DOI:10.1145/1859204.1859227

What are Bayesian networks and why are their applications growing across all fields?

BY ADNAN DARWICHE

Bayesian Networks

BAYESIAN NETWORKS HAVE been receiving considerable attention over the last few decades from scientists and engineers across a number of fields, including computer science, cognitive science, statistics, and philosophy. In computer science, the development of Bayesian networks was driven by research in artificial intelligence, which aimed at producing a practical framework for commonsense reasoning.[29] Statisticians have also contributed to the development of Bayesian networks, where they are studied under the broader umbrella of probabilistic graphical models.[5,11]

Interestingly enough, a number of other more specialized fields, such as genetic linkage analysis, speech recognition, information theory and reliability analysis, have developed representations that can be thought of as concrete instantiations or restricted cases of Bayesian networks. For example, pedigrees and their associated phenotype/genotype information, reliability block diagrams, and hidden Markov models (used in many fields including speech recognition and bioinformatics) can all be viewed as Bayesian networks. Canonical instances of Bayesian networks also exist and have been used to solve standard problems that span across domains such as computer vision, the Web, and medical diagnosis.

So what are Bayesian networks, and why are they widely used, either directly or indirectly, across so many fields and application areas? Intuitively, Bayesian networks provide a systematic and localized method for structuring probabilistic information about a situation into a coherent whole. They also provide a suite of algorithms that allow one to automatically derive many implications of this information, which can form the basis for important conclusions and decisions about the corresponding situation (for example, computing the overall reliability of a system, finding the most likely message that was sent across a noisy channel, identifying the most likely users that would respond to an ad, restoring a noisy image, mapping genes onto a chromosome, among others). Technically speaking, a Bayesian network is a compact representation of a probability distribution that is usually too large to be handled using traditional specifications from probability and statistics such as tables and equations. For example, Bayesian networks with thousands of variables have been constructed and reasoned about successfully, allowing one to efficiently represent and reason about probability distributions whose size is exponential in that number of variables (for example, in genetic link-

» key insights

- **Bayesian networks provide a systematic and localized method for structuring probabilistic information about a situation into a coherent whole, and are supported by a suite of inference algorithms.**
- **Bayesian networks have been established as a ubiquitous tool for modeling and reasoning under uncertainty.**
- **Many applications can be reduced to Bayesian network inference, allowing one to capitalize on Bayesian network algorithms instead of having to invent specialized algorithms for each new application.**

age analysis,[12] low-level vision,[34] and networks synthesized from relational models[4]).

For a concrete feel of Bayesian networks, Figure 1 depicts a small network over six binary variables. Every Bayesian network has two components: a directed acyclic graph (called a structure), and a set of conditional probability tables (CPTs). The nodes of a structure correspond to the variables of interest, and its edges have a formal interpretation in terms of probabilistic independence. We will discuss this interpretation later, but suffice to say here that in many practical applications, one can often interpret network edges as signifying direct causal influences. A Bayesian network must include a CPT for each variable, which quantifies the relationship between that variable and its parents in the network. For example, the CPT for variable *A* specifies the conditional probability distribution of *A* given its parents *F* and *T*. According to this CPT, the probability of *A* = `true` given *F* = `true` and *T* = `false` is *Pr*(*A*=`true`|*F* = `true`; *T* = `false`) = .9900 and is called a network *parameter*.[a]

A main feature of Bayesian networks is their guaranteed consistency and completeness as there is one and only one probability distribution that satisfies the constraints of a Bayesian network. For example, the network in Figure 1 induces a unique probability distribution over the 64 instantiations of its variables. This distribution provides enough information to attribute a probability to every event that can be expressed using the variables appearing in this network, for example, the probability of alarm tampering given no smoke and a report of people leaving the building.

Another feature of Bayesian networks is the existence of efficient algorithms for computing such probabilities without having to explic-

a Bayesian networks may contain continuous variables, yet our discussion here is restricted to the discrete case.

IMAGE BY ROBERT HODGIN

itly generate the underlying probability distribution (which would be computationally infeasible for many interesting networks). These algorithms, to be discussed in detail later, apply to any Bayesian network, regardless of its topology. Yet, the efficiency of these algorithms—and their accuracy in the case of approximation algorithms—may be quite sensitive to this topology and the specific query at hand. Interestingly enough, in domains such as genetics, reliability analysis, and information theory, scientists have developed algorithms that are indeed subsumed by the more general algorithms for Bayesian networks. In fact, one of the main objectives of this article is to raise awareness about these connections. The more general objective, however, is to provide an accessible introduction to Bayesian networks, which allows scientists and engineers to more easily identify problems that can be reduced to Bayesian network inference, putting them in a position where they can capitalize on the vast progress that has been made in this area over the last few decades.

Causality and Independence

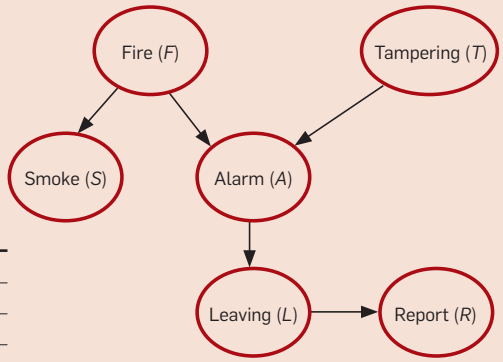
We will start by unveiling the central insight behind Bayesian networks that allows them to compactly represent very large distributions. Consider Figure 1 and the associated CPTs. Each probability that appears in one of these CPTs does specify a constraint that must be satisfied by the distribution induced by the network. For example, the distribution must assign the probability .01 to having smoke without fire, $Pr(S = \texttt{true} | F = \texttt{false})$, since this is specified by the CPT of variable S. These constraints, however, are not sufficient to pin down a unique probability distribution. So what additional information is being appealed to here?

The answer lies in the structure of a Bayesian network, which specifies additional constraints in the form of probabilistic conditional independencies. In particular, every variable in the structure is assumed to become independent of its non-descendants once its parents are known. In Figure 1, variable L is assumed to become independent of its non-descendants T, F, S once its parent A is known. In other words, once the value of variable A is known, the probability distribution of variable L will no longer change due to new information about variables T, F and S. Another example from Figure 1: variable A is assumed to become independent of its non-descendant S once its parents F and T are known. These independence constraints are known as the Markovian assumptions of a Bayesian network. Together with the numerical constraints specified by CPTs, they are satisfied by exactly one probability distribution.

