

Probabilistic Networks — An Introduction to
Bayesian Networks and Influence Diagrams

Uffe B. Kjærulff
Department of Computer Science
Aalborg University

Anders L. Madsen
HUGIN Expert A/S

10 May 2005

Contents

Preface	iii
1 Networks	1
1.1 Graphs	4
1.2 Graphical Models	6
1.2.1 Variables	6
1.2.2 Nodes vs. Variables	7
1.2.3 Taxonomy of Nodes/Variables	7
1.2.4 Node Symbols	8
1.2.5 Summary of Notation	9
1.3 Evidence	9
1.4 Flow of Information in Causal Networks	10
1.4.1 Serial Connections	12
1.4.2 Diverging Connections	14
1.4.3 Converging Connections	15
1.4.4 Summary	17
1.5 Causality	17
1.6 Two Equivalent Irrelevance Criteria	19
1.6.1 d-Separation Criterion	21
1.6.2 Directed Global Markov Criterion	22
1.7 Summary	24
2 Probabilities	25
2.1 Basics	26
2.1.1 Definition of Probability	26
2.1.2 Events	27
2.1.3 Conditional Probability	28
2.1.4 Axioms	28
2.2 Probability Distributions for Variables	30
2.2.1 Rule of Total Probability	31
2.2.2 Graphical Representation	33
2.3 Probability Potentials	34
2.3.1 Normalization	34
2.3.2 Evidence Potentials	35

2.3.3	Potential Calculus	36
2.3.4	Barren Variables	39
2.4	Fundamental Rule and Bayes' Rule	40
2.4.1	Interpretation of Bayes' Rule	42
2.5	Bayes' Factor	44
2.6	Independence	45
2.6.1	Independence and DAGs	46
2.7	Chain Rule	48
2.8	Summary	50
3	Probabilistic Networks	53
3.1	Reasoning Under Uncertainty	54
3.1.1	Discrete Bayesian Networks	55
3.1.2	Linear Conditional Gaussian Bayesian Networks	60
3.2	Decision Making Under Uncertainty	63
3.2.1	Discrete Influence Diagrams	65
3.2.2	Linear-Quadratic CG Influence Diagrams	74
3.2.3	Limited Memory Influence Diagrams	78
3.3	Object-Oriented Probabilistic Networks	80
3.3.1	Chain Rule	85
3.3.2	Unfolded OOPNs	85
3.3.3	Instance Trees	85
3.3.4	Inheritance	86
3.4	Summary	86
4	Solving Probabilistic Networks	89
4.1	Probabilistic Inference	90
4.1.1	Inference in Discrete Bayesian Networks	90
4.1.2	Inference in LCG Bayesian Networks	103
4.2	Solving Decision Models	106
4.2.1	Solving Discrete Influence Diagrams	106
4.2.2	Solving LQCG Influence Diagrams	110
4.2.3	Relevance Reasoning	111
4.2.4	Solving LIMIDs	114
4.3	Solving OOPNs	118
4.4	Summary	118
	Bibliography	121
	List of symbols	125
	Index	129

Preface

This book presents the fundamental concepts of *probabilistic graphical models*, or *probabilistic networks* as they are called in this book. Probabilistic networks have become an increasingly popular paradigm for reasoning under uncertainty, addressing such tasks as diagnosis, prediction, decision making, classification, and data mining.

The book is intended to comprise an introductory part of a forthcoming monograph on practical aspects of probabilistic network models, aiming at providing a complete and comprehensive guide for practitioners that wish to construct decision support systems based on probabilistic networks. We think, however, that this introductory part in itself can serve as a valuable text for students and practitioners in the field.

We present the basic graph-theoretic terminology, the basic (Bayesian) probability theory, the key concepts of (conditional) dependence and independence, the different varieties of probabilistic networks, as well as methods for making inference in these kinds of models.

For a quick overview, the different kinds of probabilistic network models considered in the book can be characterized very briefly as follows:

Discrete Bayesian networks represent factorizations of joint probability distributions over finite sets of discrete random variables. The variables are represented by the nodes of the network, and the links of the network represent the properties of (conditional) dependences and independences among the variables as dictated by the distribution. For each variable is specified a set of local probability distributions conditional on the configuration of its conditioning (parent) variables.

Linear conditional Gaussian (CG) Bayesian networks represent factorizations of joint probability distributions over finite sets of random variables where some are discrete and some are continuous. Each continuous variable is assumed to follow a linear Gaussian distribution conditional on the configuration of its discrete parent variables.

Discrete influence diagrams represent sequential decision scenarios for a single decision maker and are (discrete) Bayesian networks augmented with (discrete) decision variables and (discrete) utility functions. An influence

diagram is capable of computing expected utilities of various decision options given the information known at the time of the decision.

Linear-quadratic CG influence diagrams combine linear CG Bayesian networks, discrete influence diagrams, and quadratic utility functions into a single framework supporting decision making under uncertainty with both continuous and discrete variables.

Limited-memory influence diagrams relax two fundamental assumptions of influence diagrams: The no-forgetting assumption implying perfect recall of past observations and decisions, and the assumption of a total order on the decisions. LIMIDs allow us to model more types of decision problems than the ordinary influence diagrams.

Object-oriented probabilistic networks are hierarchically specified probabilistic networks (i.e., one of the above), allowing the knowledge engineer to work on different levels of abstraction, as well as exploiting the usual concepts of encapsulation and inheritance known from object-oriented programming paradigms.

The book provides numerous examples, hopefully helping the reader to gain a good understanding of the various concepts, some of which are known to be hard to understand at a first encounter.

Even though probabilistic networks provide an intuitive language for constructing knowledge-based models for reasoning under uncertainty, knowledge engineers can often benefit from a deeper understanding of the principles underlying these models. For example, knowing the rules for reading statements of dependence and independence encoded in the structure of a network may prove very valuable in evaluating if the network correctly models the dependence and independence properties of the problem domain. This, in turn, may be crucial to achieving, for example, correct posterior probability distributions from the model. Also, having a basic understanding of the relations between the structure of a network and the complexity of inference may prove useful in the model construction phase, avoiding structures that are likely to result in problems of poor performance of the final decision support system.

The book will present such basic concepts, principles, and methods underlying probabilistic models, which practitioners need to acquaint themselves with.

In Chapter 1, we describe the fundamental concepts of the graphical language used to construct probabilistic networks as well as the rules for reading statements of (conditional) dependence and independence encoded in network structures.

Chapter 2 presents the uncertainty calculus used in probabilistic networks to represent the numerical counterpart of the graphical structure, namely classical (Bayesian) probability calculus. We shall see how a basic axiom of probability calculus leads to recursive factorizations of joint probability distributions into products of conditional probability distributions, and how such factorizations along with local statements of conditional independence naturally can be expressed in graphical terms.

In Chapter 3 we shall see how putting the basic notions of Chapters 1 and 2 together we get the notion of discrete Bayesian networks. Also, we shall acquaint the reader with a range of derived types of network models, including conditional Gaussian models where discrete and continuous variables co-exist, influence diagrams that are Bayesian networks augmented with decision variables and utility functions, limited-memory influence diagrams that allow the knowledge engineer to reduce model complexity through assumptions about limited memory of past events, and object-oriented models that allow the knowledge engineer to construct hierarchical models consisting of reusable submodels. Finally, in Chapter 4 we explain the principles underlying inference in these different kinds of probabilistic networks.

Aalborg, 10 May 2005

Uffe B. Kjærulff, *Aalborg University*
Anders L. Madsen, *HUGIN Expert A/S*

Chapter 1

Networks

Probabilistic networks are graphical models of (causal) interactions among a set of variables, where the variables are represented as nodes (also known as vertices) of a graph and the interactions (direct dependences) as directed links (also known as arcs and edges) between the nodes. Any pair of unconnected/non-adjacent nodes of such a graph indicates (conditional) independence between the variables represented by these nodes under particular circumstances that can easily be read from the graph. Hence, probabilistic networks capture a set of (conditional) dependence and independence properties associated with the variables represented in the network.

Graphs have proven themselves a very intuitive language for representing such dependence and independence statements, and thus provide an excellent language for communicating and discussing dependence and independence relations among problem-domain variables. A large and important class of assumptions about dependence and independence relations expressed in factorized representations of joint probability distributions can be represented very compactly in a class of graphs known as acyclic, directed graphs (DAGs).

Chain graphs are a generalization of DAGs, capable of representing a broader class of dependence and independence assumptions (Frydenberg 1989, Wermuth & Lauritzen 1990). The added expressive power comes, however, at the cost of a significant increase in the semantic complexity, making specification of joint probability factors much less intuitive. Thus, despite their expressive power, chain graph models have gained very little popularity as practical models for decision support systems, and we shall therefore focus exclusively on models that factorize according to DAGs.

As indicated above, probabilistic networks is a class of probabilistic models that have gotten their name from the fact that the joint probability distributions represented by these models can be naturally described in graphical terms, where the nodes of a graph (or network) represent variables over which a joint probability distribution is defined and the presence and absence of links represent dependence and independence properties among the variables.

Probabilistic networks can be seen as compact representations of “fuzzy” cause-effect rules that, contrary to ordinary (logical) rule based systems, are capable of performing deductive and abductive reasoning as well as intercausal reasoning. Deductive reasoning (sometimes referred to as *causal reasoning*) follows the direction of the causal links between variables of a model; e.g., knowing that a person has caught a cold we can conclude (with high probability) that the person has fever and a runny nose (see Figure 1.1). Abductive reasoning

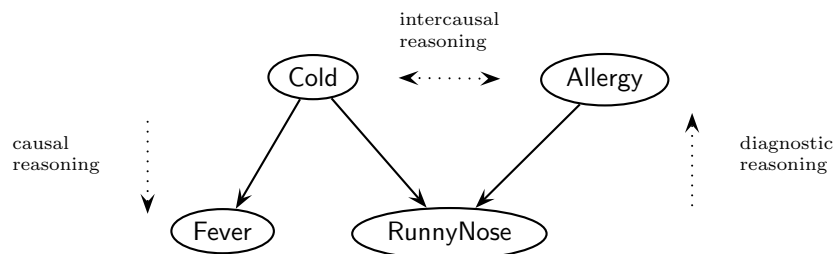


Figure 1.1: Causal networks support not only causal and diagnostic reasoning (also known as deductive and abductive reasoning, respectively) but also intercausal reasoning (explaining away): Observing **Fever** makes us believe that **Cold** is the cause of **RunnyNose**, thereby reducing significantly our belief that **Allergy** is the cause of **RunnyNose**.

(sometimes referred to as *diagnostic reasoning*) goes against the direction of the causal links; e.g., observing that a person has a runny nose provides supporting evidence for either cold or allergy being the correct diagnosis.

The property, however, that sets inference in probabilistic networks apart from other automatic reasoning paradigms is its ability to make *intercausal* reasoning: Getting evidence that supports solely a single hypothesis (or a subset of hypotheses) automatically leads to decreasing belief in the unsupported, competing hypotheses. This property is often referred to as the *explaining away* effect. For example, in Figure 1.1, there are two competing causes of runny nose. Observing fever, however, provides strong evidence that cold is the cause of the problem, while our belief in allergy being the cause decreases substantially (i.e., it is explained away by the observation of fever). The ability of probabilistic networks to automatically perform such intercausal inference is a key contribution to their reasoning power.

Often the graphical aspect of a probabilistic network is referred to as its *qualitative* aspect, and the probabilistic, numerical part as its *quantitative* aspect. This chapter is devoted to the qualitative aspect of probabilistic networks, which is characterized by DAGs where the nodes represent random variables, decision variables, or utility functions, and where the links represent direct dependences, informational constraints, or they indicate the domains of utility functions. The appearances differ for the different kinds of nodes (see Page 8), whereas the appearance of the links do not (see Figure 1.2).

Bayesian networks contain only random variables, and the links represent direct dependences (often, but not necessarily, causal relationships) among the variables. The causal network in Figure 1.1 shows the qualitative part of a Bayesian network.

Influence diagrams can be considered as Bayesian networks augmented with decision variables and utility functions, and provide a language for sequential decision problems for a single decision maker, where there is a fixed order among the decisions. Figure 1.2 shows an influence diagram, which is the causal network in Figure 1.1 augmented with two decision variables (the rectangular shaped nodes) and two utility functions (the diamond shaped nodes). First, given

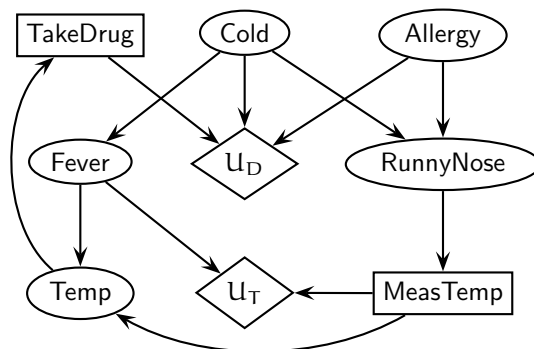


Figure 1.2: An influence diagram representing a sequential decision problem: First the decision maker should decide whether or not to measure the body temperature (**MeasTemp**) and then, based on the outcome of the measurement (if any), decide which drug to take (if any). The diagram is derived from the causal network in Figure 1.1 by augmenting it with decision variables and utility functions.

the runny-nose symptom, the decision maker must decide whether or not to measure the body temperature (**MeasTemp** has states **no** and **yes**). There is a utility function associated with this decision, represented by the node labeled U_T , which could encode, for example, the cost of measuring the temperature (in terms of time, inconvenience, etc), given the presence or absence of fever. If measured, the body temperature will then be known to the decision maker (represented by the random variable **Temp**) prior to making the decision on which drug to take, represented by the variable **TakeDrug**. This decision variable could, for example, have the states **aspirin**, **antihistamine**, and **noDrugs**. The utility (U_D) of taking a particular drug depends on the actual drug (if any) and the true cause of the symptom(s).

In Chapter 3, we describe the semantics of probabilistic network structures in much more detail, and introduce yet another kind of node that represents network instances and another kind of link that represents bindings between real nodes and place-holder nodes of network instances.

In Section 1.1 we introduce some basic graph notation that shall be used throughout the book. Section 1.2 discusses the notion of variables, which is the key entity of probabilistic networks. Another key concept is that of “evidence”, which we shall touch upon in Section 1.3. Maintaining a causal perspective in the model construction process can prove very valuable, as mentioned briefly in Section 1.5. Sections 1.4 and 1.6 are devoted to an in-depth treatment on the principles and rules for flow of information in DAGs. We carefully explain the properties of the three basic types of connections in a DAG (i.e., serial, diverging, and converging connections) through examples, and show how these combine directly into the d-separation criterion and how they support intercausal (explaining away) reasoning. We also present an alternative to the d-separation criterion known as the directed global Markov property, which in many cases proves to be a more efficient method for reading off dependence and independence statements of a DAG.

1.1 Graphs

A graph is a pair $\mathcal{G} = (V, E)$, where V is a finite set of distinct nodes and $E \subseteq V \times V$ is a set of links. An ordered pair $(u, v) \in E$ denotes a *directed link* from node u to node v , and u is said to be a *parent* of v and v a *child* of u . The set of parents and children of a node v shall be denoted by $\text{pa}(v)$ and $\text{ch}(v)$, respectively.

As we shall see later, depending on what they represent, nodes are displayed as labelled circles, ovals, or polygons, directed links as arrows, and undirected links as lines. Figure 1.3a shows a graph with 8 nodes and 8 links (all directed), where, for example, the node labeled E has two parents labeled T and L.¹ The labels of the nodes are referring to (i) the names of the nodes, (ii) the names of the variables represented by the nodes, or (iii) descriptive labels associated with the variables represented by the nodes.²

We often use the intuitive notation $u \xrightarrow{\mathcal{G}} v$ to denote $(u, v) \in E$ (or just $u \rightarrow v$ if \mathcal{G} is understood). If $(u, v) \in E$ and $(v, u) \in E$, the link between u and v is an *undirected link*, denoted by $\{u, v\} \in E$ or $u \xleftrightarrow{\mathcal{G}} v$ (or just $u - v$). We shall use the notation $u \sim v$ to denote that $u \rightarrow v$, $v \rightarrow u$, or $u - v$. Nodes u and v are said to be *connected* in \mathcal{G} if $u \overset{\mathcal{G}}{\sim} v$. If $u \rightarrow v$ and $w \rightarrow v$, then these links are said to meet *head-to-head* at v .

If E does not contain undirected links, then \mathcal{G} is a *directed graph* and if E does not contain directed links, then it is an *undirected graph*. As mentioned above, we shall not deal with mixed cases of both directed and undirected links.

A *path* $\langle v_1, \dots, v_n \rangle$ is a sequence of distinct nodes such that $v_i \sim v_{i+1}$ for each $i = 1, \dots, n - 1$; the *length* of the path is $n - 1$. The path is a *directed path* if $v_i \rightarrow v_{i+1}$ for each $i = 1, \dots, n - 1$; v_i is then an *ancestor* of v_j and v_j a *descendant* of v_i for each $j > i$. The set of ancestors and descendants of v are

¹This is the structure of the Bayesian network discussed in Example 29 on page 58.

²See Section 1.2 for the naming conventions used for nodes and variables.

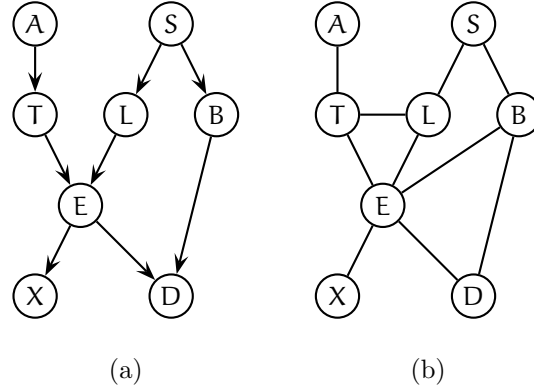


Figure 1.3: (a) A acyclic, directed graph (DAG). (b) Moralized graph.

denoted $\text{an}(v)$ and $\text{de}(v)$, respectively. The set $\text{nd}(v) = V \setminus \text{de}(v) \cup \{v\}$ are called the *non-descendants* of v . The *ancestral set* $\text{An}(\mathbf{U}) \subseteq V$ of a set $\mathbf{U} \subseteq V$ of a graph $\mathcal{G} = (V, E)$ is the set of nodes $\mathbf{U} \cup \bigcup_{u \in \mathbf{U}} \text{an}(u)$.

A path $\langle v_1, \dots, v_n \rangle$ from v_1 to v_n of an undirected graph, $\mathcal{G} = (V, E)$, is *blocked* by a set $S \subseteq V$ if $\{v_2, \dots, v_{n-1}\} \cap S \neq \emptyset$. There is a similar concept for paths of acyclic, directed graphs (see below), but the definition is somewhat more complicated (see Proposition 4 on page 21).

A graph $\mathcal{G} = (V, E)$ is *connected* if for any pair $\{u, v\} \subseteq V$ there is a path $\langle u, \dots, v \rangle$ in \mathcal{G} . A connected graph $\mathcal{G} = (V, E)$ is a *polytree* (also known as a *singly connected graph*) if for any pair $\{u, v\} \subseteq V$ there is a unique path $\langle u, \dots, v \rangle$ in \mathcal{G} . A directed polytree with only a single orphan node is called a (*rooted*) *tree*.

