes (Zhang *et al.*, 2003), inferring cellular networks Friedman (2004), and many other tasks in bioinformatics. We could go on, but instead we'll refer you to Pourret *et al.* (2008), a 400-page guide to applications of Bayesian networks.

Ross Shachter (1986), working in the influence diagram community, developed the first complete algorithm for general Bayesian networks. His method was based on goal-directed reduction of the network using posterior-preserving transformations. Pearl (1986) developed a clustering algorithm for exact inference in general Bayesian networks, utilizing a conversion to a directed polytree of clusters in which message passing was used to achieve consistency over variables shared between clusters. A similar approach, developed by the statisticians David Spiegelhalter and Steffen Lauritzen (Lauritzen and Spiegelhalter, 1988), is based on conversion to an undirected form of graphical model called a **Markov network**. This approach is implemented in the HUGIN system, an efficient and widely used tool for uncertain reasoning (Andersen *et al.*, 1989). Boutilier *et al.* (1996) show how to exploit context-specific independence in clustering algorithms.

MARKOV NETWORK

The basic idea of variable elimination—that repeated computations within the overall sum-of-products expression can be avoided by caching—appeared in the symbolic probabilistic inference (SPI) algorithm (Shachter *et al.*, 1990). The elimination algorithm we describe is closest to that developed by Zhang and Poole (1994). Criteria for pruning irrelevant variables were developed by Geiger *et al.* (1990) and by Lauritzen *et al.* (1990); the criterion we give is a simple special case of these. Dechter (1999) shows how the variable elimination idea is essentially identical to **nonserial dynamic programming** (Bertele and Brioschi, 1972), an algorithmic approach that can be applied to solve a range of inference problems in Bayesian networks—for example, finding the **most likely explanation** for a set of observations. This connects Bayesian network algorithms to related methods for solving CSPs and gives a direct measure of the complexity of exact inference in terms of the tree width of the network. Wexler and Meek (2009) describe a method of preventing exponential growth in the size of factors computed in variable elimination; their algorithm breaks down large factors into products of smaller factors and simultaneously computes an error bound for the resulting approximation.

NONSERIAL DYNAMIC PROGRAMMING

The inclusion of continuous random variables in Bayesian networks was considered by Pearl (1988) and Shachter and Kenley (1989); these papers discussed networks containing only continuous variables with linear Gaussian distributions. The inclusion of discrete variables has been investigated by Lauritzen and Wermuth (1989) and implemented in the

cHUGIN system (Olesen, 1993). Further analysis of linear Gaussian models, with connections to many other models used in statistics, appears in Roweis and Ghahramani (1999) The probit distribution is usually attributed to Gaddum (1933) and Bliss (1934), although it had been discovered several times in the 19th century. Bliss's work was expanded considerably by Finney (1947). The probit has been used widely for modeling discrete choice phenomena and can be extended to handle more than two choices (Daganzo, 1979). The logit model was introduced by Berkson (1944); initially much derided, it eventually became more popular than the probit model. Bishop (1995) gives a simple justification for its use.

Cooper (1990) showed that the general problem of inference in unconstrained Bayesian networks is NP-hard, and Paul Dagum and Mike Luby (1993) showed the corresponding approximation problem to be NP-hard. Space complexity is also a serious problem in both clustering and variable elimination methods. The method of **cutset conditioning**, which was developed for CSPs in Chapter 6, avoids the construction of exponentially large tables. In a Bayesian network, a cutset is a set of nodes that, when instantiated, reduces the remaining nodes to a polytree that can be solved in linear time and space. The query is answered by summing over all the instantiations of the cutset, so the overall space requirement is still linear (Pearl, 1988). Darwiche (2001) describes a recursive conditioning algorithm that allows a complete range of space/time tradeoffs.

The development of fast approximation algorithms for Bayesian network inference is a very active area, with contributions from statistics, computer science, and physics. The rejection sampling method is a general technique that is long known to statisticians; it was first applied to Bayesian networks by Max Henrion (1988), who called it **logic sampling**. Likelihood weighting, which was developed by Fung and Chang (1989) and Shachter and Peot (1989), is an example of the well-known statistical method of **importance sampling**. Cheng and Druzdzel (2000) describe an adaptive version of likelihood weighting that works well even when the evidence has very low prior likelihood.

Markov chain Monte Carlo (MCMC) algorithms began with the Metropolis algorithm, due to Metropolis *et al.* (1953), which was also the source of the simulated annealing algorithm described in Chapter 4. The Gibbs sampler was devised by Geman and Geman (1984) for inference in undirected Markov networks. The application of MCMC to Bayesian networks is due to Pearl (1987). The papers collected by Gilks *et al.* (1996) cover a wide variety of applications of MCMC, several of which were developed in the well-known BUGS package (Gilks *et al.*, 1994).

VARIATIONAL APPROXIMATION

There are two very important families of approximation methods that we did not cover in the chapter. The first is the family of **variational approximation** methods, which can be used to simplify complex calculations of all kinds. The basic idea is to propose a reduced version of the original problem that is simple to work with, but that resembles the original problem as closely as possible. The reduced problem is described by some **variational parameters** $\boldsymbol{\lambda}$ that are adjusted to minimize a distance function D between the original and the reduced problem, often by solving the system of equations $\partial D/\partial\boldsymbol{\lambda} = 0$. In many cases, strict upper and lower bounds can be obtained. Variational methods have long been used in statistics (Rustagi, 1976). In statistical physics, the **mean-field** method is a particular variational approximation in which the individual variables making up the model are assumed

VARIATIONAL PARAMETER

MEAN FIELD

to be completely independent. This idea was applied to solve large undirected Markov networks (Peterson and Anderson, 1987; Parisi, 1988). Saul *et al.* (1996) developed the mathematical foundations for applying variational methods to Bayesian networks and obtained accurate lower-bound approximations for sigmoid networks with the use of mean-field methods. Jaakkola and Jordan (1996) extended the methodology to obtain both lower and upper bounds. Since these early papers, variational methods have been applied to many specific families of models. The remarkable paper by Wainwright and Jordan (2008) provides a unifying theoretical analysis of the literature on variational methods.