Does this mean that every time a Bayesian network is constructed, one must verify the conditional independencies asserted by its structure? This really depends on the construction method. I will discuss three main methods in the section entitled "How Are Bayesian Networks Constructed?" that include subjective construction, synthesis from other specifications, and learning from data. The first method is the least systematic, but even in that case, one rarely thinks about conditional independence when constructing networks. Instead, one thinks about causality, adding the edge $X \rightarrow Y$ whenever X is perceived to be a direct cause of Y. This leads to a causal structure in which the Markovian assumptions read: each variable becomes independent of its non-effects once its direct causes are known. The ubiquity of Bayesian networks stems from the fact that people are quite good at identifying direct causes from a given set of variables, and at deciding whether the set of variables contains all of the relevant direct causes. This ability is all that one needs for constructing a causal structure.

The distribution induced by a Bayesian network typically satisfies additional independencies, beyond the Markovian ones discussed above. Moreover, all such independencies can be identified efficiently using a graphical test known as *d-separation*.[29] According to this test, variables X and

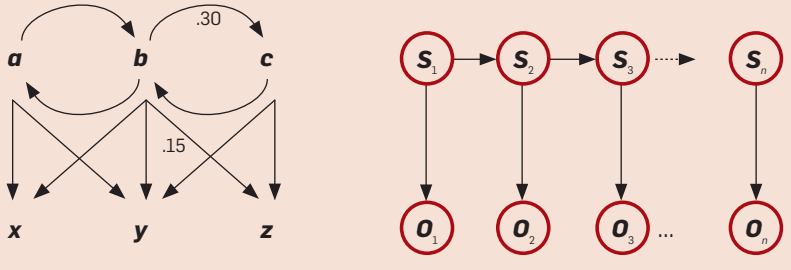
Figure 1. A Bayesian network with some of its conditional probability tables (CPTs).



Fire	Smoke	$\theta_{s|f}$
true	true	.90
false	true	.01

Fire	Tampering	Alarm	$\theta_{a|f,t}$
true	true	true	.5000
true	false	true	.9900
false	true	true	.8500
false	false	true	.0001

Figure 2. A hidden Markov model (HMM) and its corresponding dynamic Bayesian network (DBN).



(a) HMM (state diagram)

(b) DBN

Y are guaranteed to be independent given variables Z if every path between X and Y is blocked by Z. Intuitively, a path is blocked when it cannot be used to justify a dependence between X and Y in light of our knowledge of Z. For an example, consider the path $\alpha : S \leftarrow F \rightarrow A \leftarrow T$ in Figure 1 and suppose we know the alarm has triggered (that is, we know the value of variable A). This path can then be used to establish a dependence between variables S and T as follows. First, observing smoke increases the likelihood of fire since fire is a direct cause of smoke according to path α. Moreover, the increased likelihood of fire explains away tampering as a cause of the alarm, leading to a decrease in the probability of tampering (fire and tampering are two competing causes of the alarm according to path α). Hence, the path could be used to establish a dependence between S and T in this case. Variables S and T are therefore not independent given A due to the presence of this unblocked path. One can verify, however, that this path cannot be used to establish a dependence between S and T in case we know the value of variable F instead of A. Hence, the path is blocked by F.

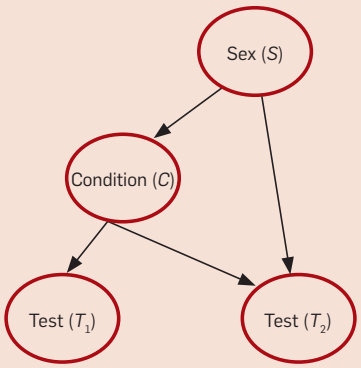
Even though we appealed to the notion of causality when describing the d-separation test, one can phrase and prove the test without any appeal to causality—we only need the Markovian assumptions. The full d-separation test gives the precise conditions under which a path between two variables is blocked, guaranteeing independence whenever all paths are blocked. The test can be implemented in time linear in the Bayesian network structure, without the need to explicitly enumerate paths as suggested previously.

The d-separation test can be used to directly derive results that have been proven for specialized probabilistic models used in a variety of fields. One example is hidden Markov models (HMMs), which are used to model dynamic systems whose states are not observable, yet their outputs are. One uses an HMM when interested in making inferences about these changing states, given the sequence of outputs they generate. HMMs are widely used in applications requiring temporal pattern recognition, including speech, handwriting, and gesture recognition; and various problems in bioinformatics.[31] Figure 2a depicts an HMM, which models a system with three states (a, b, c) and three outputs (x, y, z). The figure depicts the possible transitions between the system states, which need to be annotated by their probabilities. For example, state b can transition to states a or c, with a 30% chance of transitioning to state c. Each state can emit a number of observable outputs, again, with some probabilities. In this example, state b can emit any of the three outputs, with output z having a 15% chance of being emitted by this state.

This HMM can be represented by the Bayesian network in Figure 2b.[32] Here, variable S_t has three values a, b, c and represents the system state at time t, while variable O_t has the values x, y, z and represents the system output at time t. Using d-separation on this network, one can immediately derive the characteristic property of HMMs: once the state of the system at time t is known, its states and outputs at times $> t$ become independent of its states and outputs at times $< t$.

We also note the network in Figure 2b is one of the simplest instances of what is known as *dynamic Bayesian networks* (DBNs).[9] A number of extensions have been considered for HMMs, which can be viewed as more structured instances of DBNs. When proposing such extensions, however, one has the obligation of offering a corresponding algorithmic toolbox for inference. By viewing these extended HMMs as instances of Bayesian networks, however, one immediately inherits the corresponding Bayesian network algorithms for this purpose.

Figure 3. A Bayesian network that models a population, a medical condition, and two corresponding tests.



C = yes means that the condition C is present, and T_i = +ve means the result of test T_i is positive.

S	θ_s
male	.55

S	C	$\theta_{c|s}$
male	yes	.05
female	yes	.01

C	T_1	$\theta_{t_1|c}$
yes	+ve	.80
no	+ve	.20

S	C	T_2	$\theta_{t_2|s,c}$
male	yes	+ve	.80
male	no	+ve	.20
female	yes	+ve	.95
female	no	+ve	.05

How are Bayesian Networks Constructed?