A *cycle* is a path, $\langle v_1, \dots, v_n \rangle$, of length greater than two with the exception that $v_1 = v_n$; a *directed cycle* is defined in the obvious way. A directed graph with no directed cycles is called an *acyclic, directed graph* or simply a *DAG*; see Figure 1.3a for an example. The undirected graph obtained from a DAG, \mathcal{G} , by replacing all its directed links with undirected ones is known as the *skeleton* of \mathcal{G} .

Let $\mathcal{G} = (V, E)$ be a DAG. The undirected graph, $\mathcal{G}^m = (V, E^m)$, where

$$E^m = \{\{u, v\} \mid u \text{ and } v \text{ are connected or have a common child in } \mathcal{G}\},$$

is called the *moral graph* of \mathcal{G} . That is, \mathcal{G}^m is obtained from \mathcal{G} by first adding undirected links between pairs of unconnected nodes that share a common child, and then replacing all directed links with undirected links; see Figure 1.3b for an example.

1.2 Graphical Models

On a structural (or qualitative) level, probabilistic network models are graphs with the nodes representing variables and utility functions, and the links representing different kinds of relations among the variables and utility functions.

1.2.1 Variables

A *variable* represents an exhaustive set of mutually exclusive events, referred to as the *domain* of the variable. These events are also often called states, levels, values, choices, options, etc. The domain of a variable can be discrete or continuous; discrete domains are always finite.

Example 1 The following list are examples of domains of variables:

$$\begin{aligned} &\{F, T\} \\ &\{\text{red, green, blue}\} \\ &\{1, 3, 5, 7\} \\ &\{-1.7, 0, 2.32, 5\} \\ &\{< 0, 0-5, > 5\} \\ &]-\infty; \infty[\\ &\{]-\infty; 0],]0; 5],]5; 10\} \end{aligned}$$

where F and T stands for “false” and “true”, respectively. The penultimate domain in the above list represents a domain for a continuous variable; the remaining ones represent domains for discrete variables. ■

Throughout this book we shall use capital letters (possibly indexed) to denote variables or sets of variables and lower case letters (possibly indexed) to denote particular values of variables. Thus, $X = x$ may either denote the fact that variable X attains the value x or the fact that the set of variables $X = (X_1, \dots, X_n)$ attains the (vector) of values $x = (x_1, \dots, x_n)$. By $\text{dom}(X) = (x_1, \dots, x_{\|X\|})$ we shall denote the domain of X , where $\|X\| = |\text{dom}(X)|$ is the number of possible distinct values of X . If $X = (X_1, \dots, X_n)$, then $\text{dom}(X)$ is the Cartesian product (or product space) over the domains of the variables in X . Formally,

$$\text{dom}(X) = \text{dom}(X_1) \times \dots \times \text{dom}(X_n),$$

and thus $\|X\| = \prod_i \|X_i\|$. For two (sets of) variables X and Y we shall write either $\text{dom}(X \cup Y)$ or $\text{dom}(X, Y)$ to denote $\text{dom}(X) \times \text{dom}(Y)$. If $z \in \text{dom}(Z)$, then by z_X we shall denote the projection of z to $\text{dom}(X)$, where $X \cap Z \neq \emptyset$.

Example 2 Assume that $\text{dom}(X) = \{F, T\}$ and $\text{dom}(Y) = \{\text{red, green, blue}\}$. Then $\text{dom}(X, Y) = \{(F, \text{red}), (F, \text{green}), (F, \text{blue}), (T, \text{red}), (T, \text{green}), (T, \text{blue})\}$. For $z = (F, \text{blue})$ we get $z_X = F$ and $z_Y = \text{blue}$. ■

Chance Variables and Decision Variables

There are basically two categories of variables, namely variables representing random events and variables representing choices under the control of some, typically human, agent. Consequently, the first category of variables is often referred to as *random variables* and the second category as *decision variables*. Note that a random variable can depend functionally on other variables in which case it is sometimes referred to as a *deterministic (random) variable*. Sometimes it is important to distinguish between truly random variables and deterministic variables, but unless this distinction is important we shall treat them uniformly, and refer to them simply as “random variables”, or just “variables”.

The problem of identifying those entities of a domain that qualify as variables is not necessarily trivial. Also, it can be non-trivial tasks to identify the “right” set of variables as well as appropriate sets of states for these variables. A more detailed discussion of these questions are, however, outside the scope of this introductory text.

1.2.2 Nodes vs. Variables

The notions of variables and nodes are often used interchangeably in models containing neither decision variables nor utility functions (e.g., Bayesian networks). For models that contain decision variables and utility functions it is convenient to distinguish between variables and nodes, as a node does not necessarily represent a variable. In this book we shall therefore maintain that distinction.

As indicated above, we shall use lower-case letters u, v, w (or sometimes α, β, γ , etc.) to denote nodes, and upper case letters U, V, W to denote sets of nodes. Node names shall sometimes be used in the subscripts of variable names to identify the variables corresponding to nodes. For example, if v is a node representing a variable, then we denote that variable by X_v . If v represents a utility function, then $X_{\text{pa}(v)}$ denotes the domain of the function, which is a set of chance and/or decision variables.

1.2.3 Taxonomy of Nodes/Variables

For convenience, we shall use the following terminology for classifying variables and/or nodes of probabilistic networks.

First, as discussed above, there are three main classes of nodes in probabilistic networks, namely nodes representing chance variables, nodes representing decision variables, and nodes representing utility functions. We define the *category* of a node to represent this dimension of the taxonomy.

Second, chance and decision variables as well as utility functions can be discrete or continuous. This dimension of the taxonomy shall be characterized by the *kind* of the variable or node.

Finally, for discrete chance and decision variables, we shall distinguish between labeled, Boolean, numbered, and interval variables. For example, re-

ferring to Example 1 on page 6, the first domain is the domain of a Boolean variable, the second and the fifth domains are domains of labeled variables, the third and the fourth domains are domains of numbered variables, and the last domain is the domain of an interval variable. This dimension of the taxonomy is referred to as the *subtype* of discrete variables, and is useful for providing mathematical expressions of specifications of conditional probability tables and utility tables.

Figure 1.4 summarizes the node/variable taxonomy.

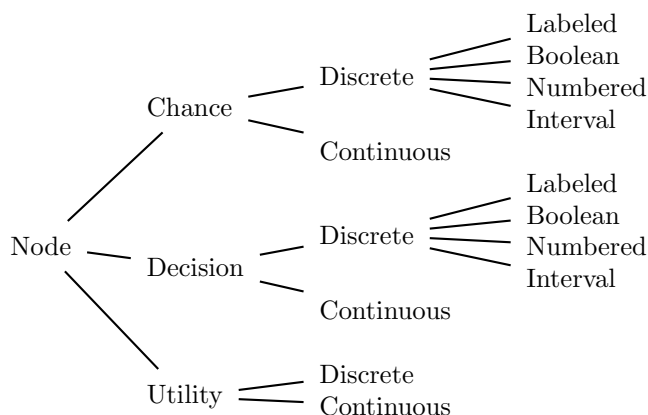


Figure 1.4: The taxonomy for nodes/variables. Note that the subtype dimension only applies for discrete chance and decision variables.

1.2.4 Node Symbols

Throughout this book we shall use ovals to indicate discrete chance variables, rectangles to indicate discrete decision variables, and diamonds to indicate discrete utility functions. Continuous variables and utility functions are indicated with double borders. See Table 1.1 for an overview.

Category	Kind	Symbol
Chance	Discrete	○
	Continuous	◌◌
Decision	Discrete	□
	Continuous	◌◌
Utility	Discrete	◇
	Continuous	◌◌

Table 1.1: Node symbols.

1.2.5 Summary of Notation

Table 1.2 summarizes the notation used for nodes (upper part), variables (middle part), and utility functions (lower part).

S, U, V, W	sets of nodes
V	set of nodes of a model
V_Δ	the subset of V that represent discrete variables
V_Γ	the subset of V that represent continuous variables
u, v, w, \dots	nodes
$\alpha, \beta, \gamma, \dots$	nodes
X, Y_i, Z_k	variables or sets of variables
X_W	subset of variables corresponding to set of nodes W
\mathcal{X}	the set of variables of a model; note that $\mathcal{X} = \mathcal{X}_V$
\mathcal{X}_W	subset of \mathcal{X} , where $W \subseteq V$
X_u, X_α	variables corresponding to nodes u and α , respectively
x, y_i, z_k	configurations/states of (sets of) variables
x_Y	projection of configuration x to $\text{dom}(Y)$, $X \cap Y \neq \emptyset$
\mathcal{X}_C	the set of chance variables of a model
\mathcal{X}_D	the set of decision variables of a model
\mathcal{X}_Δ	the subset of discrete variables of \mathcal{X}
\mathcal{X}_Γ	the subset of continuous variables of \mathcal{X}
\mathcal{U}	the set of utility functions of a model
V_U	the subset of V representing utility functions
$u(X)$	utility function $u \in \mathcal{U}$ with the of variables X as domain

Table 1.2: Notation used for nodes, variables, and utility functions.

1.3 Evidence

A key inference task with a probabilistic network is computation of posterior probabilities of the form $P(x|\varepsilon)$, where, in general, ε is *evidence* (i.e., information) received from external sources about the (possible) states/values of a subset of the variables of the network. For a set of discrete evidence variables, X , the evidence appears in the form of a *likelihood distribution* over the states of X ; also often called an *evidence function* (or *potential*³) for X . An evidence function, \mathcal{E}_X , for X is a function $\mathcal{E}_X : \text{dom}(X) \rightarrow \mathbb{R}^+$. For a set of continuous evidence variables, Y , the evidence appears in the form of a vector of real values, one for each variable in Y . Thus, the evidence function, \mathcal{E}_Y , is a function $\mathcal{E}_Y : Y \rightarrow \mathbb{R}$.

³See Section 2.3 on page 34.

Example 3 If $\text{dom}(X) = (x_1, x_2, x_3)$, then $\mathcal{E}_X = (1, 0, 0)$ is an evidence function indicating that $X = x_1$ with certainty. If $\mathcal{E}_X = (1, 2, 0)$, then with certainty $X \neq x_3$ and $X = x_2$ is twice as likely as $X = x_1$. ■

An evidence function that assigns a zero probability to all but one state is often said to provide *hard evidence*; otherwise, it is said to provide *soft evidence*. We shall often leave out the ‘hard’ or ‘soft’ qualifier, and simply talk about evidence if the distinction is immaterial. Hard evidence on a variable X is also often referred to as *instantiation* of X or that X has been *observed*. Note that, as soft evidence is a more general kind of evidence, hard evidence can be considered a special kind of soft evidence.

We shall attach the label $\boxed{\mathcal{E}}$ to nodes representing variables with hard evidence and the label $\boxed{\mathcal{E}}$ to nodes representing variables with soft (or hard) evidence. For example, hard evidence on variable X (like $\mathcal{E}_X = (1, 0, 0)$ in Example 3 on the page before) is indicated as shown in Figure 1.5(a) and soft evidence (like $\mathcal{E}_X = (1, 2, 0)$ in Example 3 on the preceding page) is indicated as shown in Figure 1.5(b).

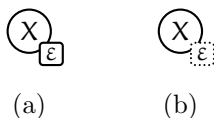


Figure 1.5: (a) Hard evidence on X . (b) Soft (or hard) evidence on X .

1.4 Flow of Information in Causal Networks

The DAG of a Bayesian network model is a compact graphical representation of the dependence and independence properties of the joint probability distribution represented by the model. In this section we shall study the rules for flow of information in DAGs, where each link represents a causal mechanism; for example, $\text{Flu} \rightarrow \text{Fever}$ represents the fact that Flu is a cause of Fever. Collectively, these rules define a criterion for reading off the properties of relevance and irrelevance encoded in such *causal networks*.⁴

As a basis for our considerations we shall consider the following small fictitious example.

Example 4 (Burglary or Earthquake (Pearl 1988)) Mr. Holmes is working in his office when he receives a *phone call* from his neighbor Dr. Watson, who tells Mr. Holmes that his *alarm* has gone off. Convinced that a *burglar* has broken into his house, Holmes rushes to his car and heads for home. On his way home, he listens to the radio, and in the *news* it is reported that there has been

⁴We often use the terms *relevance* and *irrelevance* to refer to pure graphical statements corresponding to, respectively, (probabilistic) dependence and independence among variables.

a small *earthquake* in the area. Knowing that earthquakes have a tendency to make burglar alarms go off, he returns to his work.

The causal network in Figure 1.6 shows the five relevant variables (all of which are Boolean; i.e., they have states F and T) and the entailed causal relationships. Notice that all of the links are causal: Burglary and earthquake can cause the alarm to go off, earthquake can cause a report on earthquake in the radio news, and the alarm can cause Dr. Watson to call Mr. Holmes. ■

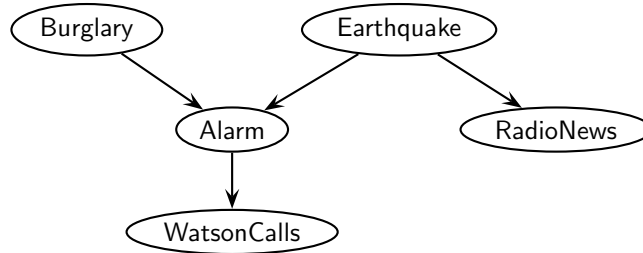


Figure 1.6: Causal network corresponding to the “Burglary or Earthquake” story of Example 4 on the preceding page.

The fact that a DAG is a compact representation of dependence/relevance and independence/irrelevance statements can be acknowledged from the DAG in Figure 1.6. Table 1.3 on the next page lists a subset of these statements where each statement takes the form “variables A and B are (in)dependent given that the values of some other variables, C , are known”, where the set C is minimal in the sense that removal of any element from C would violate the statement. If we include also non-minimal C -sets, the total number of dependence and independence statements will be 53, clearly illustrating the fact that even small probabilistic network models encode a very large number of such statements. Moderate sized networks may encode thousands or even millions of dependence and independence statements.

To learn how to read such statements from a DAG it is convenient to consider each possible basic kind of connection in a DAG. To illustrate these, consider the DAG in Figure 1.6. We see three different kinds of connections:

- *Serial connections:*
 - Burglary \rightarrow Alarm \rightarrow WatsonCalls
 - Earthquake \rightarrow Alarm \rightarrow WatsonCalls
- *Diverging connections:*
 - Alarm \leftarrow Earthquake \rightarrow RadioNews
- *Converging connections:*
 - Burglary \rightarrow Alarm \leftarrow Earthquake.

A	B	C	A and B are independent given C
Burglary	Earthquake	WatsonCalls	No
Burglary	Earthquake	Alarm	No
Burglary	WatsonCalls		No
Burglary	RadioNews	WatsonCalls	No
Burglary	RadioNews	Alarm	No
Earthquake	WatsonCalls		No
Alarm	RadioNews		No
RadioNews	WatsonCalls		No
Burglary	Earthquake		Yes
Burglary	WatsonCalls	Alarm	Yes
Burglary	RadioNews		Yes
Earthquake	WatsonCalls	Alarm	Yes
Alarm	RadioNews	Earthquake	Yes
RadioNews	WatsonCalls	Earthquake	Yes
RadioNews	WatsonCalls	Alarm	Yes

Table 1.3: 15 of the total of 53 dependence and independence statements encoded in the DAG of Figure 1.6. Each of the listed statements is minimal in the sense that removal of any element from the set C would violate the statement that A and B are (in)dependent given C.

In the following sub-sections we analyze each of these three possible kinds of connections in terms of their ability to transmit information given evidence and given no evidence on the middle variable, and we shall derive a general rule for reading statements of dependence and independence from a DAG. Also, we shall see that it is the converging connection that provides the ability of probabilistic networks to perform intercausal reasoning (explaining away).

1.4.1 Serial Connections

Let us consider the serial connection (causal chain) depicted in Figure 1.7, referring to Example 4 on page 10.

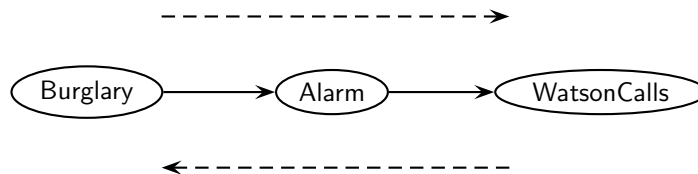


Figure 1.7: Serial connection (causal chain) with no hard evidence on Alarm. Evidence on Burglary will affect our belief about the state of WatsonCalls and vice versa.

We need to consider two cases, namely with and without hard evidence (see Section 1.3 on page 9) on the middle variable (**Alarm**).

First, assume we do not have definite knowledge about the state of **Alarm**. Then evidence about **Burglary** will make us update our belief about the state of **Alarm**, which in turn will make us update our belief about the state of **WatsonCalls**. The opposite is also true: If we receive information about the state of **WatsonCalls**, that will influence our belief about the state of **Alarm**, which in turn will influence our belief about **Burglary**.

So, in conclusion, as long as we do not know the state of **Alarm** for sure, information about either **Burglary** or **WatsonCalls** will influence our belief on the state of the other variable. If, for example, receiving the information (from some other source) that either his own or Dr. Watson's alarm had gone off, Mr. Holmes would still revise his belief about **Burglary** upon receiving the phone call from Dr. Watson. This is illustrated in Figure 1.7 on the facing page by the two dashed arrows, signifying that evidence may be transmitted through a serial connection as long as we do not have definite knowledge about the state of the middle variable.

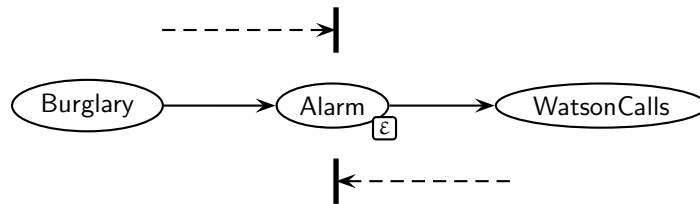


Figure 1.8: Serial connection (causal chain) with hard evidence on **Alarm**. Evidence on **Burglary** will have no affect on our belief about the state of **WatsonCalls** and vice versa.

Next, assume we do have definite knowledge about the state of **Alarm** (see Figure 1.8). Now, given that we have hard evidence on **Alarm** any information about the state of **Burglary** will not make us change our belief about **WatsonCalls** (provided **Alarm** is the only cause of **WatsonCalls**; i.e., that the model is correct). Also, information about **WatsonCalls** will have no influence on our belief about **Burglary** when the state of **Alarm** is known for sure.

In conclusion, when the state of the middle variable of a serial connection is known for sure (i.e., we have hard evidence on it), then flow of information between the other two variables cannot take place through this connection. This is illustrated in Figure 1.8 by the two dashed arrows ending at the observed variable, indicating that flow of information is blocked.

Note that soft evidence on the middle variable is insufficient to block the flow of information over a serial connection. Assume, for example, that we have gotten unreliable information (i.e., soft evidence) that Mr. Holmes' alarm has gone off; i.e., we are not absolutely certain that the alarm has actually gone off. In that case, information about **Burglary** (**WatsonCalls**) will potentially make us revise our belief about the state of **Alarm**, and hence influence our belief on

WatsonCalls (Burglary). Thus, soft evidence on the middle variable is not enough to block the flow of information over a serial connection.

The general rule for flow of information in serial connections can thus be stated as follows:

Proposition 1 (Serial connection) *Information may flow through a serial connection $X \rightarrow Y \rightarrow Z$ unless the state of Y is known.*

1.4.2 Diverging Connections

Consider the diverging connection depicted in Figure 1.9, referring to Example 4 on page 10.