A second important family of approximation algorithms is based on Pearl's polytree message-passing algorithm (1982a). This algorithm can be applied to general networks, as suggested by Pearl (1988). The results might be incorrect, or the algorithm might fail to terminate, but in many cases, the values obtained are close to the true values. Little attention was paid to this so-called **belief propagation** (or BP) approach until McEliece *et al.* (1998) observed that message passing in a multiply connected Bayesian network was exactly the computation performed by the **turbo decoding** algorithm (Berrou *et al.*, 1993), which provided a major breakthrough in the design of efficient error-correcting codes. The implication is that BP is both fast and accurate on the very large and very highly connected networks used for decoding and might therefore be useful more generally. Murphy *et al.* (1999) presented a promising empirical study of BP's performance, and Weiss and Freeman (2001) established strong convergence results for BP on linear Gaussian networks. Weiss (2000b) shows how an approximation called loopy belief propagation works, and when the approximation is correct. Yedidia *et al.* (2005) made further connections between loopy propagation and ideas from statistical physics.

BELIEF PROPAGATION

TURBO DECODING

The connection between probability and first-order languages was first studied by Carnap (1950). Gaifman (1964) and Scott and Krauss (1966) defined a language in which probabilities could be associated with first-order sentences and for which models were probability measures on possible worlds. Within AI, this idea was developed for propositional logic by Nilsson (1986) and for first-order logic by Halpern (1990). The first extensive investigation of knowledge representation issues in such languages was carried out by Bacchus (1990). The basic idea is that each sentence in the knowledge base expressed a *constraint* on the distribution over possible worlds; one sentence entails another if it expresses a stronger constraint. For example, the sentence $\forall x \quad P(Hungry(x)) > 0.2$ rules out distributions in which any object is hungry with probability less than 0.2; thus, it entails the sentence $\forall x \quad P(Hungry(x)) > 0.1$. It turns out that writing a *consistent* set of sentences in these languages is quite difficult and constructing a unique probability model nearly impossible unless one adopts the representation approach of Bayesian networks by writing suitable sentences about conditional probabilities.

Beginning in the early 1990s, researchers working on complex applications noticed the expressive limitations of Bayesian networks and developed various languages for writing "templates" with logical variables, from which large networks could be constructed automatically for each problem instance (Breese, 1992; Wellman *et al.*, 1992). The most important such language was BUGS (Bayesian inference Using Gibbs Sampling) (Gilks *et al.*, 1994), which combined Bayesian networks with the **indexed random variable** notation common in

INDEXED RANDOM VARIABLE

statistics. (In BUGS, an indexed random variable looks like $X[i]$, where i has a defined integer range.) These languages inherited the key property of Bayesian networks: every well-formed knowledge base defines a unique, consistent probability model. Languages with well-defined semantics based on unique names and domain closure drew on the representational capabilities of logic programming (Poole, 1993; Sato and Kameya, 1997; Kersting *et al.*, 2000) and semantic networks (Koller and Pfeffer, 1998; Pfeffer, 2000). Pfeffer (2007) went on to develop IBAL, which represents first-order probability models as probabilistic programs in a programming language extended with a randomization primitive. Another important thread was the combination of relational and first-order notations with (undirected) Markov networks (Taskar *et al.*, 2002; Domingos and Richardson, 2004), where the emphasis has been less on knowledge representation and more on learning from large data sets.

Initially, inference in these models was performed by generating an equivalent Bayesian network. Pfeffer *et al.* (1999) introduced a variable elimination algorithm that cached each computed factor for reuse by later computations involving the same relations but different objects, thereby realizing some of the computational gains of lifting. The first truly lifted inference algorithm was a lifted form of variable elimination described by Poole (2003) and subsequently improved by de Salvo Braz *et al.* (2007). Further advances, including cases where certain aggregate probabilities can be computed in closed form, are described by Milch *et al.* (2008) and Kisynski and Poole (2009). Pasula and Russell (2001) studied the application of MCMC to avoid building the complete equivalent Bayes net in cases of relational and identity uncertainty. Getoor and Taskar (2007) collect many important papers on first-order probability models and their use in machine learning.

RECORD LINKAGE

Probabilistic reasoning about identity uncertainty has two distinct origins. In statistics, the problem of **record linkage** arises when data records do not contain standard unique identifiers—for example, various citations of this book might name its first author "Stuart Russell" or "S. J. Russell" or even "Stewart Russle," and other authors may use the some of the same names. Literally hundreds of companies exist solely to solve record linkage problems in financial, medical, census, and other data. Probabilistic analysis goes back to work by Dunn (1946); the Fellegi–Sunter model (1969), which is essentially naive Bayes applied to matching, still dominates current practice. The second origin for work on identity uncertainty is multitarget tracking (Sittler, 1964), which we cover in Chapter 15. For most of its history, work in symbolic AI assumed erroneously that sensors could supply sentences with unique identifiers for objects. The issue was studied in the context of language understanding by Charniak and Goldman (1992) and in the context of surveillance by (Huang and Russell, 1998) and Pasula *et al.* (1999). Pasula *et al.* (2003) developed a complex generative model for authors, papers, and citation strings, involving both relational and identity uncertainty, and demonstrated high accuracy for citation information extraction. The first formally defined language for open-universe probability models was BLOG (Milch *et al.*, 2005), which came with a complete (albeit slow) MCMC inference algorithm for all well-defined mdoels. (The program code faintly visible on the front cover of this book is part of a BLOG model for detecting nuclear explosions from seismic signals as part of the UN Comprehensive Test Ban Treaty verification regime.) Laskey (2008) describes another open-universe modeling language called **multi-entity Bayesian networks**.

As explained in Chapter 13, early probabilistic systems fell out of favor in the early 1970s, leaving a partial vacuum to be filled by alternative methods. Certainty factors were invented for use in the medical expert system MYCIN (Shortliffe, 1976), which was intended both as an engineering solution and as a model of human judgment under uncertainty. The collection *Rule-Based Expert Systems* (Buchanan and Shortliffe, 1984) provides a complete overview of MYCIN and its descendants (see also Stefik, 1995). David Heckerman (1986) showed that a slightly modified version of certainty factor calculations gives correct probabilistic results in some cases, but results in serious overcounting of evidence in other cases. The PROSPECTOR expert system (Duda *et al.*, 1979) used a rule-based approach in which the rules were justified by a (seldom tenable) global independence assumption.