One can identify three main methods for constructing Bayesian networks.[8] According to the first method, which is largely subjective, one reflects on their own knowledge or the knowledge of others (typically, perceptions about causal influences) and then captures them into a Bayesian network. The network in Figure 1 is an example of this construction method. The network structure of Figure 3 depicts another example, yet the parameters of this network can be obtained from more formal sources, such as population statistics and test specifications. According to this network, we have a population that is 55% males and 45% females, whose members can suffer from a medical condition C that is more likely to occur in males. Moreover, two diagnostic tests are available for detecting this condition, T_1 and T_2, with the second test being more effective on females. The CPTs of this network also reveal that the two tests are equally effective on males.

The second method for constructing Bayesian networks is based on automatically synthesizing them from some other type of formal knowledge.

Figure 4. A reliability block diagram (top), with a systematic method for mapping its blocks into Bayesian network fragments (bottom).

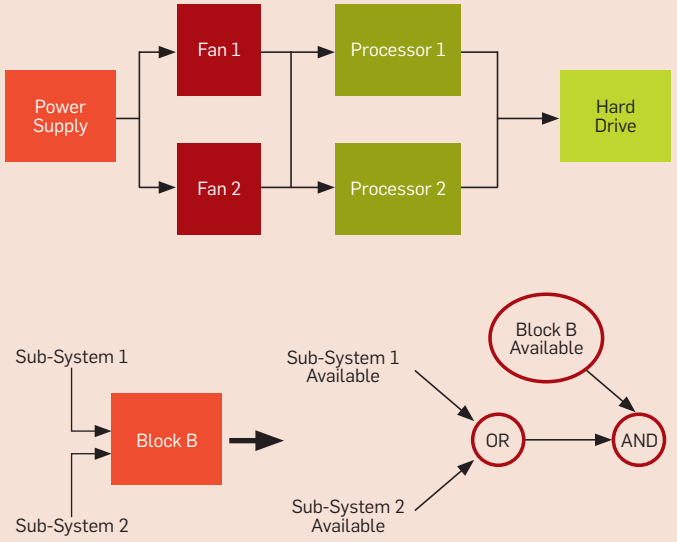
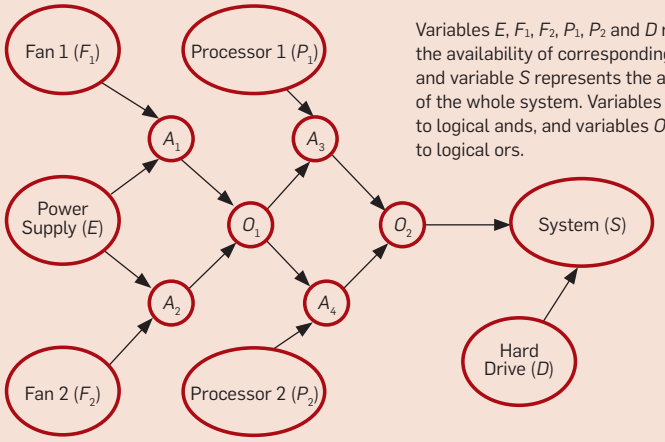


Figure 5. A Bayesian network generated automatically from a reliability block diagram.



For example, in many applications that involve system analysis, such as reliability and diagnosis, one can synthesize a Bayesian network automatically from a formal system design. Figure 4 depicts a reliability block diagram (RBD) used in reliability analysis. The RBD depicts system components and the dependencies between their availability. For example, Processor 1 requires either of the fans for its availability, and each of the fans requires power for its availability. The goal here is to compute the overall reliability of the system (probability of its availability) given the reliabilities of each of its components. Figure 4 shows also how one may systematically convert each block in an RBD into a Bayesian network fragment, while Figure 5 depicts the corresponding Bayesian network constructed according to this method. The CPTs of this figure can be completely constructed based on the reliabilities of individual components (not shown here) and the semantics of the transformation method.[8]

The third method for constructing Bayesian networks is based on learning them from data, such as medical records or student admissions data. Consider Figure 3 and the data set depicted in the table here as an example.

Sex S	Condition C	Test T_1	Test T_2
male	no	?	–ve
male	?	–ve	+ve
female	yes	+ve	?
⋮	⋮	⋮	⋮

Each row of the table corresponds to an individual and what we know about them. One can use such a data set to learn the network parameters given its structure, or learn both the structure and its parameters. Learning parameters only is an easier task computationally. Moreover, learning either structure or parameters always becomes easier when the data set is complete—that is, the value of each variable is known in each data record.

Since learning is an inductive process, one needs a principle of induction to guide the learning process. The two main principles for this purpose lead to the *maximum likelihood* and *Bayesian* approaches to learning (see, for example, the work of [5,8,17,22,27]). The maximum likelihood approach favors Bayesian networks that maximize the probability of observing the given data set. The Bayesian approach on the other hand uses the likelihood principle in addition to some prior information which encodes preferences on Bayesian networks.

Suppose we are only learning network parameters. The Bayesian approach allows one to put a prior distribution on the possible values of each network parameter. This prior distribution, together with the data set, induces a posterior distribution on the values of that parameter. One can then use this posterior to pick a value for that parameter (for example, the distribution mean). Alternatively, one can decide to avoid committing to a fixed parameter value, while computing answers to queries by averaging over all possible parameter values according to their posterior probabilities.

It is critical to observe here that the term "Bayesian network" does not necessarily imply a commitment to the Bayesian approach for learning networks. This term was coined by Judea Pearl[28] to emphasize three aspects: the often subjective nature of the information used in constructing these networks; the reliance on Bayes condi-

tioning when reasoning with Bayesian networks; and the ability to perform causal as well as evidential reasoning on these networks, which is a distinction underscored by Thomas Bayes.[1]

These learning approaches are meant to induce Bayesian networks that are meaningful independently of the tasks for which they are intended. Consider for example a network which models a set of diseases and a corresponding set of symptoms. This network may be used to perform *diagnostic* tasks, by inferring the most likely disease given a set of observed symptoms. It may also be used for *prediction* tasks, where we infer the most likely symptom given some diseases. If we concern ourselves with only one of these tasks, say diagnostics, we can use a more specialized induction principle that optimizes the diagnostic performance of the learned network. In machine learning jargon, we say we are learning a *discriminative model* in this case, as it is often used to discriminate among patients according to a predefined set of classes (for example, has cancer or not). This is to be contrasted with learning a *generative model*, which is to be evaluated based on its ability to generate the given data set, regardless of how it performs on any particular task.