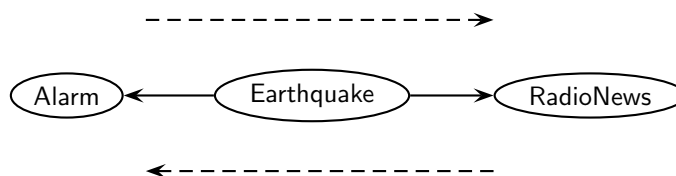


Figure 1.9: Diverging connection with no evidence on **Earthquake**. Evidence on **Alarm** will affect our belief about the state of **RadioNews** and vice versa.

Again, we consider the cases with and without hard evidence on the middle variable (**Earthquake**).

First, assume we do not know the state of **Earthquake** for sure. Then receiving information about **Alarm** will influence our belief about **Earthquake**, as earthquake is a possible explanation for alarm. The updated belief about the state of **Earthquake** will in turn make us update our belief about the state of **RadioNews**. The opposite case (i.e., receiving information about **RadioNews**) will lead to a similar conclusion. So, we get a result that is similar to the result for serial connections, namely that information can be transmitted through a diverging connection if we do not have definite knowledge about the state of the middle variable. This result is illustrated in Figure 1.9.

Next, assume the state of **Earthquake** is known for sure (i.e., we have received hard evidence on that variable). Now, if information is received about the state of the either **Alarm** or **RadioNews**, then this information is not going to change our belief about the state of **Earthquake**, and consequently we are not going to update our belief about the other, yet unobserved, variable. Again, this result is similar to the case for serial connections, and is illustrated in Figure 1.10 on the facing page.

Again, note that soft evidence on the middle variable is not enough to block the flow of information. Thus, the general rule for flow of information in diverging connections can be stated as follows:

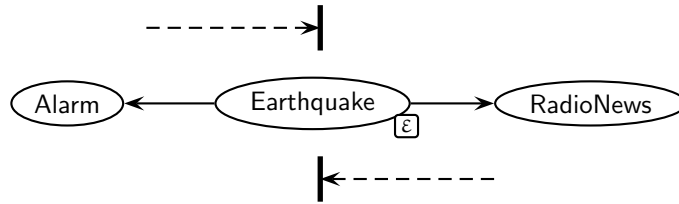


Figure 1.10: Diverging connection with hard evidence on **Earthquake**. Evidence on **Alarm** will not affect our belief about the state of **RadioNews** and vice versa.

Proposition 2 (Diverging connection) *Information may flow through a diverging connection $X \leftarrow Y \rightarrow Z$ unless the state of Y is known.*

1.4.3 Converging Connections

Consider the converging connection depicted in Figure 1.11, referring to Example 4 on page 10.

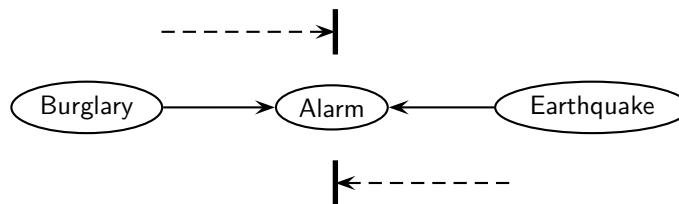


Figure 1.11: Converging connection with no evidence on **Alarm** or any of its descendants. Information about **Burglary** will not affect our belief about the state of **Earthquake** and vice versa.

First, if no evidence is available about the state of **Alarm**, then information about the state of **Burglary** will not provide any derived information about the state of **Earthquake**. In other words, burglary is not an indicator of earthquake, and vice versa (again, of course, assuming correctness of the model). Thus, contrary to serial and diverging connections, a converging connection will not transmit information if no evidence is available for the middle variable. This fact is illustrated in Figure 1.11.

Second, if evidence is available on **Alarm**, then information about the state of **Burglary** will provide an explanation for the evidence that was received about the state of **Alarm**, and thus either confirm or dismiss **Earthquake** as the cause of the evidence received for **Alarm**. The opposite, of course, also holds true. Again, contrary to serial and diverging connections, converging connections allow transmission of information whenever evidence about the middle variable is available. This fact is illustrated in Figure 1.12 on the following page.

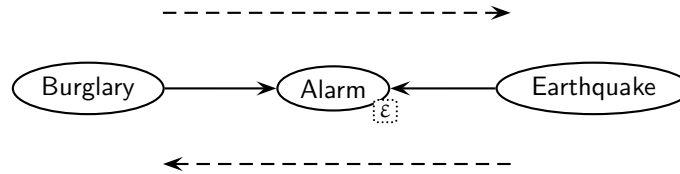


Figure 1.12: Converging connection with (possibly soft) evidence on Alarm or any of its descendants. Information about Burglary will affect our belief about the state of Earthquake and vice versa.

The rule illustrated in Figure 1.11 on the page before tells us that if nothing is known about a common effect of two (or more) causes, then the causes are independent; i.e., receiving information about one of them will have no impact on the belief about the other(s). However, as soon as some evidence is available on a common effect the causes become dependent. If, for example, Mr. Holmes receives a phone call from Dr. Watson, telling him that his burglar alarm has gone off, burglary and earthquake become competing explanations for this effect, and receiving information about the possible state of one of them obviously either confirms or dismisses the other one as the cause of the (possible) alarm. Note that even if the information received from Dr. Watson might not be totally reliable (amounting to receiving soft evidence on Alarm), Burglary and Earthquake still become dependent.

The general rule for flow of information in converging connections can then be stated as:

Proposition 3 (Converging connection) *Information may flow through a converging connection $X \rightarrow Y \leftarrow Z$ if evidence on Y or one of its descendants is available.*

intercausal Inference (Explaining Away)

The property of converging connections, $X \rightarrow Y \leftarrow Z$, that information about the state of X (Z) provides an explanation for an observed effect on Y , and hence confirms or dismisses Z (X) as the cause of the effect, is often referred to as the “explaining away” effect or as “intercausal inference”. For example, getting a radio report on earthquake provides strong evidence that the earthquake is responsible for a burglar alarm, and hence explaining away a burglary as the cause of the alarm.

The ability to perform intercausal inference is unique for graphical models, and is one of the key differences between automatic reasoning systems based on probabilistic networks and systems based on, for example, production rules. In a rule based system we would need dedicated rules for taking care of intercausal reasoning.

1.4.4 Summary

The analyzes in Sections 1.4.1–1.4.3 show that in terms of flow of information the serial and the diverging connections are identical, whereas the converging connection acts opposite to the former two (see Table 1.4). More specifically, it

	No evidence	Soft evidence	Hard evidence
Serial	open	open	closed
Diverging	open	open	closed
Converging	closed	open	open

Table 1.4: Flow of information in serial, diverging, and converging connections as a function of the type of evidence available on the middle variable.

takes hard evidence to close serial and diverging connections; otherwise, they allow flow of information. On the other hand, to close a converging connection no evidence (soft nor hard) must be available neither for the middle variable of the connection nor any of its descendants; otherwise, it allows flow of information.

1.5 Causality

Causality plays an important role in the process of constructing probabilistic network models. There are a number of reasons why proper modeling of causal relations is important or helpful, although it is not strictly necessary to have the directed links of a model follow a causal interpretation. We shall only very briefly touch upon the issue of causality, and stress a few important points about causal modeling. The reader is referred to the literature for an in-depth treatment of the subject (Pearl 2000, Spirtes, Glymour & Scheines 2000, Lauritzen 2001).

A variable X is said to be a direct cause of Y if setting the value of X by force, the value of Y may change and there is no other variable Z that is a direct cause of Y such that X is a direct cause of Z .

As an example, consider the variables *Flu* and *Fever*. Common sense tells us that flu is a cause of fever, not the other way around (see Figure 1.13). This

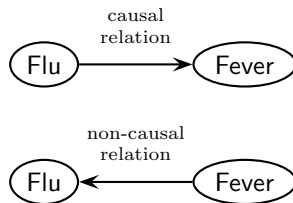


Figure 1.13: Influenza causes fever, not the other way around.

fact can be verified from the thought experiments of forcefully setting the states

of Flu and Fever: Killing fever with an aspirin or by taking a cold shower will have no effect on the state of Flu, whereas eliminating a flu would make the body temperature go back to normal (assuming flu is the only effective cause of fever).

To correctly represent the dependence and independence relations that exist among a set of variables of a problem domain it is very useful to have the causal relations among the variables be represented in terms of directed links from causes to effects. That is, if X is a direct cause of Y , we should make sure to add a directed link from X to Y . If done the other way around (i.e., $Y \rightarrow X$), we may end up with a model that do not properly represent the dependence and independence relations of the problem domain.

It is a common modeling mistake to let arrows point from effect to cause, leading to faulty statements of (conditional) dependence and independence and, consequently, faulty inference. For example, in the “Burglary or Earthquake” example (page 10), one might put a directed link from `WatsonCalls` to `Alarm` because the fact that Dr. Watson makes a phone call to Mr. Holmes “points to” the fact that Mr. Holmes’ alarm has gone off, etc. Experience shows that this kind of reasoning is very common when people are building their first probabilistic networks, and is probably due to a mental flow-of-information model, where evidence acts as the ‘input’ and the derived conclusions as the ‘output’.

Using this faulty modeling approach, the “Burglary or Earthquake” model in Figure 1.6 on page 11 would have all its links reversed (see Figure 1.14). In Section 1.4, we shall present methods for deriving statements about dependences and independences in causal networks, from which the model in Figure 1.14 leads to the conclusions that `Burglary` and `Earthquake` are dependent when nothing is known about `Alarm`, and that `Alarm` and `RadioNews` are dependent whenever evidence about `Earthquake` is available. Neither of these conclusions are, of course, true, and will make the model make wrong inferences.

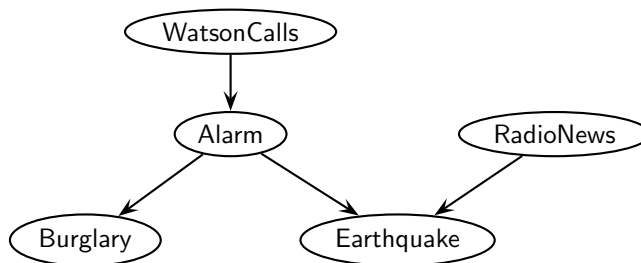


Figure 1.14: Wrong model for the “Burglary or Earthquake” story of Example 4 on page 10, where the links are directed from effects to causes, leading to faulty statements of (conditional) dependence and independence.

Although one does not *have* to construct models where the links can be interpreted as causal relations, as the above example shows, it makes the model

much more intuitive and eases the process of getting the dependence and independence relations right.

Another reason why respecting a causal modeling approach is important is that it significantly eases the process of eliciting the conditional probabilities of the model. If $Y \rightarrow X$ does not reflect a causal relationship, it can potentially be difficult to specify the conditional probability of $X = x$ given $Y = y$. For example, it might be difficult for Mr. Holmes to specify the probability that a burglar has broken into his house given that he knows the alarm has gone off, as the alarm might have gone off for other reasons. Thus, specifying the probability that the alarm goes off given its possible causes might be easier and more natural, providing a sort of complete description of a local phenomenon.

In Example 28 on page 56, we shall briefly return to the issue of the importance of correctly modeling the causal relationships in probabilistic networks.

1.6 Two Equivalent Irrelevance Criteria

Propositions 1–3 comprise the components needed to formulate a general rule for reading off the statements of relevance and irrelevance relations for two (sets of) variables, possibly given a third variable (or set of variables). This general rule is known as the d-separation criterion and is due to Pearl (1988).

In Chapter 2 we show that for any joint probability distribution that factorizes according to a DAG, \mathcal{G} , independence statements involving variables X_u and X_v are equivalent to similar statements about d-separation of nodes u and v in \mathcal{G} .⁵

Thus, the d-separation criterion may be used to answer queries of the kind “are X and Y independent given Z ” (in a probabilistic sense) or, more generally, queries of the kind “is information about X irrelevant for our belief about the state of Y given information about Z ”, where X and Y are individual variables and Z is either the empty set of variables or an individual variable.

The d-separation criterion may also be used with sets of variables, although this may be cumbersome. On the other hand, answering such queries is very efficient using the directed global Markov property (Lauritzen, Dawid, Larsen & Leimer 1990), which is a criterion that is equivalent to the d-separation criterion.

As statements of (conditional) d-separation/d-connection and (conditional) dependence/independence play a key role in probabilistic networks, some shorthand notation is convenient. We shall use the standard notations shown in Table 1.5 on the next page.⁶

Notice that statements of (conditional) d-separation or d-connection are always with respect to some DAG. When the DAG is obvious from the context, we shall often avoid the subscript of the d-separation symbol (\perp). Similarly,

⁵See Chapter 2 for definitions of probabilistic independence and structural factorization of DAGs.

⁶A precise semantics of the symbol “ \perp ” shall be given in Chapter 2.

Notation	Meaning
$u \perp_{\mathcal{G}} v$	$u \in V$ and $v \in V$ are d-separated in graph $\mathcal{G} = (V, E)$.
$U \perp_{\mathcal{G}} V$	Each $u \in U$ and each $v \in V$ are d-separated in graph \mathcal{G} . We simply say that U and V are d-separated in \mathcal{G} .
$U \perp V$	U and V are d-separated (graph understood from context).
$U \perp V W$	U and V are d-separated given (hard) evidence on W .
$U \not\perp V W$	U and V are d-connected given (hard) evidence on W .
$X \perp\!\!\!\perp_P Y$	X and Y are (marginally) independent with respect to probability distribution P .
$X \perp\!\!\!\perp Y$	X and Y are (marginally) independent (probability distribution understood from context).
$X \perp\!\!\!\perp Y Z$	X and Y are conditionally independent given (hard) evidence on Z .
$X \not\perp\!\!\!\perp Y Z$	X and Y are conditionally dependent given (hard) evidence on Z .

Table 1.5: Standard notations for (i) statements of (conditional) d-separation/d-connection between sets of nodes U and V , possibly given a third set W , and (ii) (conditional) dependence/independence between (sets of) variables X and Y possibly given a third set Z .

when the probability distribution is obvious from the context, we shall often avoid the subscript of the independence symbol ($\perp\!\!\!\perp$).

Example 5 (Burglary or Earthquake, page 10) Some of the d-separation and d-connection properties observed in the “Burglary or Earthquake” example are:

- (1) Burglary \perp Earthquake
- (2) Burglary $\not\perp$ Earthquake | Alarm
- (3) Burglary \perp RadioNews
- (4) Burglary \perp WatsonCalls | Alarm ■

Also, notice that d-separation and d-connection (and independence and dependence, respectively) depends on the information available; i.e., it depends on what you know (and do not know). Also, note that, d-separation and d-connection (and independence and dependence) relations are always symmetric (i.e., $u \perp v \equiv v \perp u$ and $X_u \perp\!\!\!\perp X_v \equiv X_v \perp\!\!\!\perp X_u$).

1.6.1 d-Separation Criterion

Propositions 1–3 can be summarized into a rule known as *d-separation* (Pearl 1988):

Proposition 4 (d-Separation) *A path $\pi = \langle u, \dots, v \rangle$ in a DAG, $\mathcal{G} = (V, E)$, is said to be blocked by $S \subseteq V$ if π contains a node w such that either*

- $w \in S$ and the links of π do not meet head-to-head at w , or
- $w \notin S$, $\text{de}(w) \cap S = \emptyset$, and the links of π meet head-to-head at w .

For three (not necessarily disjoint) subsets A, B, S of V , A and B are said to be d-separated if all paths between A and B are blocked by S .

We can make Proposition 4 operational through a focus on nodes or through a focus on connections. Let $\mathcal{G} = (V, E)$ be a DAG of a causal network and let $H_\varepsilon \subseteq S_\varepsilon \subseteq V$ be subsets of nodes representing, respectively, variables with hard evidence and variables with soft evidence on them.⁷ Assume that we wish to answer the question: “Are nodes v_1 and v_n d-separated in \mathcal{G} under evidence scenario S_ε ?”

Now, using a nodes approach to d-separation, we can answer the question as follows:

If for any path $\langle v_1, \dots, v_n \rangle$ between v_1 and v_n and for each $i = 2, \dots, n-1$ either

- $v_i \in H_\varepsilon$ and the connection $v_{i-1} \sim v_i \sim v_{i+1}$ is serial or diverging,

or

- $(\{v_i\} \cup \text{de}(v_i)) \cap S_\varepsilon = \emptyset$ and $v_{i-1} \rightarrow v_i \leftarrow v_{i+1}$,

then v_1 and v_n are d-separated given S_ε ; otherwise, they are d-connected given S_ε .

Often, however, people find it more intuitive to think in terms of flow of information, in which case a connections (or flow-of-information) approach to d-separation might be more natural:

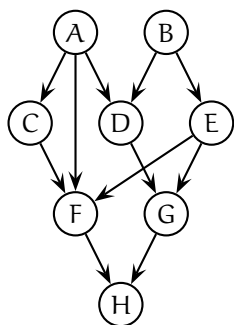
If for some path $\langle v_1, \dots, v_n \rangle$ between v_1 and v_n and for each $i = 2, \dots, n-1$ the connection $v_{i-1} \sim v_i \sim v_{i+1}$ allows flow of information from v_{i-1} to v_{i+1} , then v_1 and v_n are d-connected; otherwise, they are d-separated.

Note that, when using a flow-of-information approach, one should be careful not to use a reasoning scheme like “Since information can flow from u to v and information can flow from v to w , then information can flow from u to w ”, as this kind of reasoning is not supported by the above approach. The problem is that links might meet head-to-head at v , disallowing flow of information between the parents of v , unless evidence is available for v or one of v ’s descendants. So, each pair of consecutive connections investigated must overlap by two nodes.

⁷Recall the definition of evidence on page 9.

Example 6 (d-Separation) Consider the problem of figuring out whether variables C and G are d-separated in the DAG in Figure 1.15; that is, are C and G independent when no evidence about any of the other variables is available? Using the flow-of-information approach, we first find that there is a diverging connection $C \leftarrow A \rightarrow D$ allowing flow of information from C to D via A . Second, there is a serial connection $A \rightarrow D \rightarrow G$ allowing flow of information from A to G via D . So, information can thus be transmitted from C to G via A and D , meaning that C and G are not d-separated (i.e., they are d-connected).

C and E , on the other hand, are d-separated, since each path from C to E contains a converging connection, and since no evidence is available, each such connection will not allow flow of information. Given evidence on one or more of the variables in the set $\{D, F, G, H\}$, C and E will, however, become d-connected. For example, evidence on H will allow the converging connection $D \rightarrow G \leftarrow E$ to transmit information from D to E via G , as H is a child of G . Then information may be transmitted from C to E via the diverging connection $C \leftarrow A \rightarrow D$ and the converging connection $D \rightarrow G \leftarrow E$. ■



- (1) C and G are d-connected
- (2) C and E are d-separated
- (3) C and E are d-connected given evidence on G
- (4) A and G are d-separated given evidence on D and E
- (5) A and G are d-connected given evidence on D

Figure 1.15: Sample DAG with a few sample dependence (d-connected) and independence (d-separated) statements.

1.6.2 Directed Global Markov Criterion

The directed global Markov property (Lauritzen et al. 1990) provides a criterion that is equivalent to that of the d-separation criterion, but which in some cases may prove more efficient in terms of requiring less inspections of possible paths between the involved nodes of the graphs.