Dempster–Shafer theory originates with a paper by Arthur Dempster (1968) proposing a generalization of probability to interval values and a combination rule for using them. Later work by Glenn Shafer (1976) led to the Dempster-Shafer theory's being viewed as a competing approach to probability. Pearl (1988) and Ruspini *et al.* (1992) analyze the relationship between the Dempster–Shafer theory and standard probability theory.

Fuzzy sets were developed by Lotfi Zadeh (1965) in response to the perceived difficulty of providing exact inputs to intelligent systems. The text by Zimmermann (2001) provides a thorough introduction to fuzzy set theory; papers on fuzzy applications are collected in Zimmermann (1999). As we mentioned in the text, fuzzy logic has often been perceived incorrectly as a direct competitor to probability theory, whereas in fact it addresses a different set of issues. **Possibility theory** (Zadeh, 1978) was introduced to handle uncertainty in fuzzy systems and has much in common with probability. Dubois and Prade (1994) survey the connections between possibility theory and probability theory.

POSSIBILITY THEORY

The resurgence of probability depended mainly on Pearl's development of Bayesian networks as a method for representing and using conditional independence information. This resurgence did not come without a fight; Peter Cheeseman's (1985) pugnacious "In Defense of Probability" and his later article "An Inquiry into Computer Understanding" (Cheeseman, 1988, with commentaries) give something of the flavor of the debate. Eugene Charniak helped present the ideas to AI researchers with a popular article, "Bayesian networks without tears"[11] (1991), and book (1993). The book by Dean and Wellman (1991) also helped introduce Bayesian networks to AI researchers. One of the principal philosophical objections of the logicists was that the numerical calculations that probability theory was thought to require were not apparent to introspection and presumed an unrealistic level of precision in our uncertain knowledge. The development of **qualitative probabilistic networks** (Wellman, 1990a) provided a purely qualitative abstraction of Bayesian networks, using the notion of positive and negative influences between variables. Wellman shows that in many cases such information is sufficient for optimal decision making without the need for the precise specification of probability values. Goldszmidt and Pearl (1996) take a similar approach. Work by Adnan Darwiche and Matt Ginsberg (1992) extracts the basic properties of conditioning and evidence combination from probability theory and shows that they can also be applied in logical and default reasoning. Often, programs speak louder than words, and the ready avail-

[11] The title of the original version of the article was "Pearl for swine."

ability of high-quality software such as the Bayes Net toolkit (Murphy, 2001) accelerated the adoption of the technology.

The most important single publication in the growth of Bayesian networks was undoubtedly the text *Probabilistic Reasoning in Intelligent Systems* (Pearl, 1988). Several excellent texts (Lauritzen, 1996; Jensen, 2001; Korb and Nicholson, 2003; Jensen, 2007; Darwiche, 2009; Koller and Friedman, 2009) provide thorough treatments of the topics we have covered in this chapter. New research on probabilistic reasoning appears both in mainstream AI journals, such as *Artificial Intelligence* and the *Journal of AI Research*, and in more specialized journals, such as the *International Journal of Approximate Reasoning*. Many papers on graphical models, which include Bayesian networks, appear in statistical journals. The proceedings of the conferences on Uncertainty in Artificial Intelligence (UAI), Neural Information Processing Systems (NIPS), and Artificial Intelligence and Statistics (AISTATS) are excellent sources for current research.

EXERCISES

14.1 We have a bag of three biased coins a, b, and c with probabilities of coming up heads of 30%, 60%, and 75%, respectively. One coin is drawn randomly from the bag (with equal likelihood of drawing each of the three coins), and then the coin is flipped three times to generate the outcomes X_1, X_2, and X_3.

a. Draw the Bayesian network corresponding to this setup and define the necessary CPTs.

b. Calculate which coin was most likely to have been drawn from the bag if the observed flips come out heads twice and tails once.

14.2 Equation (14.1) on page 513 defines the joint distribution represented by a Bayesian network in terms of the parameters $\theta(X_i \mid Parents(X_i))$. This exercise asks you to derive the equivalence between the parameters and the conditional probabilities $\mathbf{P}(X_i \mid Parents(X_i))$ from this definition.

a. Consider a simple network $X \rightarrow Y \rightarrow Z$ with three Boolean variables. Use Equations (13.3) and (13.6) (pages 485 and 492) to express the conditional probability $P(z \mid y)$ as the ratio of two sums, each over entries in the joint distribution $\mathbf{P}(X, Y, Z)$.

b. Now use Equation (14.1) to write this expression in terms of the network parameters $\theta(X)$, $\theta(Y \mid X)$, and $\theta(Z \mid Y)$.

c. Next, expand out the summations in your expression from part (b), writing out explicitly the terms for the true and false values of each summed variable. Assuming that all network parameters satisfy the constraint $\sum_{x_i} \theta(x_i \mid parents(X_i)) = 1$, show that the resulting expression reduces to $\theta(x \mid y)$.

d. Generalize this derivation to show that $\theta(X_i \mid Parents(X_i)) = \mathbf{P}(X_i \mid Parents(X_i))$ for any Bayesian network.

ARC REVERSAL

14.3 The operation of **arc reversal** in a Bayesian network allows us to change the direction of an arc $X \rightarrow Y$ while preserving the joint probability distribution that the network represents (Shachter, 1986). Arc reversal may require introducing new arcs: all the parents of X also become parents of Y, and all parents of Y also become parents of X.

a. Assume that X and Y start with m and n parents, respectively, and that all variables have k values. By calculating the change in size for the CPTs of X and Y, show that the total number of parameters in the network cannot decrease during arc reversal. (*Hint*: the parents of X and Y need not be disjoint.)

b. Under what circumstances can the total number remain constant?

c. Let the parents of X be $\mathbf{U} \cup \mathbf{V}$ and the parents of Y be $\mathbf{V} \cup \mathbf{W}$, where $\mathbf{U}$ and $\mathbf{W}$ are disjoint. The formulas for the new CPTs after arc reversal are as follows:

$$\mathbf{P}(Y \mid \mathbf{U}, \mathbf{V}, \mathbf{W}) = \sum_x \mathbf{P}(Y \mid \mathbf{V}, \mathbf{W}, x)\mathbf{P}(x \mid \mathbf{U}, \mathbf{V})$$

$$\mathbf{P}(X \mid \mathbf{U}, \mathbf{V}, \mathbf{W}, Y) = \mathbf{P}(Y \mid X, \mathbf{V}, \mathbf{W})\mathbf{P}(X \mid \mathbf{U}, \mathbf{V}) / \mathbf{P}(Y \mid \mathbf{U}, \mathbf{V}, \mathbf{W}) \, .$$

Prove that the new network expresses the same joint distribution over all variables as the original network.