We finally note that it is not uncommon to assume some canonical network structure when learning Bayesian networks from data, in order to reduce the problem of learning structure and parameters to the easier problem of learning parameters only. Perhaps the most common such structure is what is known as naïve Bayes: $C \rightarrow A_1, \ldots, C \rightarrow A_n$, where C is known as the class variable and variables $A_1,\ldots,A_n$ are known as attributes. This structure has proven to be very popular and effective in a number of applications, in particular classification and clustering.[14]

Canonical Bayesian Networks

A number of canonical Bayesian networks have been proposed for modeling some well-known problems in a variety of fields. For example, genetic linkage analysis is concerned with mapping genes onto a chromosome, utilizing the fact that the distance between genes is inversely proportional to the extent to which genes are linked (two genes are linked when it is more likely than not that their states are passed together from a single grandparent, instead of one state from each grandparent). To assess the likelihood of a linkage hypothesis, one uses a pedigree with some information about the genotype and phenotype of associated individuals. Such information can be systematically translated into a Bayesian network (see Figure 6), where the likelihood of a linkage hypothesis corresponds to computing the probability of an event with respect to this network.[12] By casting this problem in terms of inference on Bayesian networks, and by capitalizing on the state-of-the-art algorithms for this purpose, the scalability of genetic linkage analysis was advanced considerably, leading to the most efficient algorithms for exact linkage computations on general pedigrees (for example, the SUPER-

Figure 6. A Bayesian network generated automatically from a pedigree that contains three individuals.

Variables GP_{ij} and GM_{ij} represent the states of gene j for individual i, which are inherited from the father and mother of i, respectively. Variable P_{ij} represents the phenotype determined by gene j of individual i. Variables SP_{ij} and SM_{ij} are meant to model recombination events, which occur when the states of two genes inherited from a parent do not come from a single grandparent. The CPTs of these variables contain parameters which encode linkage hypotheses.

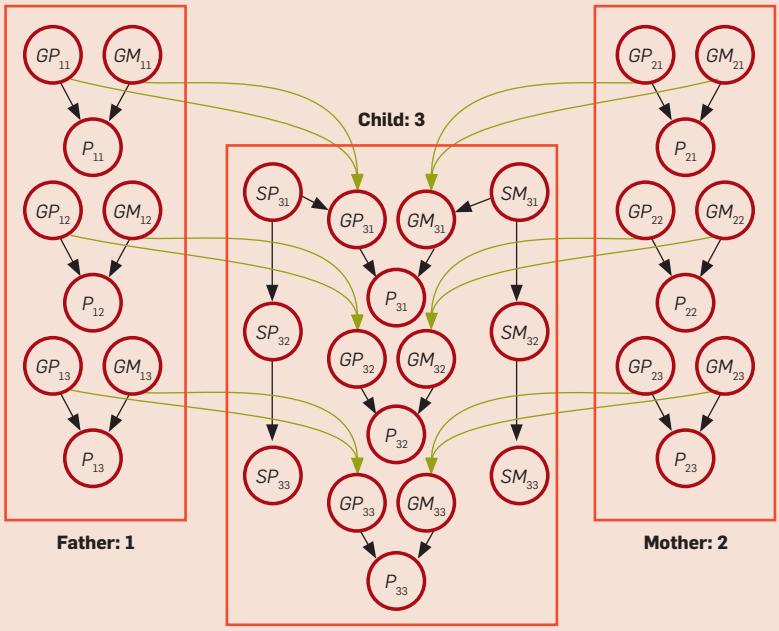
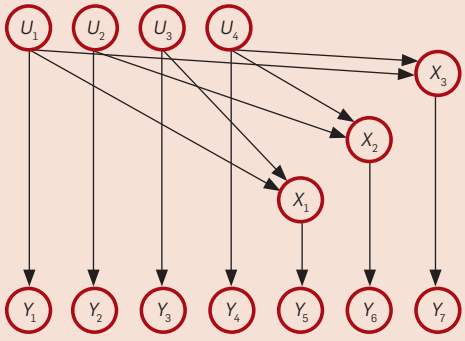


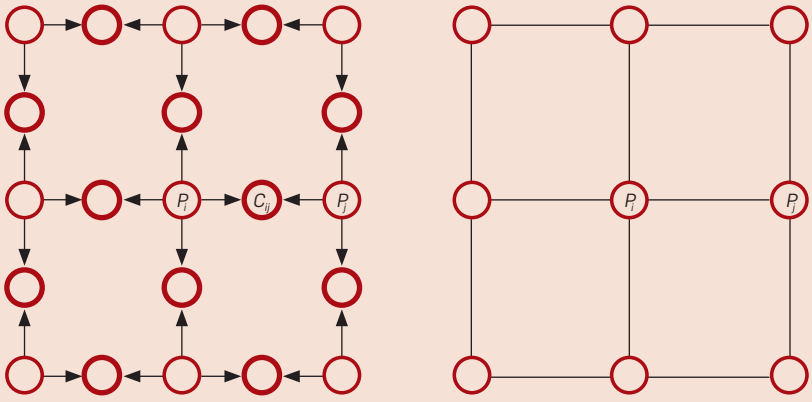
Figure 7. A Bayesian network that models a noisy channel with input ($U_1,\ldots,U_4, X_1,\ldots,X_3$) and output ($Y_1,\ldots,Y_7$).



Bits $X_1,\ldots,X_3$ are redundant coding bits and their CPTs capture the coding protocol. The CPTs of variables Y_i capture the noise model. This network can be used to recover the most likely sent message $U_1,\ldots,U_4$ given a received message $Y_1,\ldots,Y_7$.

Figure 8. Images from left to right: input, restored (using Bayesian network inference) and original.

Figure 9. Modeling low-level vision problems using two types of graphical models: Bayesian networks and MRFs.



(a) Bayesian network

(b) Markov random field (MRF)

LINK program initiated by Fishelson and Geiger[12]).

Canonical models also exist for modeling the problem of passing information over a noisy channel, where the goal here is to compute the most likely message sent over such a channel, given the channel output.[13] For example, Figure 7 depicts a Bayesian network corresponding to a situation where seven bits are sent across a noisy channel (four original bits and three redundant ones).