Proposition 5 (Directed Global Markov Property) *Let $\mathcal{G} = (V, E)$ be a DAG and A, B, S be disjoint sets of V . Then each pair of nodes $(\alpha \in A, \beta \in B)$ are d-separated by S whenever each path from α to β is blocked by nodes in S*

in the graph

$$(\mathcal{G}_{\text{An}(A \cup B \cup S)})^m.$$

Although the criterion might look somewhat complicated at a first glance, it is actually quite easy to apply. The criterion says that $A \perp_{\mathcal{G}} B | S$ if all paths from A to B passes at least one node in S in the moral graph of the sub-DAG induced by the ancestral set of $A \cup B \cup S$.

Example 7 (Directed Global Markov Property) Consider the DAG, $\mathcal{G} = (V, E)$, in Figure 1.16(a), and let the subsets $A, B, S \subseteq V$ be given as shown in Figure 1.16(b), where the set, S , of evidence variables is indicated by the filled circles. That is, we ask if $A \perp_{\mathcal{G}} B | S$. Using Proposition 5 on the preceding page, we first remove each node not belonging to the ancestral set $\text{An}(A \cup B \cup S)$. This gives us the DAG in Figure 1.16(c). Second, we moralize the resulting sub-DAG, which gives us the undirected graph in Figure 1.16(d). Then, to answer the query, we consult this graph to see if there is a path from a node in A to a node in B that does not contain a node in S . Since this is indeed the case, we conclude that $A \not\perp_{\mathcal{G}} B | S$. ■

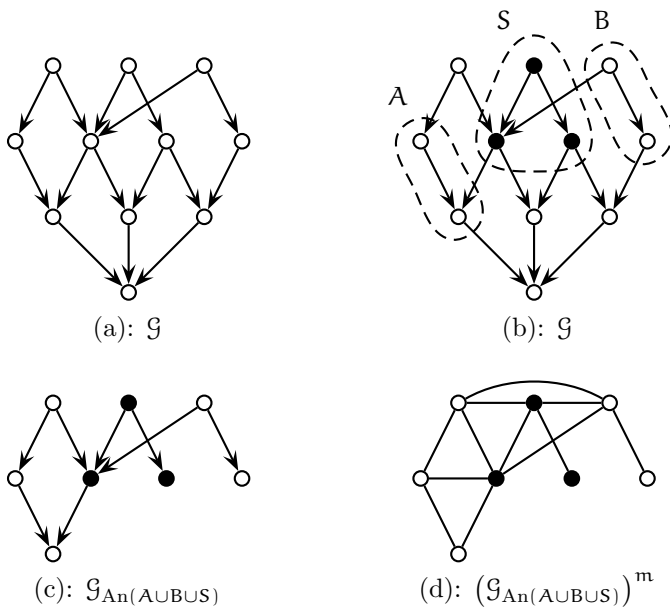


Figure 1.16: (a) A DAG, \mathcal{G} . (b) \mathcal{G} with subsets A, B , and S indicated, where the variables in S are assumed to be observed. (c) The subgraph induced by the ancestral set of $A \cup B \cup S$. (d) The moral graph of the DAG of part (c).

1.7 Summary

This chapter has shown how acyclic, directed graphs provide a powerful language for expressing and reasoning about causal relations among variables. In particular, we saw how such graphical models provide an inherent mechanism for realizing deductive (causal), abductive (diagnostic), as well as intercausal (explaining away) reasoning. As mentioned above, the ability to perform intercausal reasoning is unique for graphical models, and is one of the key differences between automatic reasoning systems based on probabilistic networks and systems based on, for example, production rules.

Part of the explanation for the qualitative reasoning power of graphical models lies in the fact that a DAG is a very compact representation of dependence and independence statements among a set of variables. As we saw in Section 1.4, even models containing only a few variables can contain numerous statements of dependence and independence.

Despite their compactness, DAGs are also very intuitive maps of causal and correlational interactions, and thus provide a powerful language for formulating, communicating, and discussing qualitative (causal) interaction models both in problem domains where causal or correlational mechanisms are (at least partly) known and in domains where such mechanisms are unknown but can be revealed through learning of model structure from data.

Having discussed the foundation of the qualitative aspect of probabilistic networks, in Chapter 2 we shall present the calculus of uncertainty that comprises the quantitative aspect of probabilistic networks.

Chapter 2

Probabilities

As mentioned in Chapter 1, probabilistic networks have a qualitative aspect and a corresponding quantitative aspect, where the qualitative aspect is given by a graphical structure in the form of an acyclic, directed graph (DAG) that represents the (conditional) dependence and independence properties of a joint probability distribution defined over a set of variables that are indexed by the nodes of the DAG.

The fact that the structure of a probabilistic network can be characterized as a DAG derives from basic axioms of probability calculus leading to recursive factorization of a joint probability distribution into a product of lower-dimensional conditional probability distributions. First, any joint probability distribution can be decomposed (or factorized) into a product of conditional distributions of different dimensionality, where the dimensionality of the largest distribution is identical to the dimensionality of the joint distribution. Second, statements of local conditional independences manifest themselves as reductions of dimensionalities of some of the conditional probability distributions. Collectively, these two facts give rise to a DAG structure.

In fact, a joint probability distribution, P , can be decomposed recursively in this fashion if and only if there is a DAG that correctly represents the (conditional) dependence and independence properties of P . This means that a set of conditional probability distributions specified according to a DAG, $\mathcal{G} = (V, E)$, (i.e., a distribution $P(A|pa(A))$ for each $A \in V$) define a joint probability distribution.

Therefore, a probabilistic network model can be specified either through direct specification of a joint probability distribution, or through a DAG (typically) expressing cause-effect relations and a set of corresponding conditional probability distributions. Obviously, a model is almost always specified in the latter fashion.

This chapter presents some basic axioms of probability calculus from which the famous Bayes' rule follows as well as the chain rule for decomposing a joint probability distribution into a product of conditional distributions. We shall also

present the fundamental operations needed to perform inference in probabilistic networks.

Although probabilistic networks can define factorizations of probability distributions over discrete variables, probability density functions over continuous variables, and mixture distributions, for simplicity of exposition, we shall restrict the presentation in this chapter to the pure discrete case. The results, however, extend naturally to the continuous and the mixed cases.

In Section 2.1 we present some basic concepts and axioms of Bayesian probability theory, and in Section 2.2 we introduce probability distributions over variables and show how these can be represented graphically. In Section 2.3 we discuss the notion of (probability) potentials, which can be considered generalizations of probability distributions that is useful when making inference in probabilistic networks, and we present the basic operations for manipulation of potentials (i.e., the fundamental arithmetic operations needed to make inference in the networks). In Section 2.4 we present and discuss Bayes' rule of probability calculus, and in Section 2.5 we briefly mention the concept of Bayes' factors, which can be useful for comparing the relative support for competing hypotheses. In Section 2.6 we define the notion of probabilistic independence and makes the connection to the notion of d-separation in DAGs. Using the fundamental rule of probability calculus (from which Bayes' rule follows) and the connection between probabilistic independence and d-separation, we show in Section 2.7 how a joint probability distribution, P , can be factorized into a product of lower-dimensional (conditional) distributions when P respects the independence properties encoded in a DAG. Finally, in Section 2.8 we summarize the results presented in this chapter.

2.1 Basics

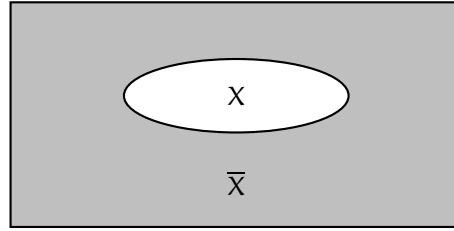
The uncertainty calculus used in probabilistic networks is based on probability theory. Thus, the notions of probability and, in particular, conditional probability plays a key role. In this section we shall introduce some basic concepts and axioms of Bayesian probability theory. These include the notions of probability, events, and three basic axioms of probability theory.

2.1.1 Definition of Probability

Let us start out by defining informally what we mean by “probability”. Apart from introducing the notion of probability, this will also provide some intuition behind the three basic axioms presented in Section 2.1.4.

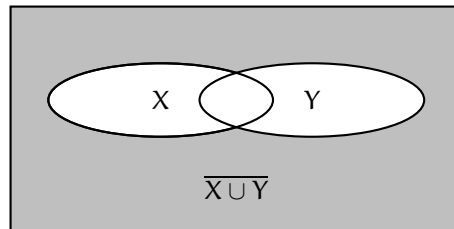
Consider a (discrete) universe, \mathcal{U} , of elements, and let $X \subseteq \mathcal{U}$. Denote by $\bar{X} = \mathcal{U} \setminus X$ the complement of X . Figure 2.1 on the facing page illustrates \mathcal{U} , where we imagine the area that X covers is proportional to the number of elements that it contains.

The chance that an element sampled randomly from \mathcal{U} belongs to X defines the *probability* that the element belongs to X , and is denoted by $P(X)$. Note

Figure 2.1: Universe of elements, $U = X \cup \bar{X}$.

that $P(X)$ can informally be regarded as the relative area occupied by X in U . That is, $P(X)$ is a number between 0 and 1.

Suppose now that $U = X \cup Y \cup \overline{X \cup Y}$; see Figure 2.2. The probability that

Figure 2.2: The probability that a random element from U belongs to either X or Y is defined as $P(X \cup Y) = P(X) + P(Y) - P(X \cap Y)$.

a random element from U belongs to $X \cup Y$ is defined as

$$P(X \cup Y) = P(X) + P(Y) - P(X \cap Y).$$

Again, we can interpret $P(X \cup Y)$ as the relative area covered jointly by X and Y . So, if X and Y are disjoint (i.e., $X \cap Y = \emptyset$), then $P(X \cup Y) = P(X) + P(Y)$. The conjunctive form $P(X \cap Y)$ (or $P(X \wedge Y)$) is often written as $P(X, Y)$.

Consider Figure 2.3 and assume that we know that a random element from U belongs to Y . The probability that it also belongs to X is then calculated as the ratio $P(X \cap Y)/P(Y)$. Again, to acknowledge this definition, it may help to consider $P(X \cap Y)$ and $P(Y)$ as relative areas of U . We shall use the notation $P(X|Y)$ to denote this ratio, where “|” reads “given that we know” or simply “given”. Thus, we have

$$P(X|Y) = \frac{P(X \cap Y)}{P(Y)} = \frac{P(X, Y)}{P(Y)},$$

where $P(X|Y)$ reads “the conditional probability of X given Y ”.

2.1.2 Events

The language of probabilities consists of statements (propositions) about probabilities of events. As an example, consider the event, a , that a sample from

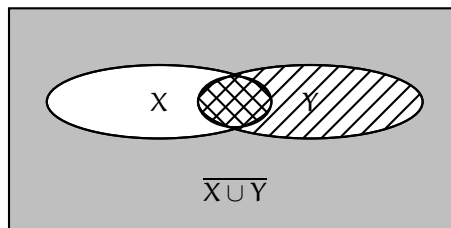


Figure 2.3: The conditional probability that a random element from U belongs to X given that we know that it belongs to Y amounts to the ratio between $P(X \cap Y)$ and $P(Y)$.

a universe $U = X \cup \bar{X}$ happens to belong to X . The probability of event a is denoted $P(a)$. In general, an *event* can be considered as an outcome of an experiment (e.g., a coin flip), a particular observation of a value of a variable (or set of variables), etc. As a probabilistic network define a probability distribution over a set of variables, V , in our context an event is a configuration, $x \in \text{dom}(X)$, (i.e., a vector of values) of a subset of variables $X \subseteq V$.

Example 8 (Burglary or Earthquake, page 10) Assume that our state of knowledge include the facts that $W = \text{yes}$ and $R = \text{yes}$. This evidence is given by the event $\varepsilon = (W = \text{yes}, R = \text{yes})$, and the probability $P(\varepsilon)$ denotes the probability of this particular piece of evidence, namely that both $W = \text{yes}$ and $R = \text{yes}$ are known. ■

2.1.3 Conditional Probability

The basic concept in the Bayesian treatment of uncertainty is that of *conditional probability*: Given event b , the conditional probability of event a is x , written as

$$P(a|b) = x.$$

This means that if event b is true and everything else known is irrelevant for event a , then the probability of event a is x .

Example 9 (Burglary or Earthquake, page 10) Assume that Mr. Holmes' alarm sounds in eight of every ten cases when there is an earthquake but no burglary. This fact would then be expressed as the conditional probability $P(A = \text{yes} | B = \text{no}, E = \text{yes}) = 0.8$. ■

2.1.4 Axioms

The following three axioms provide the basis for Bayesian probability calculus, and summarizes the considerations of Section 2.1.1.

Axiom 1 For any event, \mathbf{a} , $0 \leq P(\mathbf{a}) \leq 1$, with $P(\mathbf{a}) = 1$ if and only if \mathbf{a} occurs with certainty.

Axiom 2 For any two mutually exclusive events \mathbf{a} and \mathbf{b} the probability that either \mathbf{a} or \mathbf{b} occur is

$$P(\mathbf{a} \text{ or } \mathbf{b}) \equiv P(\mathbf{a} \vee \mathbf{b}) = P(\mathbf{a}) + P(\mathbf{b}).$$

In general, if events $\mathbf{a}_1, \dots, \mathbf{a}_n$ are pairwise exclusive, then

$$P\left(\bigcup_i^n \mathbf{a}_i\right) = P(\mathbf{a}_1) + \dots + P(\mathbf{a}_n) = \sum_i^n P(\mathbf{a}_i).$$

Axiom 3 For any two events \mathbf{a} and \mathbf{b} the probability that both \mathbf{a} and \mathbf{b} occur is

$$P(\mathbf{a} \text{ and } \mathbf{b}) \equiv P(\mathbf{a} \wedge \mathbf{b}) \equiv P(\mathbf{a}, \mathbf{b}) = P(\mathbf{b}|\mathbf{a})P(\mathbf{a}) = P(\mathbf{a}|\mathbf{b})P(\mathbf{b}).$$

$P(\mathbf{a}, \mathbf{b})$ is called the joint probability of the events \mathbf{a} and \mathbf{b} .

Axiom 1 simply says that a probability is a non-negative real number less than or equal to 1, and that it equals 1 if and only if the associated event has happened for sure.

Axiom 2 says that if two events cannot co-occur, then the probability that either one of them occurs equals the sum of the probabilities of their individual occurrences.

Axiom 3 is sometimes referred to as the *fundamental rule* of probability calculus. The axiom says that the probability of the co-occurrence of two events, \mathbf{a} and \mathbf{b} can be computed as the product of the probability of event \mathbf{a} (\mathbf{b}) occurring conditional on the fact that event \mathbf{b} (\mathbf{a}) has already occurred and the probability of event \mathbf{b} (\mathbf{a}) occurring.

Example 10 Consider the events “The cast of the die gives a 1” and “The cast of the die gives a 6”. Obviously, these events are mutually exclusive, and the probability that one of them is true equals the sum of the probabilities that the first event is true and that the second event is true. Thus, intuitively, Axiom 2 makes sense. ■

Note that if a set of events, $\{\mathbf{a}_1, \dots, \mathbf{a}_n\}$, is an exhaustive set of outcomes of some experiment (e.g., cast of a die), then $\sum_i P(\mathbf{a}_i) = 1$.¹

Example 11 (Balls in An Urn) Assume we have an urn with 2 red, 3 green, and 5 blue balls. The probabilities of picking a red, a green, or a blue ball are

$$P(\text{red}) = \frac{2}{10} = 0.2, \quad P(\text{green}) = \frac{3}{10} = 0.3, \quad P(\text{blue}) = \frac{5}{10} = 0.5.$$

¹See also the Rule of Total Probability on page 31.

By Axiom 2 we get the probability of picking either a green or a blue ball:

$$P(\text{green or blue}) = P(\text{green}) + P(\text{blue}) = 0.8.$$

Similarly, the probability of picking either a red, a green, or a blue ball is 1. Without replacement, the probabilities of the different possible colors of the second ball depends on the color of the first ball. If we first pick a red ball (and keep it), then the probabilities of picking a red, a green, or a blue ball as the next one are, respectively,

$$P(\text{2nd-is-red} | \text{1st-was-red}) = \frac{2-1}{10-1} = \frac{1}{9},$$

$$P(\text{2nd-is-green} | \text{1st-was-red}) = \frac{3}{10-1} = \frac{3}{9},$$

$$P(\text{2nd-is-blue} | \text{1st-was-red}) = \frac{5}{10-1} = \frac{5}{9}.$$

By Axiom 3 we get the probability that the 1st ball is red and the 2nd is red:

$$\begin{aligned} P(\text{1st-was-red, 2nd-is-red}) &= P(\text{2nd-is-red} | \text{1st-was-red})P(\text{1st-was-red}) \\ &= \frac{1}{9} \cdot \frac{1}{5} = \frac{1}{45} \end{aligned}$$

Similarly, the probabilities that the 1st ball is red and the 2nd is green/blue are

$$\begin{aligned} P(\text{1st-was-red, 2nd-is-green}) &= P(\text{2nd-is-green} | \text{1st-was-red})P(\text{1st-was-red}) \\ &= \frac{1}{3} \cdot \frac{1}{5} = \frac{1}{15}, \end{aligned}$$

$$\begin{aligned} P(\text{1st-was-red, 2nd-is-blue}) &= P(\text{2nd-is-blue} | \text{1st-was-red})P(\text{1st-was-red}) \\ &= \frac{5}{9} \cdot \frac{1}{5} = \frac{1}{9}, \end{aligned}$$

respectively. ■

2.2 Probability Distributions for Variables

Probabilistic networks are defined over a (finite) set of variables, each of which represents a finite set of exhaustive and mutually exclusive states (or events); see Section 1.2 on page 6. Thus, (conditional) probability distributions for variables (i.e., over exhaustive sets of mutually exclusive events) play a central role in probabilistic networks.

If X is a (random) variable with domain $\text{dom}(X) = (x_1, \dots, x_{|X|})$, then $P(X)$ denotes a probability distribution (i.e., a vector of probabilities summing to 1), where

$$P(X) = (P(X = x_1), \dots, P(X = x_{|X|})).$$

If no confusion is possible, we shall often use $P(x)$ as short for $P(X = x)$, etc.

If the probability distribution for a variable Y is given conditional on a variable (or set of variables) X , then we shall use the notation $P(Y|X)$. That is, for each possible value (state), $x \in \text{dom}(X)$, we have a probability distribution $P(Y|X = x)$; again, if no confusion is possible, we shall often write $P(Y|x)$.