14.4 Consider the Bayesian network in Figure 14.2.

a. If no evidence is observed, are $Burglary$ and $Earthquake$ independent? Prove this from the numerical semantics and from the topological semantics.

b. If we observe $Alarm = true$, are $Burglary$ and $Earthquake$ independent? Justify your answer by calculating whether the probabilities involved satisfy the definition of conditional independence.

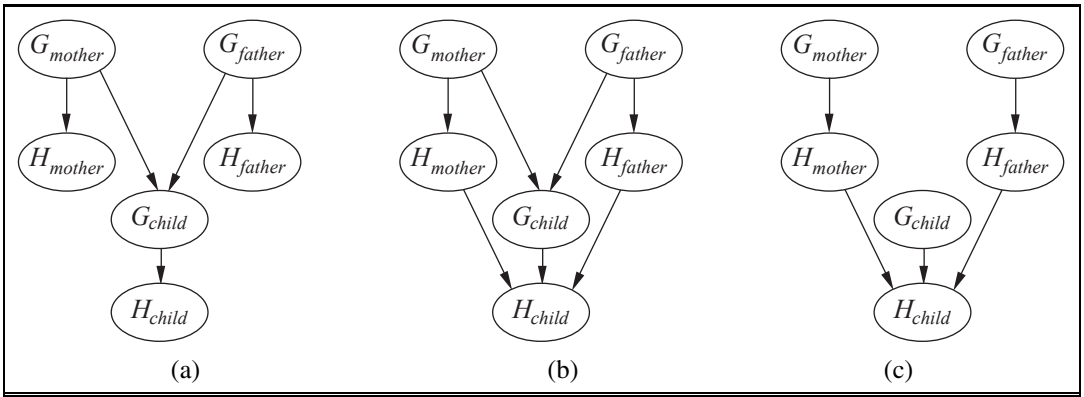

Figure 14.20 Three possible structures for a Bayesian network describing genetic inheritance of handedness.

14.5 Let H_x be a random variable denoting the handedness of an individual x, with possible values l or r. A common hypothesis is that left- or right-handedness is inherited by a simple mechanism; that is, perhaps there is a gene G_x, also with values l or r, and perhaps actual

handedness turns out mostly the same (with some probability s) as the gene an individual possesses. Furthermore, perhaps the gene itself is equally likely to be inherited from either of an individual's parents, with a small nonzero probability m of a random mutation flipping the handedness.

a. Which of the three networks in Figure 14.20 claim that $\mathbf{P}(G_{father}, G_{mother}, G_{child}) = \mathbf{P}(G_{father})\mathbf{P}(G_{mother})\mathbf{P}(G_{child})$?

b. Which of the three networks make independence claims that are consistent with the hypothesis about the inheritance of handedness?

c. Which of the three networks is the best description of the hypothesis?

d. Write down the CPT for the G_{child} node in network (a), in terms of s and m.

e. Suppose that $P(G_{father} = l) = P(G_{mother} = l) = q$. In network (a), derive an expression for $P(G_{child} = l)$ in terms of m and q only, by conditioning on its parent nodes.

f. Under conditions of genetic equilibrium, we expect the distribution of genes to be the same across generations. Use this to calculate the value of q, and, given what you know about handedness in humans, explain why the hypothesis described at the beginning of this question must be wrong.

14.6 The **Markov blanket** of a variable is defined on page 517. Prove that a variable is independent of all other variables in the network, given its Markov blanket and derive Equation (14.12) (page 538).

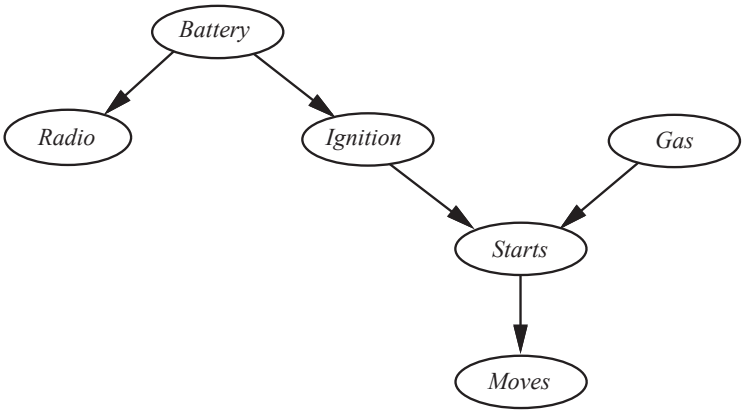

Figure 14.21 A Bayesian network describing some features of a car's electrical system and engine. Each variable is Boolean, and the *true* value indicates that the corresponding aspect of the vehicle is in working order.

14.7 Consider the network for car diagnosis shown in Figure 14.21.

a. Extend the network with the Boolean variables $IcyWeather$ and $StarterMotor$.

b. Give reasonable conditional probability tables for all the nodes.

c. How many independent values are contained in the joint probability distribution for eight Boolean nodes, assuming that no conditional independence relations are known to hold among them?

d. How many independent probability values do your network tables contain?

e. The conditional distribution for $Starts$ could be described as a **noisy-AND** distribution. Define this family in general and relate it to the noisy-OR distribution.

14.8 Consider a simple Bayesian network with root variables $Cold$, Flu, and $Malaria$ and child variable $Fever$, with a noisy-OR conditional distribution for $Fever$ as described in Section 14.3. By adding appropriate auxiliary variables for inhibition events and fever-inducing events, construct an equivalent Bayesian network whose CPTs (except for root variables) are deterministic. Define the CPTs and prove equivalence.