Another class of canonical Bayesian networks has been used in various problems relating to vision, including image restoration and stereo vision. Figure 8 depicts two examples of images that were restored by posing a query to a corresponding Bayesian network. Figure 9a depicts the Bayesian network in this case, where we have one node P_i for each pixel i in the image—the values p_i of P_i represent the gray levels of pixel i. For each pair of neighboring pixels, i and j, a child node C_{ij} is added with a CPT that imposes a smoothness constraint between the pixels. That is, the probability $Pr(C_{ij} = \texttt{true} | P_i = p_i, P_j = p_j)$ specified by the CPT decreases as the difference in gray levels $| p_i - p_j |$ increases. The only additional information needed to completely specify the Bayesian network is a CPT for each node P_i, which provides a prior distribution on the gray levels of each pixel i. These CPTs are chosen to give the highest probability to the gray level v_i appearing in the input image, with the prior probability $Pr(P_i = p_i)$ decreasing as the difference $| p_i - v_i |$ increases. By simply adjusting the domain and the prior probabilities of nodes P_i, while asserting an appropriate degree of smoothness using variables C_{ij}, one can use this model to perform other "pixel-labeling" tasks such as stereo vision, photomontage, and binary segmentation.[34] The formulation of these tasks as inference on Bayesian networks is not only elegant, but has also proven to be very powerful. For example, such inference is the basis for almost all top-performing stereo methods.[34]

Canonical Bayesian network models have also been emerging in recent years in the context of other important applications, such as the analysis of documents, and text. Many of these networks are based on topic models that view a document as arising from a set of unknown topics, and provide a framework for reasoning about the connections between words, documents, and underlying topics.[2,33] Topic models have been applied to many kinds of documents, including email, scientific abstracts, and newspaper archives, allowing one to utilize inference on Bayesian networks to tackle tasks such as measuring document similarity, discovering emergent topics, and browsing through documents based on their underlying content instead of traditional indexing schemes.

What Can One Do with a Bayesian Network?

Similar to any modeling language, the value of Bayesian networks is mainly tied to the class of queries they support.

Consider the network in Figure 3 for an example and the following queries: Given a male that came out positive on both tests, what is the probability he has the condition? Which group of the population is most likely to test negative on both tests? Considering the network in Figure 5: What is the overall reliability of the given system? What is the most likely configuration

of the two fans given that the system is unavailable? What single component can be replaced to increase the overall system reliability by 5%? Consider Figure 7: What is the most likely channel input that would yield the channel output 1001100? These are example questions that would be of interest in these application domains, and they are questions that can be answered systematically using three canonical Bayesian network queries.[8] A main benefit of using Bayesian networks in these application areas is the ability to capitalize on existing algorithms for these queries, instead of having to invent a corresponding specialized algorithm for each application area.

Probability of Evidence. This query computes the probability $Pr(\mathbf{e})$, where $\mathbf{e}$ is an assignment of values to some variables $\mathbf{E}$ in the Bayesian network—$\mathbf{e}$ is called a variable instantiation or evidence in this case. For example, in Figure 3, we can compute the probability that an individual will come out positive on both tests using the probability-of-evidence query $Pr(T_1 = +\texttt{ve}, T_2 = +\texttt{ve})$. We can also use the same query to compute the overall reliability of the system in Figure 5, $Pr(S = \texttt{avail})$. The decision version of this query is known to be *PP*-complete. It is also related to another common query, which asks for computing the probability $Pr(x|\mathbf{e})$ for each network variable X and each of its values x. This is known as the *node marginals* query.

Most Probable Explanation (MPE). Given an instantiation $\mathbf{e}$ of some variables $\mathbf{E}$ in the Bayesian network, this query identifies the instantiation $\mathbf{q}$ of variables $\mathbf{Q} = \overline{\mathbf{E}}$ that maximizes the probability $Pr(\mathbf{q}|\mathbf{e})$. In Figure 3, we can use an MPE query to find the most likely group, dissected by sex and condition, that will yield negative results for both tests, by taking the evidence $\mathbf{e}$ to be $T_1 = -\texttt{ve}; T_2 = -\texttt{ve}$ and $\mathbf{Q} = \{S, C\}$. We can also use an MPE query to restore images as shown in Figures 8 and 9. Here, we take the evidence $\mathbf{e}$ to be $C_{ij} = \texttt{true}$ for all i, j and $\mathbf{Q}$ to include P_i for all i. The decision version of MPE is known to be *NP*-complete and is therefore easier than the probability-of-evidence query under standard assumptions of complexity theory.

Maximum a Posteriori Hypothesis (MAP). Given an instantiation $\mathbf{e}$ of some

Figure 10. An arithmetic circuit for the Bayesian network $B \leftarrow A \rightarrow C$. Inputs labeled with θ variables correspond to network parameters, while those labeled with λ variables capture evidence.

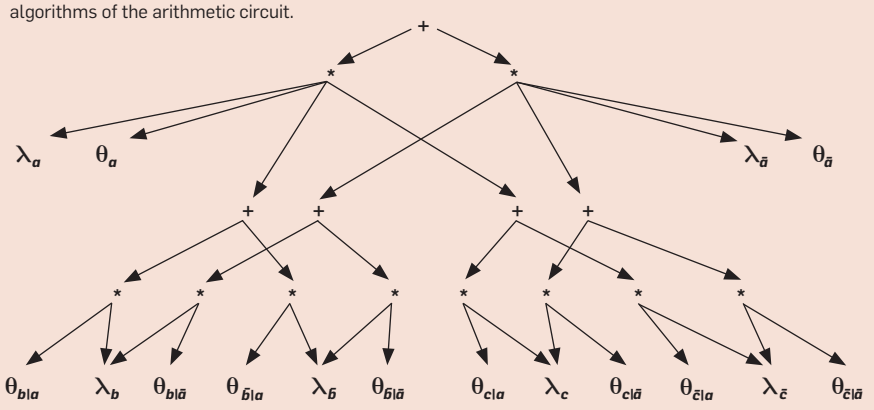


Figure 11. Two networks that represent the same set of conditional independencies.