Example 12 (Balls in An Urn, page 29) Let X_1 represent the following exhaustive set of mutually exclusive events:

$$\text{dom}(X_1) = \{\text{"1st ball is red"}, \text{"1st ball is green"}, \text{"1st ball is blue"}\}.$$

If we define X_1 to denote the random variable “The color of the 1st ball drawn from the urn”, then we may define $\text{dom}(X_1) = \{\text{red, green, blue}\}$. Similarly, if we define X_2 to denote the random variable “The color of the 2nd ball drawn from the urn”, then $\text{dom}(X_2) = \text{dom}(X_1)$. From Example 11 on page 29 we get

$$P(X_1) = \left(\frac{2}{10}, \frac{3}{10}, \frac{5}{10} \right),$$

$$P(X_2|X_1 = \text{red}) = \left(\frac{1}{9}, \frac{3}{9}, \frac{5}{9} \right).$$

$P(X_2|X_1)$ can be described in terms of a table of three (conditional) distributions:

	$X_1 = \text{red}$	$X_1 = \text{green}$	$X_1 = \text{blue}$
$X_2 = \text{red}$	$\frac{1}{9}$	$\frac{2}{9}$	$\frac{2}{9}$
$X_2 = \text{green}$	$\frac{3}{9}$	$\frac{2}{9}$	$\frac{3}{9}$
$X_2 = \text{blue}$	$\frac{5}{9}$	$\frac{5}{9}$	$\frac{4}{9}$

Note that the probabilities in each column sum to 1. ■

2.2.1 Rule of Total Probability

Let $P(X, Y)$ be a joint probability distribution for two variables X and Y with $\text{dom}(X) = \{x_1, \dots, x_m\}$ and $\text{dom}(Y) = \{y_1, \dots, y_n\}$. Using the fact that $\text{dom}(X)$ and $\text{dom}(Y)$ are exhaustive sets of mutually exclusive states of X and Y , respectively, Axiom 2 on page 29 gives us the *rule of total probability*:

$$\forall i : P(x_i) = P(x_i, y_1) + \dots + P(x_i, y_n) = \sum_{j=1}^n P(x_i, y_j). \quad (2.1)$$

Using (2.1), we can calculate $P(X)$ from $P(X, Y)$:

$$P(X) = \left(\sum_{j=1}^n P(x_1, y_j), \dots, \sum_{j=1}^n P(x_m, y_j) \right).$$

In a more compact notation, we may write

$$P(X) = \sum_{j=1}^n P(X, y_j),$$

or even shorter as

$$P(X) = \sum_Y P(X, Y), \quad (2.2)$$

denoting the fact that we sum over all indices of Y . We shall henceforth refer to the operation in (2.2) as *marginalization* or *projection*.² Also, we sometimes refer to this operation as *marginalizing out* Y of $P(X, Y)$ or *eliminating* Y from $P(X, Y)$.

Example 13 (Balls in An Urn, page 29) Using Axiom 3 on page 29 for each combination of states of X_1 and X_2 of Example 12 on the preceding page, we can compute

$$\begin{aligned} P(X_1 = \text{red}, X_2 = \text{red}) &= P(X_1 = \text{red})P(X_2 = \text{red} | X_1 = \text{red}) \\ &= \frac{2}{10} \cdot \frac{1}{9} \\ &= \frac{1}{45}, \end{aligned}$$

etc. That is, we get $P(X_1, X_2)$ by combining $P(X_1)$ and $P(X_2 | X_1)$:

	$X_1 = \text{red}$	$X_1 = \text{green}$	$X_1 = \text{blue}$
$X_2 = \text{red}$	$\frac{1}{45}$	$\frac{1}{15}$	$\frac{1}{9}$
$X_2 = \text{green}$	$\frac{1}{15}$	$\frac{1}{15}$	$\frac{1}{6}$
$X_2 = \text{blue}$	$\frac{1}{9}$	$\frac{1}{6}$	$\frac{2}{9}$

(Note that the numbers in the table sum to 1.) Now, through marginalization we get

$$\begin{aligned} P(X_2) &= P(X_1 = \text{red}, X_2) + P(X_1 = \text{green}, X_2) + P(X_1 = \text{blue}, X_2) \\ &= \begin{pmatrix} \frac{1}{45} \\ \frac{1}{15} \\ \frac{1}{9} \end{pmatrix} + \begin{pmatrix} \frac{1}{15} \\ \frac{1}{15} \\ \frac{1}{6} \end{pmatrix} + \begin{pmatrix} \frac{1}{9} \\ \frac{1}{6} \\ \frac{2}{9} \end{pmatrix} = \begin{pmatrix} \frac{1}{5} \\ \frac{3}{10} \\ \frac{1}{2} \end{pmatrix}. \end{aligned}$$

That is, the probability of getting a red, a green, and a blue ball in the second draw are, respectively, 0.2, 0.3, and 0.5, given that we know nothing about the

²See Section 2.3.3 on page 36 for more on marginalization.

color of the first ball. That is, $P(X_2) = P(X_1) = (0.2, 0.3, 0.5)$, whereas, for example, $P(X_2|X_1 = \text{red}) = (0.1111, 0.3333, 0.5556)$; i.e., once the color of the first ball is known, our belief about the color of the second changes. ■

2.2.2 Graphical Representation

The conditional probability distributions of probabilistic networks are of the form

$$P(X|Y),$$

where X is a single variable and Y is a (possibly empty) set of variables. X and Y are sometimes called the *head* and the *tail*, respectively, of $P(X|Y)$. If $Y = \emptyset$ (i.e., the empty set), $P(X|Y)$ is often called a *marginal probability distribution* and is then written as $P(X)$. This relation between X and $Y = \{Y_1, \dots, Y_n\}$ can be represented graphically as the DAG illustrated in Figure 2.4, where the child node is labelled “ X ” and the parent nodes are labelled “ Y_1 ”, “ Y_2 ”, etc.

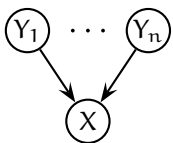


Figure 2.4: Graphical representation of $P(X|Y_1, \dots, Y_n)$.

Example 14 (Burglary or Earthquake, page 10) Consider the variables B (Burglary), E (Earthquake), and A (Alarm), where B and E are possible causes of A . A natural way of specifying the probabilistic relations between these three variables, would be through a conditional probability distribution for A given B and E . Thus, for each combination of outcomes (states) of B and E , we need to specify a probability distribution over the states of A :

	B = no	B = yes		
	E = no	E = yes	E = no	E = yes
A = no	0.99	0.1	0.1	0.01
A = yes	0.01	0.9	0.9	0.99

This conditional probability table (CPT) expresses the probability (whether obtained as an objective frequency or a subjective belief) of an alarm if either a burglary or an earthquake has taken place (but not both) to be 0.9, etc.

The CPT is reflected by the converging connection in Figure 1.6 on page 11. In general, any conditional probability distribution $P(X_v|X_{u_1}, \dots, X_{u_n})$ will give rise to $n(n-1)/2$ converging connections, as each pair of parent nodes, (u_i, u_j) , introduces a converging connection along with the child node, v . ■

2.3 Probability Potentials

In working with probabilistic networks the notion of “potentials” often appears to be convenient. Formally, a *probability potential* is a non-negative function defined over the product space over the domains of a set of variables. We shall use Greek letters (ϕ , ψ , etc.) to denote potentials, and sometimes use subscripts to identify their domain (e.g., ϕ_X denotes a potential defined on $\text{dom}(X)$) or to indicate that a potential ϕ_X is a marginal potential of ϕ .³

2.3.1 Normalization

A (probability) potential, ϕ_X defined on $\text{dom}(X)$, is turned into a probability distribution, $P(X)$, through the operation known as *normalization*. We shall use the notation $\eta(\phi_X)$ to denote normalization of ϕ_X , where $\eta(\phi_X)$ is defined as

$$\eta(\phi_X) \triangleq \frac{\phi_X}{\sum_X \phi_X}. \quad (2.3)$$

Hence, $P(X) = \eta(\phi_X)$. The conditional probability distribution $P(X|Y)$ can be obtained from the joint distribution $P(X, Y)$ through *conditional normalization* with respect to X :

$$\eta_X(P(X, Y)) \triangleq \frac{P(X, Y)}{\sum_Y P(X, Y)} = \frac{P(X, Y)}{P(X)} = P(Y|X).$$

In general,

$$\eta_{X'}(P(X)) \triangleq \frac{P(X)}{\sum_{X \setminus X'} P(X)} = \frac{P(X)}{P(X')} = P(X \setminus X' | X'), \quad (2.4)$$

where X' is a subset of the set of variables X . In particular,

$$\eta(P(X)) = \eta_\emptyset(P(X)) = P(X),$$

whenever $P(X)$ is a probability distribution over X . This also holds true for conditional distributions:

$$\eta_Y(P(X|Y)) = P(X|Y),$$

since

$$\sum_X P(X|Y) = \mathbf{1}_Y, \quad (2.5)$$

where $\mathbf{1}_Y$ denotes a vector of 1s over $\text{dom}(Y)$. A uniform potential, e.g. $\mathbf{1}_Y$, is called a *vacuous potential*. Intuitively, a vacuous potential can be thought of as a non-informative (or superfluous) potential.

We shall be using the notion of potentials extensively in Chapter 4, but for now we shall just give a couple of simple examples illustrating the usefulness of this notion.

³Note that the two interpretations are consistent. See Section 2.3.3 on page 36 for details on marginalization.

Example 15 Let $P(A, B) = P(A)P(B|A)$ be a factorization of the joint distribution for the Boolean variables A and B , and assume that $P(A)$ and $P(B|A)$ are given by the potentials ϕ and ψ , respectively, where

$$\phi = (2, 1) \quad \text{and} \quad \psi = \begin{array}{c|cc} & A = F & A = T \\ \hline B = F & 5 & 7 \\ B = T & 3 & 1 \end{array} .$$

Then

$$P(A) = \eta(\phi) = \left(\frac{2}{3}, \frac{1}{3}\right) \quad \text{and} \quad P(B|A) = \eta_A(\psi) = \begin{array}{c|cc} & A = F & A = T \\ \hline B = F & \frac{5}{8} & \frac{7}{8} \\ B = T & \frac{3}{8} & \frac{1}{8} \end{array} ,$$

and thus $P(A, B) = \eta(\phi) * \eta_A(\psi)$.⁴ Note that, in general, $\eta(\phi) * \eta_A(\psi) \neq \eta(\phi * \psi)$, although in this particular case equality holds, since the vector of normalization constants, $\sum_B \psi = (8, 8)$, in $\eta_A(\psi)$ is uniform. ■

2.3.2 Evidence Potentials

As indicated in Section 1.3 on page 9, evidence functions are actually potentials. To compute the joint posterior distribution resulting from incorporating a set of observations in the form of evidence functions, we simply extend the set of probability function constituents (possibly in the form of potentials) with corresponding evidence potentials, multiply, and normalize the product.

Before any evidence has been taken into account the probability distribution $P(X')$ for a subset $X' \subseteq X$ of variables is referred to as the *prior* probability distribution for X' . The conditional probability distribution $P(X'|\varepsilon)$, where ε denotes evidence, is referred to as the *posterior* probability distribution for X' given ε . Given an evidence potential \mathcal{E}_E on a subset $E \subseteq X$ (expressing ε), the posterior joint distribution is obtained as

$$P(X', \varepsilon) = \sum_{X \setminus X'} P(X) * \mathcal{E}_E,$$

and the posterior conditional distribution is achieved through normalization

$$P(X'|\varepsilon) = \eta(P(X', \varepsilon)).$$

Example 16 (Example 15 continued) Assume that we observe $B = T$, represented by the evidence potential $\mathcal{E}_B = (0, 1)$. Then the posterior joint distribution is given by

$$P(A, B|\varepsilon) = \eta(\phi_A * \phi_B * \mathcal{E}_B).$$

⁴See Section 2.3.3 on the following page for a definition of combination of potentials.

More explicitly, we have

$$P(A, B|\varepsilon) = \eta \left(\begin{array}{c|cc} & A = F & A = T \\ \hline B = F & 0 & 0 \\ B = T & 6 & 1 \end{array} \right),$$

and we get $P(A|\varepsilon) = (\frac{6}{7}, \frac{1}{7})$. ■

2.3.3 Potential Calculus

To perform inference in probabilistic networks we only need a few simple operations, namely multiplication (combination), division, addition, and marginalization (projection). These are all defined very straightforwardly as follows.

Let ϕ and ψ be potentials defined on $\text{dom}(X)$ and $\text{dom}(Y)$, respectively, and let $z \in \text{dom}(X \cup Y)$ be an arbitrary element (configuration).

We define $\phi * \psi$ as

$$(\phi * \psi)(z) \triangleq \phi(z_X)\psi(z_Y), \quad (2.6)$$

where z_X and z_Y are projections of z to $\text{dom}(X)$ and $\text{dom}(Y)$, respectively.⁵ Addition is defined analogously. We need to take special care to avoid division by zero, but otherwise division is defined in the obvious way:

$$(\phi/\psi)(z) \triangleq \begin{cases} 0 & \text{if } \phi(z_X) = 0 \\ \phi(z_X)/\psi(z_Y) & \text{if } \psi(z_Y) \neq 0 \\ \text{undefined} & \text{otherwise.} \end{cases} \quad (2.7)$$

As we shall see later, for all relevant operations involved in inference in probabilistic networks, $\phi(z_X) = 0$ implies $\psi(z_Y) = 0$ upon division of ϕ by ψ , and thus, defining $0/0 = 0$, the division operator is always defined.

Let $X' \subseteq X$ and let ϕ_X be a potential defined on $\text{dom}(X)$. Then $\phi_{X'} = \sum_{X \setminus X'} \phi_X$ denotes the marginalization (or projection) of ϕ_X to $\text{dom}(X')$ and is defined as

$$\phi_{X'}(x') \triangleq \sum_{z \in \text{dom}(X \setminus X')} \phi_X(z, x'), \quad (2.8)$$

where z, x' is the element in $\text{dom}(X)$ for which $(z, x')_{X \setminus X'} = z$ and $(z, x')_{X'} = x'$. Figure 2.5 illustrates the process, where all values of ϕ_X corresponding to configurations with $X' = x'$ are added together to yield the value of $\phi_{X'}(x')$.

Example 17 Let $X = \{A, B\}$ and $Y = \{B, C, D\}$, where A, B, C, D are all binary variables with $\text{dom}(A) = \{a_1, a_2\}$, etc. Let ϕ_X and ϕ_Y be potentials defined over $\text{dom}(X)$ and $\text{dom}(Y)$, respectively, where

$$\phi_X = \begin{array}{c|cc} & a_1 & a_2 \\ \hline b_1 & 0.1 & 0.9 \\ b_2 & 0.4 & 0.6 \end{array} \quad \text{and} \quad \phi_Y = \begin{array}{c|cccc} & \begin{matrix} c_1 \\ d_1 \end{matrix} & \begin{matrix} d_2 \\ d_1 \end{matrix} & \begin{matrix} c_2 \\ d_1 \end{matrix} & \begin{matrix} d_2 \\ d_2 \end{matrix} \\ \hline b_1 & 0.11 & 0.14 & 0.06 & 0.09 \\ b_2 & 0.23 & 0.07 & 0.18 & 0.12 \end{array}.$$

⁵As defined in Section 1.2 on page 6.

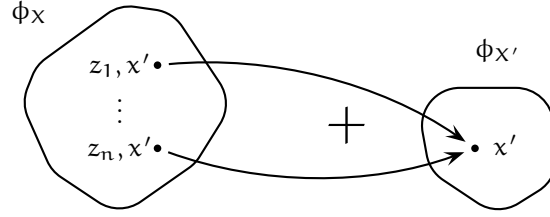


Figure 2.5: Marginalization (or projection) of $\phi_X(z_i, x')$ to $\text{dom}(X')$ for all $z_i \in \text{dom}(X \setminus X')$: $\phi_{X'}(x') = \phi_X(z_1, x') + \dots + \phi_X(z_n, x')$.

From (2.6) we get $\psi = \phi_X * \phi_Y$ to be

$$\psi = \begin{array}{c|cc} & \begin{array}{cc} c_1 & c_2 \end{array} \\ \hline & \begin{array}{cc} d_1 & d_2 \end{array} \\ \hline b_1 & (0.011, 0.099) & (0.014, 0.126) & (0.006, 0.054) & (0.009, 0.081) \\ b_2 & (0.092, 0.138) & (0.028, 0.042) & (0.072, 0.108) & (0.048, 0.072) \end{array},$$

where $(\phi_X * \phi_Y)(a_1, b_1, c_1, d_1) = \phi_X(a_1, b_1)\phi_Y(b_1, c_1, d_1) = 0.1 \cdot 0.11 = 0.011$, $(\phi_X * \phi_Y)(a_2, b_1, c_1, d_1) = \phi_X(a_2, b_1)\phi_Y(b_1, c_1, d_1) = 0.9 \cdot 0.11 = 0.099$, etc.

Now, if $Z = \{A, D\}$, then from (2.8) we get the marginal of ψ with respect to Z to be

$$\begin{aligned} \psi_Z &= \begin{array}{c|cc} & d_1 & d_2 \\ \hline a_1 & 0.011 + 0.092 + 0.006 + 0.072 & 0.014 + 0.028 + 0.009 + 0.048 \\ a_2 & 0.099 + 0.138 + 0.054 + 0.108 & 0.126 + 0.042 + 0.081 + 0.072 \end{array} \\ &= \begin{array}{c|cc} & d_1 & d_2 \\ \hline a_1 & 0.181 & 0.099 \\ a_2 & 0.399 & 0.321 \end{array}, \end{aligned}$$

where, using the above notation, we have $\psi_Z(a_1, d_1) = \psi((b_1, c_1), (a_1, d_1)) + \psi((b_2, c_1), (a_1, d_1)) + \psi((b_1, c_2), (a_1, d_1)) + \psi((b_2, c_2), (a_1, d_1)) = 0.011 + 0.092 + 0.006 + 0.072 = 0.181$, etc. Note that ψ (and hence also ψ_Z) is a probability distribution (i.e., $\sum_x \psi(x) = 1$), since ϕ_X is a conditional probability distribution for A given B and ϕ_Y is a joint probability distribution for $\{B, C, D\}$. ■

Distributive Law

Let ϕ and ψ be potentials defined on $\text{dom}(X) = \{x_1, \dots, x_m\}$ and $\text{dom}(Y) = \{y_1, \dots, y_n\}$, where $X \cap Y = \emptyset$. Using the distributive law, we then get

$$\begin{aligned}
\sum_{X \setminus X'} \sum_{Y \setminus Y'} (\phi * \psi) &= \sum_{x \in \text{dom}(X \setminus X')} \sum_{y \in \text{dom}(Y \setminus Y')} \phi(x)\psi(y) \\
&= \phi(x_1)\psi(y_1) + \dots + \phi(x_1)\psi(y_n) + \dots + \\
&\quad \phi(x_m)\psi(y_1) + \dots + \phi(x_m)\psi(y_n) \\
&= \phi(x_1) [\psi(y_1) + \dots + \psi(y_n)] + \dots + \\
&\quad \phi(x_m) [\psi(y_1) + \dots + \psi(y_n)] \\
&= \sum_{x \in \text{dom}(X \setminus X')} \phi(x) \sum_{y \in \text{dom}(Y \setminus Y')} \psi(y) \\
&= \sum_{X \setminus X'} \phi \sum_{Y \setminus Y'} \psi, \tag{2.9}
\end{aligned}$$

where $X' \subseteq X$, $Y' \subseteq Y$, and $\sum_X \phi \sum_Y \psi$ is short for $\sum_X (\phi * (\sum_Y \psi))$. Thus, if we wish to compute the marginal distribution $(\phi * \psi)_{X' \cup Y'}$ and $X \cap Y = \emptyset$, then using the distributive law may help a lot in terms of reducing the computational complexity.

Example 18 Let ϕ , ψ , and ξ be potentials defined on $\text{dom}(A, B, C)$, $\text{dom}(B, D)$, and $\text{dom}(C, D, E)$, respectively, and let \mathcal{E}_E be an evidence potential defined on $\text{dom}(E)$, where the variables A, \dots, E are all binary. Assume that we wish to compute $P(A|\varepsilon)$, where ε denotes the evidence provided through \mathcal{E}_E . A brute-force approach would be simply to combine all potentials, marginalize out variables B, \dots, E , and normalize:

$$P(A|\varepsilon) = \eta \left(\sum_B \sum_C \sum_D \sum_E (\phi * \psi * \xi * \mathcal{E}_E) \right).$$

Combining potentials ξ and \mathcal{E}_E requires 8 multiplications. Next, combining ψ and $\xi * \mathcal{E}_E$ requires 16 multiplications, and, finally, combining ϕ and $\psi * \xi * \mathcal{E}_E$ requires 32 multiplications. Marginalizing out E , D , C , and B , respectively, require 16, 8, 4, and 2 additions.