14.9 Consider the family of linear Gaussian networks, as defined on page 520.

a. In a two-variable network, let X_1 be the parent of X_2, let X_1 have a Gaussian prior, and let $\mathbf{P}(X_2 \mid X_1)$ be a linear Gaussian distribution. Show that the joint distribution $P(X_1, X_2)$ is a multivariate Gaussian, and calculate its covariance matrix.

b. Prove by induction that the joint distribution for a general linear Gaussian network on $X_1, \ldots, X_n$ is also a multivariate Gaussian.

14.10 The probit distribution defined on page 522 describes the probability distribution for a Boolean child, given a single continuous parent.

a. How might the definition be extended to cover multiple continuous parents?

b. How might it be extended to handle a *multivalued* child variable? Consider both cases where the child's values are ordered (as in selecting a gear while driving, depending on speed, slope, desired acceleration, etc.) and cases where they are unordered (as in selecting bus, train, or car to get to work). (*Hint*: Consider ways to divide the possible values into two sets, to mimic a Boolean variable.)

14.11 In your local nuclear power station, there is an alarm that senses when a temperature gauge exceeds a given threshold. The gauge measures the temperature of the core. Consider the Boolean variables A (alarm sounds), F_A (alarm is faulty), and F_G (gauge is faulty) and the multivalued nodes G (gauge reading) and T (actual core temperature).

a. Draw a Bayesian network for this domain, given that the gauge is more likely to fail when the core temperature gets too high.

b. Is your network a polytree? Why or why not?

c. Suppose there are just two possible actual and measured temperatures, normal and high; the probability that the gauge gives the correct temperature is x when it is working, but y when it is faulty. Give the conditional probability table associated with G.

d. Suppose the alarm works correctly unless it is faulty, in which case it never sounds. Give the conditional probability table associated with A.

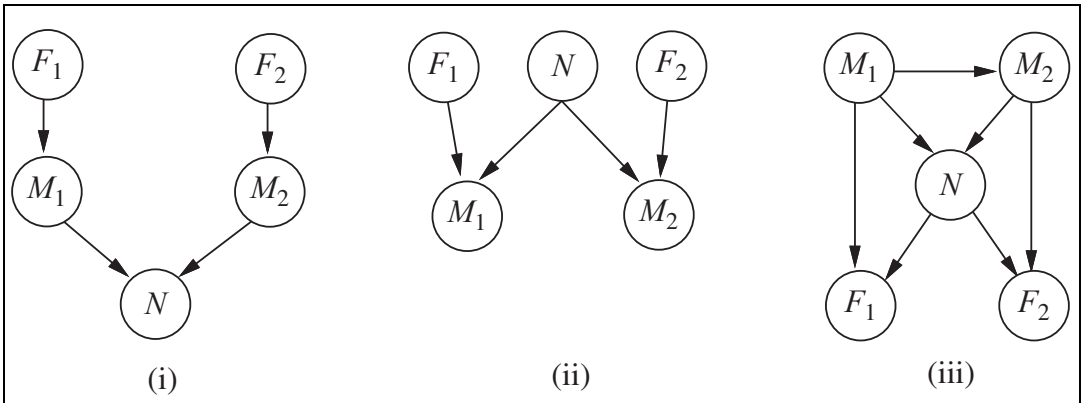

Figure 14.22 Three possible networks for the telescope problem.

e. Suppose the alarm and gauge are working and the alarm sounds. Calculate an expression for the probability that the temperature of the core is too high, in terms of the various conditional probabilities in the network.

14.12 Two astronomers in different parts of the world make measurements M_1 and M_2 of the number of stars N in some small region of the sky, using their telescopes. Normally, there is a small possibility e of error by up to one star in each direction. Each telescope can also (with a much smaller probability f) be badly out of focus (events F_1 and F_2), in which case the scientist will undercount by three or more stars (or if N is less than 3, fail to detect any stars at all). Consider the three networks shown in Figure 14.22.

a. Which of these Bayesian networks are correct (but not necessarily efficient) representations of the preceding information?

b. Which is the best network? Explain.

c. Write out a conditional distribution for $\mathbf{P}(M_1 \mid N)$, for the case where $N \in \{1, 2, 3\}$ and $M_1 \in \{0, 1, 2, 3, 4\}$. Each entry in the conditional distribution should be expressed as a function of the parameters e and/or f.

d. Suppose $M_1 = 1$ and $M_2 = 3$. What are the *possible* numbers of stars if you assume no prior constraint on the values of N?

e. What is the *most likely* number of stars, given these observations? Explain how to compute this, or if it is not possible to compute, explain what additional information is needed and how it would affect the result.

14.13 Consider the Bayes net shown in Figure 14.23.

a. Which, if any, of the following are asserted by the network *structure* (ignoring the CPTs for now)?

(i) $\mathbf{P}(B, I, M) = \mathbf{P}(B)\mathbf{P}(I)\mathbf{P}(M)$.

(ii) $\mathbf{P}(J \mid G) = \mathbf{P}(J \mid G, I)$.

(iii) $\mathbf{P}(M \mid G, B, I) = \mathbf{P}(M \mid G, B, I, J)$.

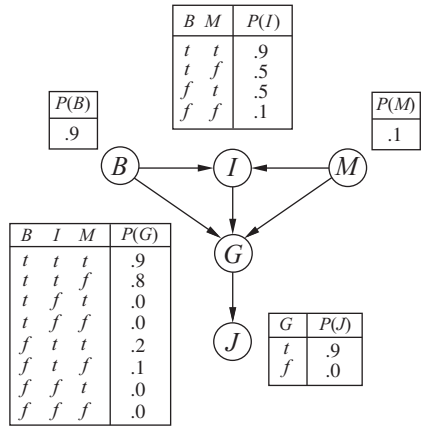

Figure 14.23 A simple Bayes net with Boolean variables $B = BrokeElectionLaw$, $I = Indicted$, $M = PoliticallyMotivatedProsecutor$, $G = FoundGuilty$, $J = Jailed$.

b. Calculate the value of $P(b, i, m, \neg g, j)$.

c. Calculate the probability that someone goes to jail given that they broke the law, have been indicted, and face a politically motivated prosecutor.

d. A **context-specific independence** (see page 542) allows a variable to be independent of some of its parents given certain values of others. In addition to the usual conditional independences given by the graph structure, what context-specific independences exist in the Bayes net in Figure 14.23?

e. Suppose we want to add the variable $P = PresidentialPardon$ to the network; draw the new network and briefly explain any links you add.