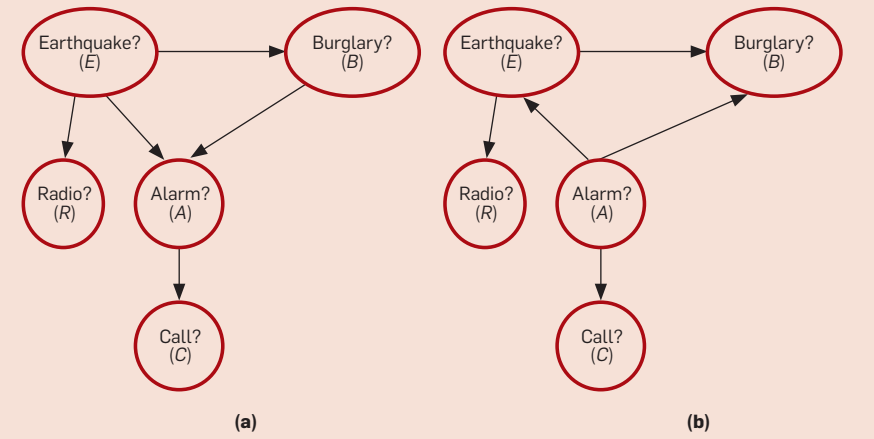
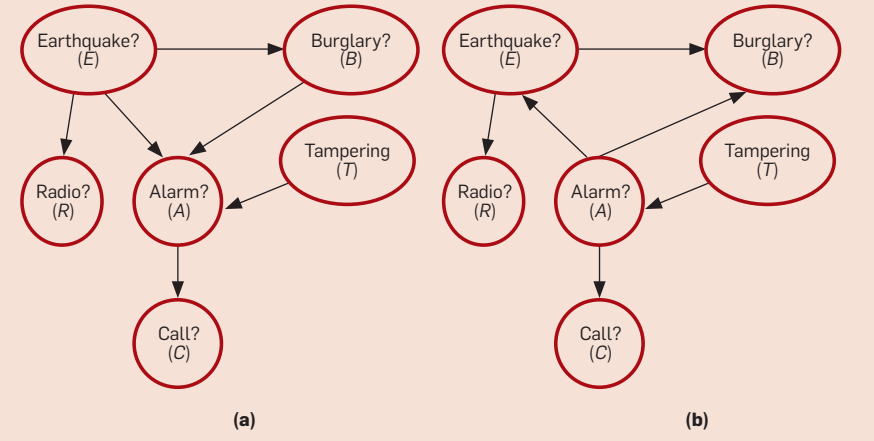


Figure 12. Extending Bayesian networks to account for interventions.



variables **E** in the Bayesian network, this query identifies the instantiation **q** of some variables $\mathbf{Q} \subseteq \overline{\mathbf{E}}$ that maximizes the probability $Pr(\mathbf{q}|\mathbf{e})$. Note the subtle difference with MPE queries: **Q** is a subset of variables $\overline{\mathbf{E}}$ instead of being all of these variables. MAP is a more difficult problem than MPE since its decision version is known to be NP^{PP}-complete, while MPE is only *NP*-complete. As an example of this query, consider Figure 5 and suppose we are interested in the most likely configuration of the two fans given that the system is unavailable. We can find this configuration using a MAP query with the evidence **e** being $S = \texttt{un_avail}$ and $\mathbf{Q} = \{F_1, F_2\}$.

One can use these canonical queries to implement more sophisticated queries, such as the ones demanded by sensitivity analysis. This is a mode of analysis that allows one to check the robustness of conclusions drawn from Bayesian networks against perturbations in network parameters (for example, see Darwiche[8]). Sensitivity analysis can also be used for automatically revising these parameters in order to satisfy some global constraints that are imposed by experts, or derived from the formal specifications of tasks under consideration. Suppose for example that we compute the overall system reliability using the network in Figure 5 and it turns out to be 95%. Suppose we wish this reliability to be no less than 99%: $Pr(S = \texttt{avail}) \geq 99\%$. Sensitivity analysis can be used to identify components whose reliability is relevant to achieving this objective, together with the new reliabilities they must attain for this purpose. Note that component reliabilities correspond to network parameters in this example.

How Well Do Bayesian Networks Scale?

Algorithms for inference on Bayesian networks fall into two main categories: exact and approximate algorithms. Exact algorithms make no compromises on accuracy and tend to be more expensive computationally. Much emphasis was placed on exact inference in the 1980s and early 1990s, leading to two classes of algorithms based on the concepts of *elimination*[10,24,36] and *conditioning*.[6,29] In their pure form, the complexity of these algorithms is exponential only in the network *treewidth*, which is a graph-theoretic parameter that measures the resemblance of a graph to a tree structure. For example, the treewidth of trees is ≤ 1 and, hence, inference on such tree networks is quite efficient. As the network becomes more and more connected, its treewidth increases and so does the complexity of inference. For example, the network in Figure 9 has a treewidth of n assuming an underlying image with $n \times n$ pixels. This is usually too high, even for modest-size images, to make these networks accessible to treewidth-based algorithms.

The pure form of elimination and conditioning algorithms are called *structure-based* as their complexity is sensitive only to the network structure. In particular, these algorithms will consume the same computational resources when applied to two networks that share the same structure, regardless of what probabilities are used to annotate them. It has long been observed that inference algorithms can be made more efficient if they also exploit the structure exhibited by network parameters, including determinism (0 and 1 parameters) and context-specific independence (independence that follows from network parameters and is not detectable by d-separation[3]). Yet, algorithms for exploiting parametric structure have only matured in the last few years, allowing one to perform exact inference on some networks whose treewidth is quite large (see survey[8]). Networks that correspond to genetic linkage analysis (Figure 6) tend to fall in this category[12] and so do networks that are synthesized from relational models.[4]

One of the key techniques for exploiting parametric structure is based on compiling Bayesian networks into arithmetic circuits, allowing one to reduce probabilistic inference to a process of circuit propagation;[7] see Figure 10. The size of these compiled circuits is determined by both the network topology and its parameters, leading to relatively compact circuits in some situations where the parametric structure is excessive, even if the network treewidth is quite high (for example, Chavira et al.[4]). Reducing inference to circuit propagation makes it also easier to support applications that require real-time inference, as in certain diagnosis applications.[25]

Around the mid-1990s, a strong belief started forming in the inference community that the performance of exact algorithms must be exponential in treewidth—this is before parametric structure was being exploited effectively. At about the same time, methods for automatically constructing Bayesian networks started maturing to the point of yielding networks whose treewidth is too large to be handled efficiently by exact algorithms at the time. This has led to a surge of interest in approximate inference algorithms, which are generally independent of treewidth. Today, approximate inference algorithms are the only choice for networks that have a high treewidth, yet lack sufficient parametric structure—the networks used in low-level vision applications tend to have this property. An influential class of approximate inference algorithms is based on reducing the inference problem to a constrained optimization problem, with *loopy belief propagation* and its generalizations as one key example.[29,35] Loopy belief propagation is actually the common algorithm of choice today for handling networks with very high treewidth, such as the ones arising in vision or channel coding applications. Algorithms based on stochastic sampling have also been pursued for a long time and are especially important for inference in Bayesian networks that contain continuous variables.[8,15,22] Variational methods provide another important class of approximation techniques[19,22] and are key for inference on some Bayesian networks, such as the ones arising in topic models.[2]