Alternatively, we could take advantage of the distributive law to compute the same thing:

$$P(A|\varepsilon) = \eta \left(\sum_B \sum_C \phi \sum_D \psi \sum_E (\xi * \mathcal{E}_E) \right).$$

First, combining ξ and \mathcal{E}_E requires 8 multiplications. Then, marginalizing out E requires 4 additions. Multiplying the resulting potential by ψ requires 8 multiplications, and marginalizing out D requires 4 additions. Next, multiplying

the resulting potential by ϕ requires 8 multiplications, and finally, marginalizing out C and B requires 4 and 2 additions, respectively.

Summing up the number of arithmetic operations used in the two computations we find that the brute-force approach takes 56 multiplications and 30 additions, whereas the one exploiting the distributive law takes only 24 multiplications and 14 additions, less than half of what the brute-force approach requires. (On top of these numbers we should also count the number of operations needed to normalize the final marginal, but that is the same in both cases.)

In terms of the amount of temporary working storage required, the brute-force approach requires 32 units of storage to hold a table of the joint distribution over $\{A, B, C, D, E\}$, whereas the approach that exploits the distributive law only requires 8 units of storage to hold the current potential (i.e., first $\xi * \xi_E$ with domain $\{C, D, E\}$, second $\psi \sum_E (\xi * \xi_E)$ with domain $\{B, C, D\}$, and finally $\phi \sum_D \psi \sum_E (\xi * \xi_E)$ with domain $\{A, B, C\}$).

Note that the ordering (B, C, D, E) is just one out of $4! = 24$ different sequences in which we might marginalize out these four variables, and to each ordering is associated a certain number of arithmetic operations required to compute $P(A|\varepsilon)$. ■

The single most important key to efficient inference in probabilistic networks is the ability to take advantage of the distributive law (i.e., to find optimal (or near optimal) sequences in which the variables are marginalized out). We shall return to this issue in Chapter 4.

2.3.4 Barren Variables

Variables of a graphical model that have no observed descendants, are never observed itself, and for which we do not wish to compute their posterior probabilities are called *barren variables*, as they provide no information relevant for the inference process. In fact, they provide “information” in the form of vacuous potentials (cf. (2.5) on page 34), and may hence be removed from the model.

Example 19 Consider a model $P(X, Y, Z) = P(X)P(Y|X)P(Z|Y)$ over the variables X, Y, and Z. Following the discussion in Section 2.2.2 on page 33, this model can be represented graphically as indicated in Figure 2.6a. Let ξ_Y and ξ_Z be evidence potentials for Y and Z, respectively, but where ξ_Z is always

vacuous. Then the posterior probability distribution for X can be calculated as

$$\begin{aligned}
 P(X|\varepsilon) &= \eta \left(\sum_Y P(Y|X) * \varepsilon_Y \sum_Z P(Z|Y) * \varepsilon_Z \right) \\
 &= \eta \left(\sum_Y P(Y|X) * \varepsilon_Y \sum_Z P(Z|Y) \right) \\
 &= \eta \left(\sum_Y P(Y|X) * \varepsilon_Y * 1_Y \right) \\
 &= \eta \left(\sum_Y P(Y|X) * \varepsilon_Y \right),
 \end{aligned}$$

where $\sum_Z P(Z|Y) = 1_Y$ follows from (2.5) on page 34 and ε denotes the evidence. This means that the term $P(Z|Y)$ can be neglected in the inference process, and the model can be simplified to the one shown in Figure 2.6b, as variable Z is barren. ■

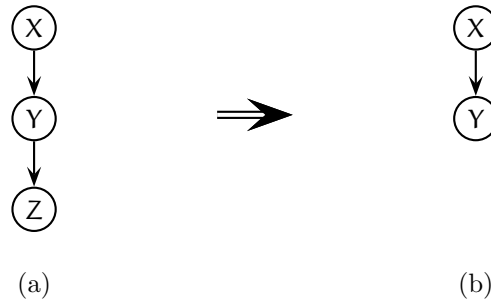


Figure 2.6: (a) Model for $P(X, Y, Z)$. (b) Equivalent model when Z is barren.

We shall return to the issue of barren variables in more detail in Section 4.1.1 on page 92.

2.4 Fundamental Rule and Bayes' Rule

Generalizing Axiom 3 on page 29 to arbitrary (random) variables X and Y we get the *fundamental rule* of probability calculus:

$$P(X, Y) = P(X|Y)P(Y) = P(Y|X)P(X). \quad (2.10)$$

Bayes' rule follows immediately:

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}. \quad (2.11)$$

Using Axiom 3 on page 29 and the rule of total probability, (2.11) can be rewritten like

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X|Y = y_1)P(Y = y_1) + \dots + P(X|Y = y_n)P(Y = y_{||Y||})}$$

That is, the denominator in (2.11) can be derived from the numerator in (2.11). Since, furthermore, the denominator is obviously the same for all states of Y , we often write Bayes' rule as

$$P(Y|X) \propto P(X|Y)P(Y), \tag{2.12}$$

read as “ $P(Y|X)$ is proportional to $P(X|Y)P(Y)$ ”. Note that the proportionality factor $P(X)^{-1}$ is in fact a vector of proportionality constants, one for each state of X , determined in a normalization operation.

Division by zero in (2.11) is not a problem as we define $0/0 = 0$, since for

$$P(x_i) = \sum_j P(x_i|y_j)P(y_j)$$

to be zero at least one of $P(x_i|y_j)$ and $P(y_j)$ must be zero for each j , and if this is the case then both the numerator term, $P(x_i|y_j)P(y_j)$, and the denominator term, $P(x_i)$, of (2.11) will be zero.

Example 20 (Burglary or Earthquake, page 10) Let $P(E) = (0.99, 0.01)$, $P(B) = (0.9, 0.1)$, and let the conditional probability table (CPT) for $P(A|B, E)$ be given as in Example 14 on page 33. Then we can use Bayes' rule to compute, $P(B|A)$, the conditional probability distribution for burglary (B) given alarm (A):

$$P(B|A) \propto \sum_E P(A|B, E)P(B)P(E) = P(A, B).$$

First, we compute the joint distribution for A , B , and E :

$$\begin{aligned}
 P(A, B, E) &= P(A|B, E)P(B)P(E) \\
 &= \begin{array}{c|cc|cc} & \begin{array}{cc} B = \text{no} & B = \text{yes} \end{array} & & & \\ & \begin{array}{cc} E = \text{no} & E = \text{yes} \end{array} & \begin{array}{cc} E = \text{no} & E = \text{yes} \end{array} & & \\ \hline A = \text{no} & 0.88209 & 0.0009 & 0.0099 & 0.00001 \\ A = \text{yes} & 0.00891 & 0.0081 & 0.0891 & 0.00099 \end{array} .
 \end{aligned}$$

Next, we marginalize out E of $P(A, B, E)$ to obtain

$$P(A, B) = \sum_E P(A, B, E) = \begin{array}{c|cc} & B = \text{no} & B = \text{yes} \\ \hline A = \text{no} & 0.88299 & 0.00991 \\ A = \text{yes} & 0.01701 & 0.09009 \end{array} .$$

Finally, we normalize $P(A, B)$ with respect to A , and get

$$P(B|A) = \eta_A(P(A, B)) = \begin{array}{c|cc} & A = \text{no} & A = \text{yes} \\ \hline B = \text{no} & 0.9889 & 0.1588 \\ B = \text{yes} & 0.0111 & 0.8412 \end{array} .$$

■

2.4.1 Interpretation of Bayes' Rule

Since Bayes' rule is so central to inference in Bayesian probability calculus, let us dwell a little on how Bayes' rule can be used and understood. Assume that we have two (possibly, sets of) variables X and Y , a model $P(X, Y)$ given in the factorized form $P(X|Y)P(Y)$, and that we observe $X = x$. We would then typically want to compute $P(Y|x)$.

The prior distribution, $P(Y)$, expresses our initial belief about Y , and the posterior distribution, $P(Y|x)$, expresses our revised belief about Y in light of the observation $X = x$. Bayes' rule tells us how to obtain the posterior distribution by multiplying the prior $P(Y)$ by the ratio $P(x|Y)/P(x)$. Again, since $P(x)$ is a constant for each $y \in \text{dom}(Y)$, we get

$$P(Y|x) \propto P(Y)P(x|Y).$$

The quantity $P(x|Y) \triangleq L(Y|x)$ is called the *likelihood* for Y given x . Hence, we have

$$P(Y|x) \propto P(Y)L(Y|x). \quad (2.13)$$

In general,

$$\text{posterior} \propto \text{prior} \times \text{likelihood}.$$

In a machine learning context, Bayes' rule plays an important role. For example, consider a prior distribution, $P(M)$, for a random variable M , expressing a set of possible models. For any value d of another variable D , expressing data, the quantity $P(d|M)$ — considered as a function of M — is the likelihood function for M given data d . The posterior distribution for M given the data is then

$$P(M|d) \propto P(M)P(d|M),$$

which provides a set of goodness-of-fit measures for models M (i.e., we obtain a conditional probability $P(m|d)$ for each $m \in \text{dom}(M)$).

Arc Reversal

Application of Bayes' rule can also be given a graphical interpretation. Consider, for example, two variables A and B and a model $P(A, B) = P(A)P(B|A)$. Again, following the discussion in Section 2.2.2 on page 33, this model can be represented graphically as indicated in Figure 2.7(a). To apply Bayes' rule on this model we perform the following calculations: (i) $P(A, B) = P(A)P(B|A)$, (ii) $P(B) = \sum_A P(A, B)$, and (iii) $P(A|B) = P(A, B)/P(B)$, whereby we obtain an equivalent model shown in Figure 2.7(b). Thus, one way of interpreting the application of Bayes' rule is through so-called *arc reversal*. As the typical inference task in probabilistic networks can be described as computing $P(X|\epsilon)$, inference in probabilistic networks can be thought of as (repetitive) application of Bayes' rule or, in other words, as a sequence of arc reversals. Shachter (1990) has exploited this view of inference in his arc reversal algorithm for inference in probabilistic networks.

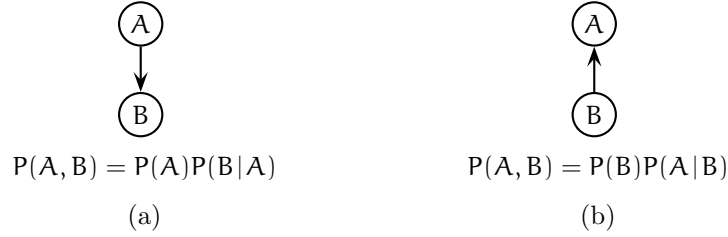


Figure 2.7: Two equivalent models that can be obtained from each other through arc reversal.

Example 21 (Arc reversal) Consider the model in Figure 2.8a, and assume that we wish to calculate the posterior marginal distribution for X given evidence, \mathcal{E}_Z , on Z . Using Shachter's arc reversal procedure we may proceed as follows:

$$\begin{aligned}
 P(X|\varepsilon) &= \eta \left(\sum_Y \sum_Z P(X)P(Y|X)P(Z|Y)\mathcal{E}_Z \right) \\
 &= \eta \left(\sum_Y \sum_Z P(X)P(Y, Z|X)\mathcal{E}_Z \right) \tag{2.14}
 \end{aligned}$$

$$= \eta \left(\sum_Y \sum_Z P(X) \frac{P(Y, Z|X)}{\sum_Y P(Y, Z|X)} \sum_Y P(Y, Z|X)\mathcal{E}_Z \right) \tag{2.15}$$

$$= \eta \left(\sum_Y \sum_Z P(X)P(Y|X, Z)P(Z|X)\mathcal{E}_Z \right) \tag{2.16}$$

$$= \eta \left(\sum_Z P(X)P(Z|X)\mathcal{E}_Z \sum_Y P(Y|X, Z) \right) \tag{2.17}$$

$$= \eta \left(\sum_Z P(X)P(Z|X)\mathcal{E}_Z \right) \tag{2.18}$$

$$= \eta \left(\sum_Z \frac{P(X)P(Z|X)}{\sum_X P(X)P(Z|X)} \sum_X P(X)P(Z|X)\mathcal{E}_Z \right) \tag{2.19}$$

$$= \eta \left(\sum_Z P(X|Z)P(Z)\mathcal{E}_Z \right), \tag{2.20}$$

where we combine $P(Y|X)$ and $P(Z|Y)$ into $P(Y, Z|X)$ (2.14), use Bayes' rule to reverse $Y \rightarrow Z$ (2.15), which induces a new link $X \rightarrow Z$ (2.16), use the distributive law (2.17), eliminate barren variable Y (2.18), and finally use Bayes' rule to reverse $X \rightarrow Z$ (2.19). Now, if \mathcal{E}_Z represent hard evidence (i.e., $Z = z$), (2.20) reduces to

$$P(X|\varepsilon) = P(X|Z = z),$$

i.e., a simple look-up. ■

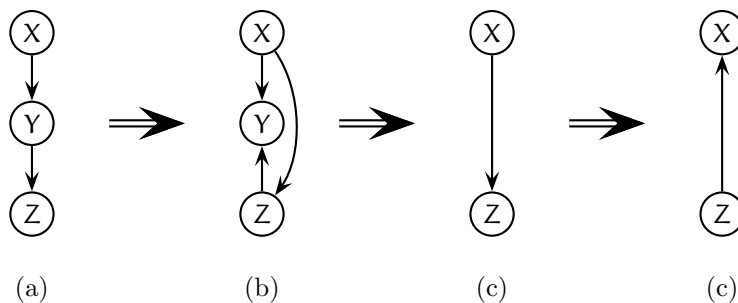


Figure 2.8: (a) Model for $P(X, Y, Z)$. (b) Equivalent model obtained by reversing $Y \rightarrow Z$. (c) Equivalent model provided Y is barren. (d) Equivalent model obtained by reversing $X \rightarrow Z$.

We shall return to the arc reversal approach to inference in more detail in Section 4.1.1 on page 94.

2.5 Bayes' Factor

To calculate the relative support provided by an observation, $Y = \mathbf{y}$, for two competing hypotheses, $H_0 : X = x_0$ and $H_1 : X = x_1$, the notion of *Bayes' factor* is useful:

$$\begin{aligned} B &= \frac{\text{posterior odds ratio}}{\text{prior odds ratio}} = \frac{P(x_0 | \mathbf{y}) / P(x_1 | \mathbf{y})}{P(x_0) / P(x_1)} \\ &= \frac{P(x_0, \mathbf{y}) / P(x_1, \mathbf{y})}{P(x_0) / P(x_1)} = \frac{P(x_0, \mathbf{y}) / P(x_0)}{P(x_1, \mathbf{y}) / P(x_1)} = \frac{P(\mathbf{y} | x_0)}{P(\mathbf{y} | x_1)} = \frac{L(x_0 | \mathbf{y})}{L(x_1 | \mathbf{y})}; \end{aligned} \quad (2.21)$$

that is, the ratio of the likelihoods of the two hypotheses given the observation. Bayes' factor is also known as the *Bayesian likelihood ratio*.

From (2.21) we see that

$B > 1$ if the observation provides more support for H_0 than for H_1 ,

$B < 1$ if the observation provides less support for H_0 than for H_1 , and

$B = 1$ if the observation does not provide useful information for differentiating between H_0 and H_1 .

Example 22 (Balls in An Urn, page 29) Let

H_0 : The second ball drawn will be green: $X_2 = \text{green}$

H_1 : The second ball drawn will be blue: $X_2 = \text{blue}$

be two competing hypotheses, and assume that the first ball drawn is blue (i.e., $X_1 = \text{blue}$). Now, using the numbers calculated in Example 12 on page 31, we get the Bayes' factor

$$B = \frac{P(X_2 = \text{green} | X_1 = \text{blue}) / P(X_2 = \text{blue} | X_1 = \text{blue})}{P(X_2 = \text{green}) / P(X_2 = \text{blue})} = \frac{\frac{3}{9} / \frac{4}{9}}{\frac{3}{10} / \frac{5}{10}} = \frac{5}{4}.$$

That is, since the posterior odds ratio (3/4) is greater than the prior odds ratio (3/5), the observation provides more support for H_0 than for H_1 . Still, however, the probability that H_1 is going to be true is greater than the probability that H_0 is going to be true, as $P(H_0 | X_1 = \text{blue}) = 3/9$ and $P(H_1 | X_1 = \text{blue}) = 4/9$. ■

2.6 Independence

A variable X is *independent* of another variable Y with respect to a probability distribution P if

$$P(x|y) = P(x), \forall x \in \text{dom}(X), \forall y \in \text{dom}(Y). \quad (2.22)$$

Using the standard notation introduced in Section 1.6 on page 19 we express this property symbolically as $X \perp\!\!\!\perp_P Y$, or simply as $X \perp\!\!\!\perp Y$ when P is obvious from the context. Symmetry of independence (i.e., $X \perp\!\!\!\perp Y \equiv Y \perp\!\!\!\perp X$) can be verified from Bayes' rule:

$$P(x) = P(x|y) = \frac{P(y|x)P(x)}{P(y)} \Leftrightarrow P(y) = P(y|x).$$

The statement $X \perp\!\!\!\perp Y$ is often referred to as *marginal independence* between X and Y .

If $X \perp\!\!\!\perp Y$, then from the fundamental rule (2.10) and (2.22) we get that

$$P(X, Y) = P(X|Y)P(Y) = P(X)P(Y), \quad (2.23)$$

and, in general, whenever X_1, \dots, X_n are pairwise independent random variables their joint distribution equals the product of the marginals:

$$P(X_1, \dots, X_n) = \prod_i P(X_i). \quad (2.24)$$

A variable X is *conditionally independent* of Y given Z (with respect to a probability distribution P) if

$$P(x|y, z) = P(x|z), \forall x \in \text{dom}(X), \forall y \in \text{dom}(Y), \forall z \in \text{dom}(Z). \quad (2.25)$$

The conditional independence statement expressed in (2.25) is indicated as $X \perp\!\!\!\perp Y | Z$ in our standard notation.