14.14 Consider the variable elimination algorithm in Figure 14.11 (page 528).

a. Section 14.4 applies variable elimination to the query

$$\mathbf{P}(Burglary \mid JohnCalls = true, MaryCalls = true) \ .$$

Perform the calculations indicated and check that the answer is correct.

b. Count the number of arithmetic operations performed, and compare it with the number performed by the enumeration algorithm.

c. Suppose a network has the form of a *chain*: a sequence of Boolean variables $X_1, \ldots, X_n$ where $Parents(X_i) = \{X_{i-1}\}$ for $i = 2, \ldots, n$. What is the complexity of computing $\mathbf{P}(X_1 \mid X_n = true)$ using enumeration? Using variable elimination?

d. Prove that the complexity of running variable elimination on a polytree network is linear in the size of the tree for any variable ordering consistent with the network structure.

14.15 Investigate the complexity of exact inference in general Bayesian networks:

a. Prove that any 3-SAT problem can be reduced to exact inference in a Bayesian network constructed to represent the particular problem and hence that exact inference is NP-

hard. (*Hint*: Consider a network with one variable for each proposition symbol, one for each clause, and one for the conjunction of clauses.)

b. The problem of counting the number of satisfying assignments for a 3-SAT problem is #P-complete. Show that exact inference is at least as hard as this.

14.16 Consider the problem of generating a random sample from a specified distribution on a single variable. Assume you have a random number generator that returns a random number uniformly distributed between 0 and 1.

CUMULATIVE DISTRIBUTION

a. Let X be a discrete variable with $P(X = x_i) = p_i$ for $i \in \{1, \ldots, k\}$. The **cumulative distribution** of X gives the probability that $X \in \{x_1, \ldots, x_j\}$ for each possible j. (See also Appendix A.) Explain how to calculate the cumulative distribution in $O(k)$ time and how to generate a single sample of X from it. Can the latter be done in less than $O(k)$ time?

b. Now suppose we want to generate N samples of X, where $N \gg k$. Explain how to do this with an expected run time per sample that is *constant* (i.e., independent of k).

c. Now consider a continuous-valued variable with a parameterized distribution (e.g., Gaussian). How can samples be generated from such a distribution?

d. Suppose you want to query a continuous-valued variable and you are using a sampling algorithm such as LIKELIHOODWEIGHTING to do the inference. How would you have to modify the query-answering process?

14.17 Consider the query $\mathbf{P}(Rain \mid Sprinkler = true, WetGrass = true)$ in Figure 14.12(a) (page 529) and how Gibbs sampling can answer it.

a. How many states does the Markov chain have?

b. Calculate the **transition matrix Q** containing $q(\mathbf{y} \rightarrow \mathbf{y}')$ for all $\mathbf{y}$, $\mathbf{y}'$.

c. What does $\mathbf{Q}^2$, the square of the transition matrix, represent?

d. What about $\mathbf{Q}^n$ as $n \rightarrow \infty$?

e. Explain how to do probabilistic inference in Bayesian networks, assuming that $\mathbf{Q}^n$ is available. Is this a practical way to do inference?

14.18 This exercise explores the stationary distribution for Gibbs sampling methods.

a. The convex composition $[\alpha, q_1; 1 - \alpha, q_2]$ of q_1 and q_2 is a transition probability distribution that first chooses one of q_1 and q_2 with probabilities α and $1 - \alpha$, respectively, and then applies whichever is chosen. Prove that if q_1 and q_2 are in detailed balance with π, then their convex composition is also in detailed balance with π. (*Note*: this result justifies a variant of GIBBS-ASK in which variables are chosen at random rather than sampled in a fixed sequence.)

b. Prove that if each of q_1 and q_2 has π as its stationary distribution, then the sequential composition $q = q_1 \circ q_2$ also has π as its stationary distribution.

METROPOLIS–HASTINGS

14.19 The **Metropolis–Hastings** algorithm is a member of the MCMC family; as such, it is designed to generate samples $\mathbf{x}$ (eventually) according to target probabilities $\pi(\mathbf{x})$. (Typically

PROPOSAL DISTRIBUTION

ACCEPTANCE PROBABILITY

we are interested in sampling from $\pi(\mathbf{x}) = P(\mathbf{x} \mid \mathbf{e})$.) Like simulated annealing, Metropolis–Hastings operates in two stages. First, it samples a new state $\mathbf{x}'$ from a **proposal distribution** $q(\mathbf{x}' \mid \mathbf{x})$, given the current state $\mathbf{x}$. Then, it probabilistically accepts or rejects $\mathbf{x}'$ according to the **acceptance probability**

$$\alpha(\mathbf{x}' \mid \mathbf{x}) = \min\left(1, \frac{\pi(\mathbf{x}')q(\mathbf{x} \mid \mathbf{x}')}{\pi(\mathbf{x})q(\mathbf{x}' \mid \mathbf{x})}\right) .$$

If the proposal is rejected, the state remains at $\mathbf{x}$.

a. Consider an ordinary Gibbs sampling step for a specific variable X_i. Show that this step, considered as a proposal, is guaranteed to be accepted by Metropolis–Hastings. (Hence, Gibbs sampling is a special case of Metropolis–Hastings.)

b. Show that the two-step process above, viewed as a transition probability distribution, is in detailed balance with π.

14.20 Three soccer teams A, B, and C, play each other once. Each match is between two teams, and can be won, drawn, or lost. Each team has a fixed, unknown degree of quality—an integer ranging from 0 to 3—and the outcome of a match depends probabilistically on the difference in quality between the two teams.

a. Construct a relational probability model to describe this domain, and suggest numerical values for all the necessary probability distributions.

b. Construct the equivalent Bayesian network for the three matches.

c. Suppose that in the first two matches A beats B and draws with C. Using an exact inference algorithm of your choice, compute the posterior distribution for the outcome of the third match.

d. Suppose there are n teams in the league and we have the results for all but the last match. How does the complexity of predicting the last game vary with n?

e. Investigate the application of MCMC to this problem. How quickly does it converge in practice and how well does it scale?