Causality, Again

One of the most intriguing aspects of Bayesian networks is the role they play in formalizing causality. To illustrate this point, consider Figure 11, which depicts two Bayesian network structures over the same set of variables. One can verify using d-separation that these structures represent the same set of conditional independencies. As such, they are representationally equivalent as they can induce the same set of probability distributions when augmented with appropriate

CPTs. Note, however, that the network in Figure 11a is consistent with common perceptions of causal influences, yet the one in Figure 11b violates these perceptions due to edges $A \rightarrow E$ and $A \rightarrow B$. Is there any significance to this discrepancy? In other words, is there some additional information that can be extracted from one of these networks, which cannot be extracted from the other? The answer is yes according to a body of work on causal Bayesian networks, which is concerned with a key question:[16,30] how can one characterize the additional information captured by a causal Bayesian network and, hence, what queries can be answered only by Bayesian networks that have a causal interpretation?

According to this body of work, only causal networks are capable of updating probabilities based on interventions, as opposed to observations. To give an example of this difference, consider Figure 11 again and suppose that we want to compute the probabilities of various events given that someone has tampered with the alarm, causing it to go off. This is an intervention, to be contrasted with an observation, where we know the alarm went off but without knowing the reason. In a causal network, interventions are handled as shown in Figure 12a: by simply adding a new direct cause for the alarm variable. This local fix, however, cannot be applied to the non-causal network in Figure 11b. If we do, we obtain the network in Figure 12b, which asserts the following (using d-separation): if we observe the alarm did go off, then knowing it was not tampered with is irrelevant to whether a burglary or an earthquake took place. This independence, which is counterintuitive, does not hold in the causal structure and represents one example of what may go wrong when using a non-causal structure to answer questions about interventions.

Causal structures can also be used to answer more sophisticated queries, such as counterfactuals. For example, the probability of "the patient would have been alive had he not taken the drug" requires reasoning about interventions (and sometimes might even require functional information, beyond standard causal Bayesian networks[30]). Other types of queries include ones for distinguishing between direct and indirect causes and for determining the sufficiency and necessity of causation.[30] Learning causal Bayesian networks has also been treated,[16,30] although not as extensively as the learning of general Bayesian networks.

One of the most intriguing aspects of Bayesian networks is the role they play in formalizing causality.

Beyond Bayesian Networks

Viewed as graphical representations of probability distributions, Bayesian networks are only one of several other models for this purpose. In fact, in areas such as statistics (and now also in AI), Bayesian networks are studied under the broader class of *probabilistic graphical models*, which include other instances such as Markov networks and chain graphs (for example, Edwards[11] and Koller and Friedman[22]). Markov networks correspond to undirected graphs, and chain graphs have both directed and undirected edges. Both of these models can be interpreted as compact specifications of probability distributions, yet their semantics tend to be less transparent than Bayesian networks. For example, both of these models include numeric annotations, yet one cannot interpret these numbers directly as probabilities even though the whole model can be interpreted as a probability distribution. Figure 9b depicts a special case of a Markov network, known as a Markov random field (MRF), which is typically used in vision applications. Comparing this model to the Bayesian network in Figure 9a, one finds that smoothness constraints between two adjacent pixels P_i and P_j can now be represented by a single undirected edge $P_i - P_j$ instead of two directed edges and an additional node, $P_i \rightarrow C_{ij} \leftarrow P_j$. In this model, each edge is associated with a function $f(P_i, P_j)$ over the states of adjacent pixels. The values of this function can be used to capture the smoothness constraint for these pixels, yet do not admit a direct probabilistic interpretation.

Bayesian networks are meant to model probabilistic beliefs, yet the interest in such beliefs is typically motivated by the need to make rational decisions. Since such decisions are often contemplated in the presence of uncertainty, one needs to know the likelihood and utilities associated with various decision outcomes. A

classical example in this regard concerns an oil wildcatter that needs to decide whether or not to drill for oil at a specific site, with an additional decision on whether to request seismic soundings that may help determine the geological structure of the site. Each of these decisions has an associated cost. Moreover, their potential outcomes have associated utilities and probabilities. The need to integrate these probabilistic beliefs, utilities and decisions has lead to the development of *Influence Diagrams*, which are extensions of Bayesian networks that include three types of nodes: chance, utility, and decision.[18] Influence diagrams, also called decision networks, come with a toolbox that allows one to compute optimal strategies: ones that are guaranteed to produce the highest expected utility.[20,22]

Bayesian networks have also been extended in ways that are meant to facilitate their construction. In many domains, such networks tend to exhibit regular and repetitive structures, with the regularities manifesting in both CPTs and network structure. In these situations, one can synthesize large Bayesian networks automatically from compact high-level specifications. A number of concrete specifications have been proposed for this purpose. For example, template-based approaches require two components for specifying a Bayesian network: a set of network templates whose instantiation leads to network segments, and a specification of which segments to generate and how to connect them together.[22,23] Other approaches include languages based on first-order logic, allowing one to reason about situations with varying sets of objects (for example, Milch et al.[26]).

The Challenges Ahead

Bayesian networks have been established as a ubiquitous tool for modeling and reasoning under uncertainty. The reach of Bayesian networks, however, is tied to their effectiveness in representing the phenomena of interest, and the scalability of their inference algorithms. To further improve the scope and ubiquity of Bayesian networks, one therefore needs sustained progress on both fronts. The main challenges on the first front lie in increasing the expressive power of Bayesian network representations, while maintaining the key features that have proven necessary for their success: modularity of representation, transparent graphical nature, and efficiency of inference. On the algorithmic side, there is a need to better understand the theoretical and practical limits of exact inference algorithms based on the two dimensions that characterize Bayesian networks: their topology and parametric structure.