Again, from the fundamental rule and (2.25) we get

$$\begin{aligned} P(X, Y, Z) &= P(X|Y, Z)P(Y, Z) \\ &= P(X|Y, Z)P(Y|Z)P(Z) \\ &= P(X|Z)P(Y|Z)P(Z). \end{aligned}$$

Example 23 (Conditional independence) Consider the Burglary or Earthquake example from page 10. With $P(R|E)$ given as

$$P(R|E) = \begin{array}{c|cc} & E = \text{no} & E = \text{yes} \\ \hline R = \text{no} & 0.999 & 0.01 \\ R = \text{yes} & 0.001 & 0.99 \end{array}$$

and $P(A, E) = \sum_B P(A, B, E)$ given as in Example 20 on page 41 we can compute $P(A|E, R)$ to be given as

$$\begin{aligned} P(A|E, R) &= \frac{P(A, E, R)}{\sum_A P(A, E, R)} = \frac{P(A, E)P(R|E)}{\sum_A P(A, E)P(R|E)} & (2.26) \\ &= \begin{array}{c|cccc} & \begin{array}{cc} R = \text{no} & R = \text{yes} \end{array} \\ \hline & E = \text{no} & E = \text{yes} & E = \text{no} & E = \text{yes} \\ \hline A = \text{no} & 0.901 & 0.091 & 0.901 & 0.091 \\ A = \text{yes} & 0.099 & 0.909 & 0.099 & 0.909 \end{array} \end{aligned}$$

Obviously, $P(A = a|E = e, R = \text{no}) = P(A = a|E = e, R = \text{yes})$ for each pair $(a, e) \in \{\text{no}, \text{yes}\} \times \{\text{no}, \text{yes}\}$ and thus $P(A|E, R) = P(A|E)$, meaning that $A \perp\!\!\!\perp_P R|E$, which also appears clearly in (2.26), as

$$P(A, E, R) = P(R|A, E)P(A, E) = P(R|E)P(A, E),$$

entailing this conditional independence assumption. ■

2.6.1 Independence and DAGs

Let P be a probability distribution over a set of variables X_V and let $\mathcal{G} = (V, E)$ be a DAG. Then for each $A, B, S \subseteq V$, where $A \cap B = \emptyset$:

- \mathcal{G} is a *dependence map* (or *D-map*) of P if

$$X_A \perp\!\!\!\perp_P X_B | X_S \implies A \perp_G B | S, \quad (2.27)$$

- \mathcal{G} is an *independence map* (or *I-map*) of P if

$$X_A \perp\!\!\!\perp_P X_B | X_S \iff A \perp_G B | S, \quad (2.28)$$

- \mathcal{G} is a *perfect map* of P if it is both a D-map and an I-map of P :

$$X_A \perp\!\!\!\perp_P X_B | X_S \iff A \perp_G B | S. \quad (2.29)$$

In other words, \mathcal{G} is a D-map of P if any independence statement in P has a corresponding irrelevance statement in \mathcal{G} (see Section 1.6 on page 19). Note that the empty graph (i.e., $E = \emptyset$) is a D-map of any distribution over X_V . \mathcal{G} is an I-map of \mathcal{G} if any irrelevance statement in \mathcal{G} has a corresponding independence statement in P . Note that the complete graph (i.e., $E = V \times V$) is an I-map of any distribution over X_V .

Example 24 Let X , Y , and Z be three variables for which $X \perp\!\!\!\perp_P Y|Z$. Following the ordering (X, Y, Z) and using the fundamental rule (see (2.10) on page 40) twice yield

$$P(X, Y, Z) = P(X|Y, Z)P(Y|Z)P(Z).$$

Since $X \perp\!\!\!\perp_P Y|Z$, this can be simplified to

$$P(X, Y, Z) = P(X|Z)P(Y|Z)P(Z). \quad (2.30)$$

Similarly, following the orderings (X, Z, Y) and (Y, Z, X) we get, respectively,

$$P(X, Y, Z) = P(X|Z)P(Z|Y)P(Y) \quad (2.31)$$

and

$$P(X, Y, Z) = P(Y|Z)P(Z|X)P(X). \quad (2.32)$$

Factorizations (2.30)–(2.32) have graphical representations as shown in Figures 2.9a–c (cf. Section 2.2.2 on page 33). ■

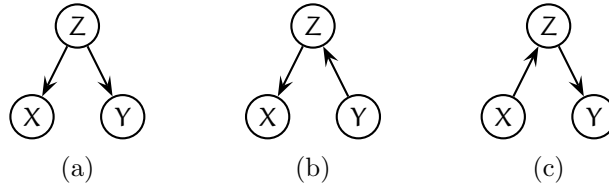


Figure 2.9: Graphical representations of $X \perp\!\!\!\perp_P Y|Z$, representing, respectively, factorizations (2.30)–(2.32).

DAGs defined in terms of recursive factorizations of joint probability distributions (for example, as those in Figures 2.9a–c) will always be perfect maps, unless there are some regularities in the probability distributions that entails additional independences (e.g., if for the model in (2.30) it holds true that $P(Y|z_i) = P(Y|z_j)$ for all i, j (i.e., $Y \perp\!\!\!\perp_P Z$), then the DAG in Figure 2.9(a) is not a perfect map of P). In general, however, if the independence statements are provided as an arbitrary list of statements, it might only be possible to represent a subset of them in a DAG, rendering the DAG an imperfect map. That is, the DAG language is not rich enough to simultaneously capture all sets of independence statements. To illustrate this fact, consider the following example.

Example 25 Let P be a probability distribution over $\{X_\alpha, X_\beta, X_\gamma, X_\delta\}$ that encodes the following independence statements:

$$\begin{aligned} \text{CID}_1 &: X_\alpha \perp\!\!\!\perp_P X_\beta \\ \text{CID}_2 &: X_\alpha \perp\!\!\!\perp_P X_\delta | \{X_\beta, X_\gamma\} \\ \text{CID}_3 &: X_\beta \perp\!\!\!\perp_P X_\gamma | \{X_\alpha, X_\delta\}, \end{aligned}$$

and let $\mathcal{G} = (\{\alpha, \beta, \gamma, \delta\}, E)$ be a DAG.

Let us try to identify the set of links of \mathcal{G} such that it becomes a perfect map of P . First, we can conclude that there must be links between each pair of nodes except (α, β) , (α, δ) , and (β, γ) , as at least one independence statement has been specified for each of the pairs $\{X_\alpha, X_\beta\}$, $\{X_\alpha, X_\delta\}$, and $\{X_\beta, X_\gamma\}$, respectively. A preliminary skeleton of the possible DAGs therefore appears as shown in Figure 2.10a.

Recalling the d-separation criterion or the directed global Markov criterion (see Section 1.6 on page 19) we see that for CID_1 to hold true there must be a converging connection at γ or δ . However, a converging connection at e.g. γ implies $\alpha \perp_{\mathcal{G}} \delta$, violating the possibility of \mathcal{G} to be an I-map of P , as $\alpha \perp_{\mathcal{G}} \delta \not\perp P X_\alpha \perp P X_\delta$. To remedy that, we will have to include a link between α and δ . Now, to ensure $\alpha \perp_{\mathcal{G}} \beta$, the links $\alpha - \delta$ and $\beta - \delta$ must meet head-to-head at δ (i.e., must converge at δ). The resulting DAG in Figure 2.10b is an I-map of P but it is not a D-map of P because $X_\alpha \perp P X_\delta | \{X_\beta, X_\gamma\} \not\perp \alpha \perp_{\mathcal{G}} \delta | \{\beta, \gamma\}$. Similarly, the DAG in Figure 2.10c is an I-map of P but not a D-map, as $X_\beta \perp P X_\gamma | \{X_\alpha, X_\delta\} \not\perp \beta \perp_{\mathcal{G}} \gamma | \{\alpha, \delta\}$.

We conclude that there does not exist a perfect map of P — one would have to settle with either the DAG in Figure 2.10b or the one in Figure 2.10c, which are both I-maps of P but not D-maps. ■

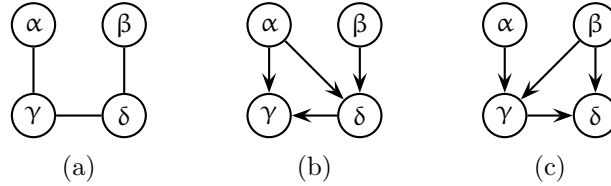


Figure 2.10: (a) Preliminary skeleton for the independence statements of Example 25. (b) Perfect map of $\{CID_1, CID_3\}$. (c) Perfect map of $\{CID_1, CID_2\}$.

2.7 Chain Rule

For a probability distribution, $P(X)$, over a set of variables $X = \{X_1, \dots, X_n\}$, we can use the fundamental rule repetitively to decompose it into a product of conditional probability distributions:

$$\begin{aligned}
 P(X) &= P(X_1 | X_2, \dots, X_n) P(X_2, \dots, X_n) \\
 &= P(X_1 | X_2, \dots, X_n) P(X_2 | X_3, \dots, X_n) \cdots P(X_{n-1} | P_n) P(X_n) \\
 &= \prod_{i=1}^n P(X_i | X_{i+1}, \dots, X_n). \tag{2.33}
 \end{aligned}$$

Notice that the actual conditional distributions that comprise the factors of the decomposition are determined by the order in which we select the head variables of the conditional distributions. Thus, there are n factorial different factorizations of $P(X)$, and to each factorization corresponds a unique DAG, but all of these DAGs are equivalent in terms of dependence and independence properties, as they are all complete graphs, and hence represent no independence statements.

Example 26 (Chain decomposition and DAGs) Let $V = \{\alpha, \beta, \gamma, \delta\}$ be a set of nodes, and let P be a probability distribution defined over X_V . Then $P(X_V)$ factorizes as

$$\begin{aligned} P(X_V) &= P(X_\alpha, X_\beta, X_\gamma, X_\delta) = P(X_\alpha | X_\beta, X_\gamma, X_\delta) P(X_\beta, X_\gamma, X_\delta) \\ &= P(X_\alpha | X_\beta, X_\gamma, X_\delta) P(X_\beta | X_\gamma, X_\delta) P(X_\gamma, X_\delta) \\ &= P(X_\alpha | X_\beta, X_\gamma, X_\delta) P(X_\beta | X_\gamma, X_\delta) P(X_\gamma | X_\delta) P(X_\delta) & (2.34) \\ &= P(X_\beta | X_\alpha, X_\gamma, X_\delta) P(X_\delta | X_\alpha, X_\gamma) P(X_\gamma | X_\alpha) P(X_\alpha) & (2.35) \\ &= \dots \end{aligned}$$

The DAGs corresponding to (2.34) and (2.35) appear in Figures 2.11a and 2.11b, respectively. ■

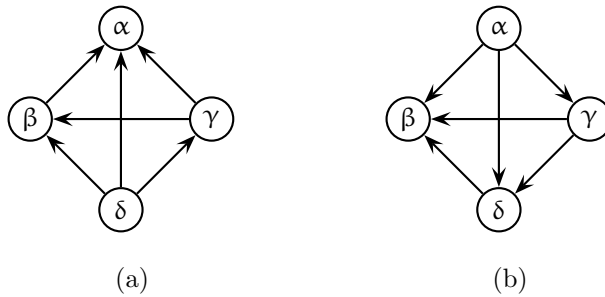


Figure 2.11: (a) DAG corresponding to decomposition (2.34). (b) DAG corresponding to decomposition (2.35).

Assume that a DAG \mathcal{G} is an I-map of P and that the order in which we select the head variables of the conditional distributions in (2.33) respect a topological ordering $(X_{v_1}, \dots, X_{v_n})$ ⁶ with respect to \mathcal{G} : $\text{pa}(v_i) \subseteq \{v_1, \dots, v_{i-1}\}$ for all $i = 1, \dots, n$ (i.e., the parents of each variable are selected before the variable itself). That is, the tail variables of the conditional distributions in (2.33) always include the parents.

⁶For notational convenience, we assume (without loss of generality) that v_1, \dots, v_n is a topological ordering.

It follows easily from the d-separation criterion or the directed Markov criterion (see Section 1.6 on page 19) that for any node v of \mathcal{G} , $v \perp_{\mathcal{G}} \text{nd}(v) | \text{pa}(v)$.⁷ Since \mathcal{G} is an I-map of \mathbb{P} , it follows that $X_v \perp_{\mathbb{P}} \text{nd}(X_v) | X_{\text{pa}(v)}$. Therefore, each term $\mathbb{P}(X_{v_i} | X_{v_1}, \dots, X_{v_i})$ can be reduced to $\mathbb{P}(X_{v_i} | X_{\text{pa}(v_i)})$. The product (2.33) then simplifies to the *chain rule*:

$$\mathbb{P}(X_V) = \prod_{i=1}^n \mathbb{P}(X_{v_i} | X_{\text{pa}(v_i)}). \quad (2.36)$$

Example 27 (Example 26 continued) Assume that the complete set of independence properties that \mathbb{P} satisfies comprises $X_{\beta} \perp_{\mathbb{P}} X_{\gamma} | X_{\alpha}$ and $X_{\alpha} \perp_{\mathbb{P}} X_{\delta} | \{X_{\beta}, X_{\gamma}\}$. Then the DAG in Figure 2.12 is a perfect map of \mathbb{P} . From the chain rule we can therefore write the joint distribution as

$$\mathbb{P}(X_V) = \mathbb{P}(X_{\alpha})\mathbb{P}(X_{\beta} | X_{\alpha})\mathbb{P}(X_{\gamma} | X_{\alpha})\mathbb{P}(X_{\delta} | X_{\beta}, X_{\gamma}),$$

where the tail variables are exactly the set of parents for each head variable. ■

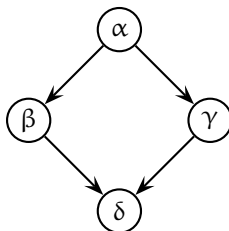


Figure 2.12: Perfect map of a distribution \mathbb{P} that encodes the independence statements $X_{\beta} \perp_{\mathbb{P}} X_{\gamma} | X_{\alpha}$ and $X_{\alpha} \perp_{\mathbb{P}} X_{\delta} | \{X_{\beta}, X_{\gamma}\}$.

2.8 Summary

This chapter has provided an introduction to Bayesian probability theory and made a connection between the notion of irrelevance in DAGs (d-separation) discussed in Chapter 1 and the concept of probabilistic independence, and shown that, whenever a joint probability distribution (i.e., a probabilistic model) is specified in terms of a product of lower-dimensional conditional distributions, the associated DAG induced by this recursive factorization will be an I-map (and also often a D-map) of the independence statements entailed by the joint distribution. This is in fact how a probabilistic graphical model is almost always specified, namely as descriptions of local (causal) phenomena, where a

⁷This result is probably easiest to acknowledge from the directed Markov criterion: The graph $(\mathcal{G}_{\text{An}(\{v\} \cup \text{nd}(v) \cup \text{pa}(v))})^m$ obviously excludes all descendants of v forcing all paths to v to go through $\text{pa}(v)$.

domain expert provides assessments of cause-effect relations and their associated conditional probability distributions, encountering for the strengths of the relations.

So, the problem mentioned in Section 2.6.1, where there might not always exist a perfect map of a probability distribution is seldomly a practical one. In the rare cases, where a model is not given through a product of lower-dimensional conditional distributions but rather is described in terms of a list of independence statements of some (unknown) joint probability distribution such that a perfect map does not exist, should one then go for a D-map, an I-map, or some kind of compromise between the two? Fortunately, there is an easy answer to this question. Since, namely, a probability distribution can always be represented in a complete DAG and since a model encoding false independence statements may produce false conclusions, one should always go for a DAG with the least number of links that is an I-map (i.e., a minimal I-map) of the distribution.

The Bayesian view of probabilities differentiates itself from the classical frequentist view by considering probabilities as expressing subjective assessments of belief rather than objective measures of asymptotic frequencies of events. In a frequentist view of probabilities, for example, the probability that a coin will land heads up can be established only through trials as the fraction of heads in the limit of an infinite number of trials. In a Bayesian perspective such a probability can be established through a subjective assessment, reflecting a personal degree of belief whether the event will occur.

Collectively, the use of the language of causality on the one hand and subjective assessments of probabilities on the other to express strengths of causal relations provide a strong paradigm for formulating models for reasoning under uncertainty. In Chapter 3 we shall discuss the details of such models, and in addition see how they can be augmented with decision variables and utility functions for explicit representation and solution of sequential decision problems.

Chapter 3

Probabilistic Networks

In this chapter we introduce probabilistic networks for reasoning and decision making under uncertainty.

Many real-life situations can be modeled as a domain of entities represented as random variables in a probabilistic network. A probabilistic network is a clever graphical representation of dependence and independence relations between random variables. A domain of random variables can, for instance, form the basis of a decision support system to help decision makers identify the most beneficial decision in a given situation.

A probabilistic network represents and processes probabilistic knowledge. The representational components of a probabilistic network are a qualitative and a quantitative component. The qualitative component encodes a set of (conditional) dependence and independence statements among a set of random variables, informational precedence, and preference relations. The statements of (conditional) dependence and independence, information precedence, and preference relations are visually encoded using a graphical language. The quantitative component, on the other hand, specifies the strengths of dependence relations using probability theory and preference relations using utility theory.

The graphical representation of a probabilistic network describes knowledge of a problem domain in a precise manner. The graphical representation is intuitive and easy to comprehend, making it an ideal tool for communication of domain knowledge between experts, users, and systems. For these reasons, the formalism of probabilistic networks is becoming an increasingly popular domain knowledge representation for reasoning and decision making under uncertainty.

Since a probabilistic network consists of two components it is customary to consider its constructions as a two phase process. The construction of the qualitative component and subsequently the construction of the quantitative component. The qualitative component defines the structure of the quantitative component. As the first step, the qualitative structure of the model is constructed using a graphical language. This step consists of identifying variables and relations between variables. As the second step, the parameters of the quantitative part as defined by the qualitative part are assessed.

In this book, we consider the subclass of probabilistic networks known as Bayesian networks and influence diagrams. Bayesian networks and influence diagrams are ideally suited knowledge representations for use in many situations involving reasoning and decision making under uncertainty. These models are often characterized as normative expert systems as they provide model-based domain descriptions, where the model is reflecting properties of the problem domain (rather than the domain expert) and probability calculus is used as the calculus for uncertainty.

A Bayesian network model representation of a problem domain can be used as the basis for performing inference and analyzing about the domain. Decision options and utilities associated with these options can be incorporated explicitly into the model, in which case the model becomes an influence diagram, capable of computing expected utilities of all decision options given the information known at the time of decision. Bayesian networks and influence diagrams are applicable for a very large range of domain areas with inherent uncertainty.

Section 3.1 considers Bayesian networks as probabilistic models for reasoning under uncertainty. We consider Bayesian network models containing discrete variables only and models containing a mixture of continuous and discrete variables. Section 3.2 considers influence diagrams as probabilistic networks for decision making under uncertainty. The influence diagram is a Bayesian network augmented with decision variables, informational precedence relations, and preference relations. We consider influence diagram models containing discrete variables only and models containing a mixture of continuous and discrete variables. In Section 3.3 object-oriented probabilistic networks are considered. An object-oriented probabilistic network is a flexible framework for building hierarchical knowledge representations using the notions of classes and instances.

3.1 Reasoning Under Uncertainty

A probabilistic interaction model between a set of random variables may be represented as a joint probability distribution. Considering the case where random variables are discrete, it is obvious that the size of the joint probability distribution will grow exponentially with the number of variables as the joint distribution must contain one probability for each configuration of the random variables. Therefore, we need a more compact representation for reasoning about the state of large and complex systems involving a large number of variables.

To facilitate an efficient representation of a large and complex domain with many random variables, the framework of Bayesian networks uses a graphical representation to encode dependence and independence relations among the random variables. The dependence and independence relations induce a compact representation of the joint probability distribution. By representing the dependence and independence relations of a domain explicitly in a graph, a compact representation of the dependence and independence relations is obtained.

3.1.1 Discrete Bayesian Networks

A (discrete) Bayesian network, $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P})$, over variables, \mathcal{X} , consists of an acyclic, directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and a set of conditional probability distributions \mathcal{P} . Each node v in \mathcal{G} corresponds one-to-one with a discrete random variable $X_v \in \mathcal{X}$ with a finite set of mutually exclusive states. The directed links $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ of \mathcal{G} specify assumptions of conditional dependence and independence between random variables according to the d-separation criterion (see Definition 4 on page 21).