15 PROBABILISTIC REASONING OVER TIME

In which we try to interpret the present, understand the past, and perhaps predict the future, even when very little is crystal clear.

Agents in partially observable environments must be able to keep track of the current state, to the extent that their sensors allow. In Section 4.4 we showed a methodology for doing that: an agent maintains a **belief state** that represents which states of the world are currently possible. From the belief state and a **transition model**, the agent can predict how the world might evolve in the next time step. From the percepts observed and a **sensor model**, the agent can update the belief state. This is a pervasive idea: in Chapter 4 belief states were represented by explicitly enumerated sets of states, whereas in Chapters 7 and 11 they were represented by logical formulas. Those approaches defined belief states in terms of which world states were *possible*, but could say nothing about which states were *likely* or *unlikely*. In this chapter, we use probability theory to quantify the degree of belief in elements of the belief state.

As we show in Section 15.1, time itself is handled in the same way as in Chapter 7: a changing world is modeled using a variable for each aspect of the world state *at each point in time*. The transition and sensor models may be uncertain: the transition model describes the probability distribution of the variables at time t, given the state of the world at past times, while the sensor model describes the probability of each percept at time t, given the current state of the world. Section 15.2 defines the basic inference tasks and describes the general structure of inference algorithms for temporal models. Then we describe three specific kinds of models: **hidden Markov models**, **Kalman filters**, and **dynamic Bayesian networks** (which include hidden Markov models and Kalman filters as special cases). Finally, Section 15.6 examines the problems faced when keeping track of more than one thing.

15.1 TIME AND UNCERTAINTY

We have developed our techniques for probabilistic reasoning in the context of *static* worlds, in which each random variable has a single fixed value. For example, when repairing a car, we assume that whatever is broken remains broken during the process of diagnosis; our job is to infer the state of the car from observed evidence, which also remains fixed.

Now consider a slightly different problem: treating a diabetic patient. As in the case of car repair, we have evidence such as recent insulin doses, food intake, blood sugar measurements, and other physical signs. The task is to assess the current state of the patient, including the actual blood sugar level and insulin level. Given this information, we can make a decision about the patient's food intake and insulin dose. Unlike the case of car repair, here the *dynamic* aspects of the problem are essential. Blood sugar levels and measurements thereof can change rapidly over time, depending on recent food intake and insulin doses, metabolic activity, the time of day, and so on. To assess the current state from the history of evidence and to predict the outcomes of treatment actions, we must model these changes.

The same considerations arise in many other contexts, such as tracking the location of a robot, tracking the economic activity of a nation, and making sense of a spoken or written sequence of words. How can dynamic situations like these be modeled?

15.1.1 States and observations

TIME SLICE

We view the world as a series of snapshots, or **time slices**, each of which contains a set of random variables, some observable and some not.[1] For simplicity, we will assume that the same subset of variables is observable in each time slice (although this is not strictly necessary in anything that follows). We will use $\mathbf{X}_t$ to denote the set of state variables at time t, which are assumed to be unobservable, and $\mathbf{E}_t$ to denote the set of observable evidence variables. The observation at time t is $\mathbf{E}_t = \mathbf{e}_t$ for some set of values $\mathbf{e}_t$.

Consider the following example: You are the security guard stationed at a secret underground installation. You want to know whether it's raining today, but your only access to the outside world occurs each morning when you see the director coming in with, or without, an umbrella. For each day t, the set $\mathbf{E}_t$ thus contains a single evidence variable $Umbrella_t$ or $\quad_t$ for short (whether the umbrella appears), and the set $\mathbf{X}_t$ contains a single state variable $Rain_t$ or $\quad_t$ for short (whether it is raining). Other problems can involve larger sets of variables. In the diabetes example, we might have evidence variables, such as $MeasuredBloodSugar_t$ and $PulseRate_t$, and state variables, such as $BloodSugar_t$ and $StomachContents_t$. (Notice that $BloodSugar_t$ and $MeasuredBloodSugar_t$ are not the same variable; this is how we deal with noisy measurements of actual quantities.)

The interval between time slices also depends on the problem. For diabetes monitoring, a suitable interval might be an hour rather than a day. In this chapter we assume the interval between slices is fixed, so we can label times by integers. We will assume that the state sequence starts at $t = 0$; for various uninteresting reasons, we will assume that evidence starts arriving at $t = 1$ rather than $t = 0$. Hence, our umbrella world is represented by state variables $\quad_0, \quad_1, \quad_2, \quad$ and evidence variables $\quad_1, \quad_2, \quad$. We will use the notation $\quad : \quad$ to denote the sequence of integers from $\quad$ to $\quad$ (inclusive), and the notation $\mathbf{X}_{a:b}$ to denote the set of variables from $\mathbf{X}_a$ to $\mathbf{X}_b$. For example, $\quad_{1:3}$ corresponds to the variables $\quad_1, \quad_2, \quad_3$.

[1] Uncertainty over *continuous* time can be modeled by **stochastic differential equations** (SDEs). The models studied in this chapter can be viewed as discrete-time approximations to SDEs.

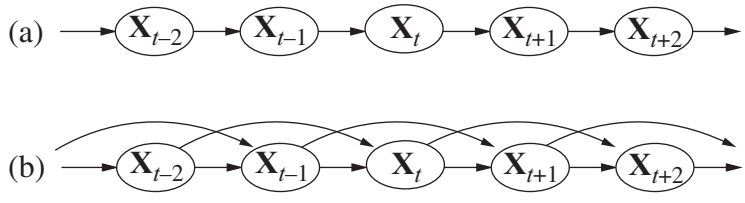

Figure 15.1 (a) Bayesian network structure corresponding to a first-order Markov process with state defined by the variables $\mathbf{X}_t$. (b) A second-order Markov process.

15.1.2 Transition and sensor models

With the set of state and evidence variables for a given problem decided on, the next step is to specify how the world evolves (the transition model) and how the evidence variables get their values (the sensor model).