With regard to approximate inference algorithms, the main challenges seem to be in better understanding their behavior to the point where we can characterize conditions under which they are expected to yield good approximations, and provide tools for practically trading off approximation quality with computational resources. Pushing the limits of inference algorithms will immediately push the envelope with regard to learning Bayesian networks since many learning algorithms rely heavily on inference. One cannot emphasize enough the importance of this line of work, given the extent to which data is available today, and the abundance of applications that require the learning of networks from such data. C

References

1. Bayes, T. An essay towards solving a problem in the doctrine of chances. *Phil. Trans. 3* (1963), 370–418. Reproduced in W.E. Deming.
2. Blei, D.M., Ng, A.Y. and Jordan, M.I. Latent Dirichlet allocation. *Journal of Machine Learning Research 3* (2003), 993–1022.
3. Boutilier, C., Friedman, N., Goldszmidt, M. and Koller, D. Context-specific independence in Bayesian networks. In *Proceedings of the 12th Conference on Uncertainty in Artificial Intelligence* (1996), 115–123.
4. Chavira, M., Darwiche, A. and Jaeger, M. Compiling relational Bayesian networks for exact inference. *International Journal of Approximate Reasoning 42*, 1-2 (May 2006) 4–20.
5. Cowell, R., Dawid, A., Lauritzen, S. and Spiegelhalter, D. *Probabilistic Networks and Expert Systems.* Springer, 1999.
6. Darwiche, A. Recursive conditioning. *Artificial Intelligence 126*, 1-2 (2001), 5–41.
7. Darwiche, A. A differential approach to inference in Bayesian networks. *Journal of the ACM 50*, 3 (2003).
8. Darwiche, A. *Modeling and Reasoning with Bayesian Networks.* Cambridge University Press, 2009.
9. Dean, T. and Kanazawa, K. A model for reasoning about persistence and causation. *Computational Intelligence 5*, 3 (1989), 142–150.
10. Dechter, R. Bucket elimination: A unifying framework for probabilistic inference. In *Proceedings of the 12th Conference on Uncertainty in Artificial Intelligence* (1996), 211–219.
11. Edwards, D. *Introduction to Graphical Modeling.* Springer, 2nd edition, 2000.
12. Fishelson, M. and Geiger, D. Exact genetic linkage computations for general pedigrees. *Bioinformatics 18*, 1 (2002), 189–198.
13. Frey, B. editor. *Graphical Models for Machine Learning and Digital Communication.* MIT Press, Cambridge, MA, 1998.
14. Friedman, N., Geiger, D. and Goldszmidt, M. Bayesian network classifiers. *Machine Learning 29*, 2-3 (1997), 131–163.
15. Gilks, W., Richardson, S. and Spiegelhalter, D. *Markov Chain Monte Carlo in Practice: Interdisciplinary Statistics.* Chapman & Hall/CRC, 1995.
16. Glymour, C. and Cooper, G. eds. *Computation, Causation, and Discovery.* MIT Press, Cambridge, MA, 1999.
17. Heckerman, D. A tutorial on learning with Bayesian networks. *Learning in Graphical Models.* Kluwer, 1998, 301–354.
18. Howard, R.A. and Matheson, J.E. Influence diagrams. *Principles and Applications of Decision Analysis, Vol. 2.* Strategic Decision Group, Menlo Park, CA, 1984, 719–762.
19. Jaakkola, T. Tutorial on variational approximation methods. *Advanced Mean Field Methods.* D. Saad and M. Opper, ed, MIT Press, Cambridge, MA, 2001, 129–160.
20. Jensen, F.V. and Nielsen, T.D. *Bayesian Networks and Decision Graphs.* Springer, 2007.
21. Jordan, M., Ghahramani, Z., Jaakkola, T. and Saul, L. An introduction to variational methods for graphical models. *Machine Learning 37*, 2 (1999), 183–233.
22. Koller, D. and Friedman, N. *Probabilistic Graphical Models: Principles and Techniques.* MIT Press, Cambridge, MA, 2009.
23. Koller, D. and Pfeffer, A. Object-oriented Bayesian networks. In *Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence* (1997), 302–313.
24. Lauritzen, S.L. and Spiegelhalter, D.J. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of Royal Statistics Society, Series B 50*, 2 (1998), 157–224.
25. Mengshoel, O., Darwiche, A., Cascio, K., Chavira, M., Poll, S. and Uckun, S. Diagnosing faults in electrical power systems of spacecraft and aircraft. In *Proceedings of the 20th Innovative Applications of Artificial Intelligence Conference* (2008), 1699–1705.
26. Milch, B., Marthi, B., Russell, S., Sontag, D., Ong, D. and Kolobov, A. BLOG: Probabilistic models with unknown objects. In *Proceedings of the International Joint Conference on Artificial Intelligence* (2005), 1352–1359.
27. Neapolitan, R. *Learning Bayesian Networks.* Prentice Hall, Englewood, NJ, 2004.
28. Pearl, J. Bayesian networks: A model of self-activated memory for evidential reasoning. In *Proceedings of the Cognitive Science Society* (1985), 329–334.
29. Pearl, J. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann, 1988.
30. Pearl, J. *Causality: Models, Reasoning, and Inference.* Cambridge University Press, 2000.
31. Durbin, A.K.R., Eddy, S. and Mitchison, G. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids.* Cambridge University Press, 1998.
32. Smyth, P., Heckerman, D. and Jordan, M. Probabilistic independence networks for hidden Markov probability models. *Neural Computation 9*, 2 (1997), 227–269.
33. Steyvers, M. and Griffiths, T. Probabilistic topic models. *Handbook of Latent Semantic Analysis.* T. K. Landauer, D. S. McNamara, S. Dennis, and W. Kintsch, eds. 2007, 427–448.
34. Szeliski, R., Zabih, R., Scharstein, D., Veksler, O., Kolmogorov, V., Agarwala, A., Tappen, M.F. and Rother, C. A comparative study of energy minimization methods for Markov random fields with smoothness-based priors. *IEEE Trans. Pattern Anal. Mach. Intell. 30*, 6 (2008), 1068–1080.
35. Yedidia, J., Freeman, W. and Weiss, Y. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Transactions on Information Theory 1*, 7 (2005), 2282–2312.
36. Zhang, N.L. and Poole, D. A simple approach to Bayesian network computations. In *Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence*, (1994), 171–178.

Adnan Darwiche (darwiche@cs.ucla.edu) is a professor and former chair of the computer science department at the University of California, Los Angeles, where he also directs the Automated Reasoning Group.

© 2010 ACM 0001-0782/10/1200 $10.00