The transition model specifies the probability distribution over the latest state variables, given the previous values, that is, $\mathbf{P}(\mathbf{X}_t \mid \mathbf{X}_{0:t-1})$. Now we face a problem: the set $\mathbf{X}_{0:t-1}$ is unbounded in size as t increases. We solve the problem by making a **Markov assumption**—that the current state depends on only a *finite fixed number* of previous states. Processes satisfying this assumption were first studied in depth by the Russian statistician Andrei Markov (1856–1922) and are called **Markov processes** or **Markov chains**. They come in various flavors; the simplest is the **first-order Markov process**, in which the current state depends only on the previous state and not on any earlier states. In other words, a state provides enough information to make the future conditionally independent of the past, and we have

MARKOV ASSUMPTION

MARKOV PROCESS

FIRST-ORDER MARKOV PROCESS

$$\mathbf{P}(\mathbf{X}_t \mid \mathbf{X}_{0:t-1}) = \mathbf{P}(\mathbf{X}_t \mid \mathbf{X}_{t-1}) \tag{15.1}$$

Hence, in a first-order Markov process, the transition model is the conditional distribution $\mathbf{P}(\mathbf{X}_t \mid \mathbf{X}_{t-1})$. The transition model for a second-order Markov process is the conditional distribution $\mathbf{P}(\mathbf{X}_t \mid \mathbf{X}_{t-2}, \mathbf{X}_{t-1})$. Figure 15.1 shows the Bayesian network structures corresponding to first-order and second-order Markov processes.

Even with the Markov assumption there is still a problem: there are infinitely many possible values of t. Do we need to specify a different distribution for each time step? We avoid this problem by assuming that changes in the world state are caused by a **stationary process**—that is, a process of change that is governed by laws that do not themselves change over time. (Don't confuse *stationary* with *static*: in a *static* process, the state itself does not change.) In the umbrella world, then, the conditional probability of rain, $\mathbf{P}(\ \ _t \mid \ \ _{t-1})$, is the same for all t, and we only have to specify one conditional probability table.

STATIONARY PROCESS

Now for the sensor model. The evidence variables $\mathbf{E}_t$ *could* depend on previous variables as well as the current state variables, but any state that's worth its salt should suffice to generate the current sensor values. Thus, we make a **sensor Markov assumption** as follows:

SENSOR MARKOV ASSUMPTION

$$\mathbf{P}(\mathbf{E}_t \mid \mathbf{X}_{0:t}, \mathbf{E}_{0:t-1}) = \mathbf{P}(\mathbf{E}_t \mid \mathbf{X}_t) \tag{15.2}$$

Thus, $\mathbf{P}(\mathbf{E}_t \mid \mathbf{X}_t)$ is our sensor model (sometimes called the **observation model**). Figure 15.2 shows both the transition model and the sensor model for the umbrella example. Notice the

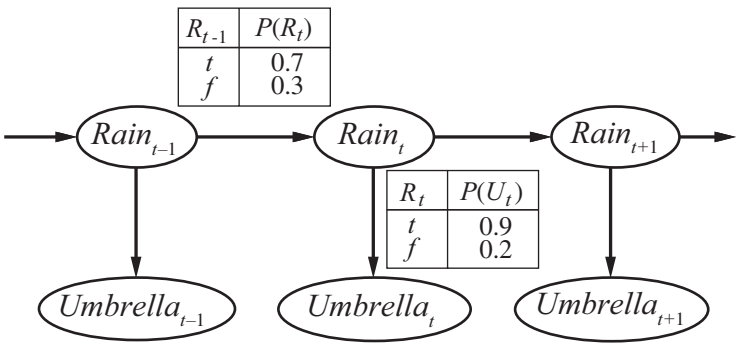

Figure 15.2 Bayesian network structure and conditional distributions describing the umbrella world. The transition model is $(Rain_t \mid Rain_{t-1})$ and the sensor model is $(Umbrella_t \mid Rain_t)$.

direction of the dependence between state and sensors: the arrows go from the actual state of the world to sensor values because the state of the world *causes* the sensors to take on particular values: the rain *causes* the umbrella to appear. (The inference process, of course, goes in the other direction; the distinction between the direction of modeled dependencies and the direction of inference is one of the principal advantages of Bayesian networks.)

In addition to specifying the transition and sensor models, we need to say how everything gets started—the prior probability distribution at time 0, $\mathbf{P}(\mathbf{X}_0)$. With that, we have a specification of the complete joint distribution over all the variables, using Equation (14.2). For any t,

$$\mathbf{P}(\mathbf{X}_{0:t}, \mathbf{E}_{1:t}) = \mathbf{P}(\mathbf{X}_0) \prod_{i=1}^{t} \mathbf{P}(\mathbf{X}_i \mid \mathbf{X}_{i-1}) \, \mathbf{P}(\mathbf{E}_i \mid \mathbf{X}_i) \tag{15.3}$$

The three terms on the right-hand side are the initial state model $\mathbf{P}(\mathbf{X}_0)$, the transition model $\mathbf{P}(\mathbf{X}_i \mid \mathbf{X}_{i-1})$, and the sensor model $\mathbf{P}(\mathbf{E}_i \mid \mathbf{X}_i)$.

The structure in Figure 15.2 is a first-order Markov process—the probability of rain is assumed to depend only on whether it rained the previous day. Whether such an assumption is reasonable depends on the domain itself. The first-order Markov assumption says that the state variables contain *all* the information needed to characterize the probability distribution for the next time slice. Sometimes the assumption is exactly true—for example, if a particle is executing a random walk along the -axis, changing its position by ± 1 at each time step, then using the -coordinate as the state gives a first-order Markov process. Sometimes the assumption is only approximate, as in the case of predicting rain only on the basis of whether it rained the previous day. There are two ways to improve the accuracy of the approximation:

1. Increasing the order of the Markov process model. For example, we could make a second-order model by adding $Rain_{t-2}$ as a parent of $Rain_t$, which might give slightly more accurate predictions. For example, in Palo Alto, California, it very rarely rains more than two days in a row.
2. Increasing the set of state variables. For example, we could add $Season_t$ to allow