There is a conditional probability distribution, $P(X_v | X_{\text{pa}(v)}) \in \mathcal{P}$, for each variable $X_v \in \mathcal{X}$. The set of variables represented by the parents, $\text{pa}(v)$, of $v \in \mathcal{V}$ in $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ are sometimes called the *conditioning variables* of X_v — the *conditioned* variable.

Definition 1 (Jensen 2001) A (discrete) Bayesian network $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P})$ consists of

- A DAG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with nodes $\mathcal{V} = \{v_1, \dots, v_n\}$ and directed links \mathcal{E} .
- A set of discrete random variables, \mathcal{X} , represented by the nodes of \mathcal{G} .
- A set of conditional probability distributions, \mathcal{P} , containing one distribution, $P(X_v | X_{\text{pa}(v)})$, for each random variable $X_v \in \mathcal{X}$.

A Bayesian network encodes a joint probability distribution over a set of random variables, \mathcal{X} , of a problem domain. The set of conditional probability distributions, \mathcal{P} , specifies a multiplicative factorization of the joint probability distribution over \mathcal{X} as represented by the chain rule of Bayesian networks (see Section 2.7 on page 48):

$$P(\mathcal{X}) = \prod_{v \in \mathcal{V}} P(X_v | X_{\text{pa}(v)}). \quad (3.1)$$

Even though the joint probability distribution specified by a Bayesian network is defined in terms of conditional independence, a Bayesian network is most often constructed using the notion of cause-effect relations, see Section 1.5. In practice, cause-effect relations between entities of a problem domain can be represented in a Bayesian network using a graph of nodes representing random variables and links representing cause-effect relations between the entities. Usually, the construction of a Bayesian network (or any probabilistic network for that matter) proceeds according to an iterative procedure where the set of nodes and their states, and the set of links are updated iteratively as the model becomes more and more refined.

To solve a Bayesian network $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P})$ is to compute all posterior marginals given a set of evidence ε , i.e., $P(X | \varepsilon)$ for all $X \in \mathcal{X}$. If the evidence set is empty, i.e., $\varepsilon = \emptyset$, then the task is to compute all prior marginals, i.e., $P(X)$ for all $X \in \mathcal{X}$.

Example 28 (Apple Jack (Madsen & Nielsen 1996)) Consider the small orchard belonging to Jack Fletcher (let's call him Apple Jack). One day Apple Jack discovers that his finest apple tree is *loosing its leaves*. Apple Jack wants to know why this is happening. He knows that if the tree is *dry* (for instance, caused by a drought) there is no mystery as it is very common for trees to loose their leaves during a drought. On the other hand, the loss of leaves can be *an indication of a disease*.

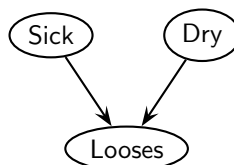


Figure 3.1: The Apple Jack network.

The qualitative knowledge about the cause-effect relations of this situation can be modeled by the DAG \mathcal{G} shown in Figure 3.1. The graph consists of three nodes: Sick, Dry, and Looses that represent variables of the same names. Each variable may be in one of two states: no and yes, i.e., $\text{dom}(\text{Dry}) = \text{dom}(\text{Looses}) = \text{dom}(\text{Sick}) = \{\text{no}, \text{yes}\}$. The variable Sick tells us that the apple tree is sick by being in state yes. Otherwise, it will be in state no. The variables Dry and Looses tell us whether or not the tree is dry and whether or not the tree is losing its leaves, respectively.

The graph, \mathcal{G} , shown in Figure 3.1 models the cause-effect relations between variables Sick and Looses as well as between Dry and Looses. This is represented by the two (causal) links (Sick, Looses) and (Dry, Looses). In this way Sick and Dry are common causes of the effect Looses.

Let us return to the discussion of causality considered previously in Section 1.5. When there is a causal dependence relation going from a variable A to another variable B , we expect that when A is in a certain state this has an impact on the state of B . One should be careful when modeling causal dependence relations in a Bayesian network. Sometimes it is not quite obvious in which direction a link should point. In the Apple Jack example, we say that there is a causal impact from Sick on Looses, because when a tree is sick this might cause the tree to loose its leaves. Why can we not say that when the tree looses its leaves it might be sick and turn the link in the other direction? The reason is that it is the sickness that causes the tree to loose its leaves and not the lost leaves that causes the sickness.

Figure 3.1 shows the graphical representation of the Bayesian network model. This is referred to as the qualitative representation. To have a complete Bayesian network, we need to specify the quantitative representation. Recall that each variable has two states no and yes.

D	S	L	
		no	yes
no	no	0.98	0.02
no	yes	0.1	0.9
yes	no	0.15	0.85
yes	yes	0.05	0.95

Table 3.1: The conditional probability distribution $P(L|D, S)$.

The quantitative representation of a Bayesian network is the set of conditional probability distributions, \mathcal{P} , defined by the structure of \mathcal{G} . Table 3.1 shows the conditional probability distribution of Looses given Sick and Dry, i.e., $P(\text{Looses}|\text{Dry}, \text{Sick})$. For variables Sick and Dry we assume that $P(S) = (0.9, 0.1)$ and $P(D) = (0.9, 0.1)$ (we use D as short for Dry, S as short for Sick, and L as short for Looses).

Note that all distributions specify the probability of a variable being in a specific state depending on the configuration of its parent variables, but since Sick and Dry do not have any parent variables, their distributions are marginal distributions.

The model may be used to compute all prior marginals and the posterior distribution of each non-evidence variable given evidence in the form of observations on a subset of the variables in the model. The priors for D and S equals the specified marginal distributions, i.e., $P(D) = P(S) = (0.9, 0.1)$, while the prior distribution for L is computed through combination of the distributions specified for the three variables, followed by marginalization, where variables D and S are marginalized out. This yields $P(L) = (0.82, 0.18)$ (see Example 17 on page 36 for details on combination and marginalization). Following a similar procedure, the posteriors of D and S given $L = \text{yes}$ can be computed to be $P(D|L = \text{yes}) = (0.53, 0.47)$ and $P(S|L = \text{yes}) = (0.51, 0.49)$. Thus, according to the model the tree being sick is the most likely cause of the loss of leaves. ■

The specification of a conditional probability distribution $P(X_v|X_{\text{pa}(v)})$ can be a labor intensive knowledge acquisition task as the number of parameters grows exponentially with the size of $\text{dom}(X_{\text{fa}(v)})$, where $\text{fa}(v) = \text{pa}(v) \cup \{v\}$. Different techniques can be used to simplify the knowledge acquisition task, assumptions can be made, or the parameters can be estimated from data.

The complexity of a Bayesian network is defined in terms of the family $\text{fa}(v)$ with the largest state space size $\|X_{\text{fa}(v)}\| \triangleq |\text{dom}(X_{\text{fa}(v)})|$. As the state space size of a family of variables grows exponentially with the size of the family, we seek to reduce the size of the parent sets to a minimum. Another useful measure of the complexity of a Bayesian network is the number of cycles and the length of cycles in its graph.

Definition 2 A Bayesian network $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P})$ is minimal if and only if, for every variable $X_v \in \mathcal{X}$ and for every parent $Y \in X_{\text{pa}(v)}$, X_v is not independent of Y given $X_{\text{pa}(v)} \setminus \{Y\}$.

Definition 2 says that the parent set $X_{\text{pa}(v)}$ of X_v should be limited to the set of variables with a direct impact on X_v .

Example 29 (Chest Clinic (Lauritzen & Spiegelhalter 1988)) A physician at a chest clinic wants to diagnose her patients with respect to three diseases based on observations of symptoms and possible causes of the diseases. The fictitious qualitative medical knowledge is the following.

The physician is trying to diagnose a patient who may be suffering from one or more of *tuberculosis*, *lung cancer*, or *bronchitis*. *Shortness-of-breath* (dyspnoea) may be due to tuberculosis, lung cancer, bronchitis, none of them, or more than one of them. *A recent visit to Asia* increases the chances of tuberculosis, while *smoking* is known to be a risk factor for both lung cancer and bronchitis. *The results of a single chest X-ray* do not discriminate between lung cancer and tuberculosis, as neither does the presence or absence of dyspnoea.

From the description of the situation it is clear that there are three possible diseases to consider (lung cancer, tuberculosis, and bronchitis). The three diseases produce three variables Tuberculosis (T), Cancer (L), and Bronchitis (B) with states no and yes. These variables are the targets of the reasoning and may, for this reason, be referred to as *hypothesis variables*. The diseases may be manifested in two symptoms (results of the X-ray and shortness-of-breath). The two symptoms produce two variables X_ray (X), and Dyspnoea (D) with states no and yes. In addition, there are two causes or risk factors (smoking and a visit to Asia) to consider. The two risk factors produce variables Asia (A) and Smoker (S) with states no and yes

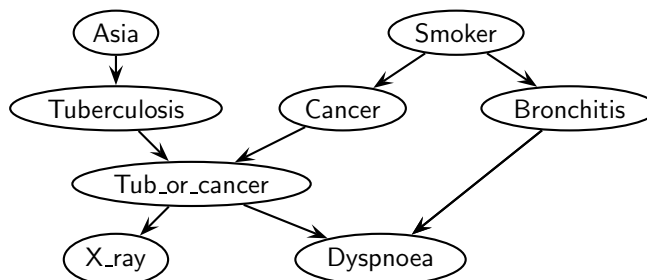


Figure 3.2: A graph specifying the independence and dependence relations of the Asia example.

An acyclic, directed graph, \mathcal{G} , encoding the above medical qualitative knowledge is shown in Figure 3.2, where the variable Tub_or_cancer (E) is a mediating

$P(L S)$	$S = \text{no}$	$S = \text{yes}$	$P(B S)$	$S = \text{no}$	$S = \text{yes}$
$L = \text{no}$	0.99	0.9	$B = \text{no}$	0.7	0.4
$L = \text{yes}$	0.01	0.1	$B = \text{yes}$	0.3	0.6

$P(T A)$	$A = \text{no}$	$A = \text{yes}$	$P(X E)$	$E = \text{no}$	$E = \text{yes}$
$T = \text{no}$	0.99	0.95	$X = \text{no}$	0.95	0.02
$T = \text{yes}$	0.01	0.05	$X = \text{yes}$	0.05	0.98

Table 3.2: The conditional probability distributions $P(L|S)$, $P(B|S)$, $P(T|A)$, and $P(X|E)$.

variable specifying whether or not the patient has tuberculosis or lung cancer (or both).

Using the structure of \mathcal{G} , we may perform an analysis of dependence and independence properties between variables in order to ensure that the qualitative structure encodes the domain knowledge correctly. This analysis would be based on an application of the d-separation criterion.

Figure 3.2 on the facing page only presents the qualitative structure \mathcal{G} (and the variables) of $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P})$. In order to have a fully specified Bayesian network, it is necessary to specify the quantitative part, \mathcal{P} , too.

The quantitative domain knowledge is specified in the following set of (conditional) probability distributions $P(A) = (0.99, 0.01)$, $P(S) = (0.5, 0.5)$, and the remaining conditional probability distributions, except $P(E|L, T)$, shown in Tables 3.2 and 3.3.

The conditional probability table of the random variable E can be generated from a mathematical expression. From our domain knowledge of the diagnosis problem we know that E represents the disjunction of L and T . That is, E represents whether or not the patient has tuberculosis or lung cancer. From this we can express E as $E = T \vee L$. This produces the conditional probability $P(E = \text{yes} | L = l, T = t) = 1$ whenever l or t is yes.

	$B = \text{no}$		$B = \text{yes}$	
	$E = \text{no}$	$E = \text{yes}$	$E = \text{no}$	$E = \text{yes}$
$D = \text{no}$	0.9	0.3	0.2	0.1
$D = \text{yes}$	0.3	0.7	0.8	0.9

Table 3.3: The conditional probability distribution $P(D|B, E)$.

Using the Bayesian network model just developed, we may compute the posterior probability of the three diseases given various subsets of evidence on the causes and symptoms as shown in Table 3.4 on the next page. ■

ε	$P(B = \text{yes} \varepsilon)$	$P(L = \text{yes} \varepsilon)$	$P(T = \text{yes} \varepsilon)$
\emptyset	0.45	0.055	0.01
$\{S = \text{yes}\}$	0.6	0.1	0.01
$\{S = \text{yes}, D = \text{yes}\}$	0.88	0.15	0.015
$\{S = \text{yes}, D = \text{yes}, X = \text{yes}\}$	0.71	0.72	0.08

Table 3.4: Posterior distributions of the disease variables given various evidence scenarios.

3.1.2 Linear Conditional Gaussian Bayesian Networks

Up until now, we have considered Bayesian networks over discrete random variables only. However, there are many reasons for extending our considerations to include continuous variables. In this section we will consider Bayesian networks consisting of both continuous and discrete variables. For reasons to become clear later, we restrict our attention to the case of linear conditional Gaussian (also known as Normal) distributions and the case of linear conditional Gaussian Bayesian networks. We refer to a linear conditional Gaussian Bayesian network as an LCG Bayesian network.

An LCG Bayesian network $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P}, \mathcal{F})$ consists of an acyclic, directed graph $\mathcal{G} = (V, E)$, a set of conditional probability distributions \mathcal{P} , and a set of density functions \mathcal{F} . There will be one conditional probability distribution for each discrete random variable X of \mathcal{X} and one density function for each continuous random variable Y of \mathcal{X} .

An LCG Bayesian network specifies a distribution over a mixture of discrete and continuous variables (Lauritzen 1992, Lauritzen & Jensen 2001). The variables, \mathcal{X} , are partitioned into the set of continuous variables, \mathcal{X}_Γ , and the set of discrete variables, \mathcal{X}_Δ . Each node of \mathcal{G} represents either a discrete random variable with a finite set of mutually exclusive and exhaustive states or a continuous random variable with a linear conditional Gaussian distribution conditional on the configuration of its discrete parent variables. This implies an important constraint on the structure of \mathcal{G} , namely that a discrete random variable X_v may only have discrete parents, i.e., $X_{\text{pa}(v)} \subseteq \mathcal{X}_\Delta$ for any $X_v \in \mathcal{X}_\Delta$.

Any Gaussian distribution function can be specified by its mean and variance parameter. As mentioned above, we consider the case where a continuous random variable can have a single Gaussian distribution function for each configuration of its discrete parent variables. If a continuous variable has one or more continuous variables as parents, the mean may depend linearly on the state of the continuous parent variables. Continuous parent variables of discrete variables are disallowed.

A random variable, X , has a continuous distribution if there exists a non-negative function p , defined on the real line, such that for any interval J :

$$P(X \in J) = \int_J p(x) dx,$$

where the function p is the probability density function of X (DeGroot 1986). The probability density function of a *Gaussian* (or *Normal*) *distributed variable*, X , with a mean value, μ , and a positive variance, σ^2 , is (i.e., $X \sim \mathbb{N}(\mu, \sigma^2)$) or $\mathcal{L}(X) = \mathbb{N}(\mu, \sigma^2)$)

$$p(x; \mu, \sigma^2) = \mathbb{N}(\mu, \sigma^2) = \frac{1}{\sqrt{(2\pi\sigma^2)}} \exp \left[-\frac{(x - \mu)^2}{2\sigma^2} \right],$$

where $x \in \mathbb{R}$.

A continuous random variable, X , has a *linear conditional Gaussian distribution* (or LCG distribution), conditional on the configuration of the parent variables ($Z \subseteq \mathcal{X}_\Gamma, I \subseteq \mathcal{X}_\Delta$) if

$$\mathcal{L}(X|Z = z, I = i) = \mathbb{N}(A(i) + B(i)^T z, C(i)), \quad (3.2)$$

where A is a table of mean values (one value for each configuration i of the discrete parent variables I), B is a table of regression coefficient vectors (one vector for each configuration i of I with one regression coefficient for each continuous parent variable), and C is a table of variances (one for each configuration i of I). Notice that the mean value $A(i) + B(i)^T z$ of X depends linearly on the values of the continuous parent variables Z , while the variance is independent of Z . We allow for the situation where the variance is zero such that deterministic relations between continuous variables can be represented.

The quantitative part of an LCG Bayesian network consists of a conditional probability distribution for each $X \in \mathcal{X}_\Delta$ and a conditional Gaussian distribution for each $X \in \mathcal{X}_\Gamma$. For each $X \in \mathcal{X}_\Gamma$ with discrete parents, I , and continuous parents, Z , we need to specify a one-dimensional Gaussian probability distribution for each configuration i of I as shown in (3.2).

Definition 3 An LCG Bayesian network $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P}, \mathcal{F})$ consists of

- A DAG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with nodes \mathcal{V} and directed links \mathcal{E} .
- A set of random variables, \mathcal{X} , represented by the nodes of \mathcal{G} .
- A set of conditional probability distributions, \mathcal{P} , containing one distribution, $P(X_v | X_{\text{pa}(v)})$, for each discrete random variable X_v .
- A set of conditional-linear Gaussian probability density functions, \mathcal{F} , containing one density function, $p(Y_w | X_{\text{pa}(w)})$, for each continuous random variable Y_w .

The joint distribution over all the variables in an LCG Bayesian network has the form $P(\mathcal{X}_\Delta = i) * \mathbb{N}_{|\mathcal{X}_\Gamma|}(\mu(i), \sigma^2(i))$, where $\mathbb{N}_k(\mu, \sigma^2)$ denotes a k -dimensional Gaussian distribution. The chain rule of LCG Bayesian networks is

$$P(\mathcal{X}_\Delta = i) * \mathbb{N}_{|\mathcal{X}_\Gamma|}(\mu(i), \sigma^2(i)) = \prod_{v \in \mathcal{V}_\Delta} P(i_v | i_{\text{pa}(v)}) * \prod_{w \in \mathcal{V}_\Gamma} p(y_w | X_{\text{pa}(w)}),$$

for each configuration \mathbf{i} of \mathcal{X}_Δ .

In the graphical representation of an LCG Bayesian network, continuous variables are represented by double ovals.

Example 30 Figure 3.3 shows an example of the qualitative specification of an LCG Bayesian network, \mathcal{N} , with three variables, i.e., $\mathcal{X} = \{X_1, X_2, X_3\}$, where $\mathcal{X}_\Delta = \{X_1\}$ and $\mathcal{X}_\Gamma = \{X_2, X_3\}$. Hence, \mathcal{N} consists of a continuous random variable X_3 having one discrete random variable X_1 (binary with states F and T) and one continuous random variable X_2 as parents.

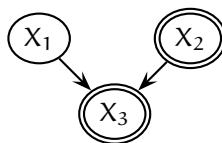


Figure 3.3: LCG Bayesian network with X_1 discrete, and X_2 and X_3 continuous.

To complete the model, we need to specify the relevant conditional probability distribution and density functions. The quantitative specification could, for instance, consist of the following linear conditional Gaussian distribution functions for X_3

$$\begin{aligned}\mathcal{L}(X_3 | \text{F}, x_2) &= \mathbb{N}(-5 + (-2 * x_2), 1.1) \\ \mathcal{L}(X_3 | \text{T}, x_2) &= \mathbb{N}(5 + (2 * x_2), 1.2).\end{aligned}$$

The quantitative specification is completed by letting X_2 have a standard Normal distribution (i.e., $X_2 \sim \mathbb{N}(0, 1)$) and $P(X_1) = (0.75, 0.25)$.

The qualitative and quantitative specifications complete the specification of \mathcal{N} . The joint distribution induced by \mathcal{N} is

$$\begin{aligned}P(X_1 = \text{F}) * p(X_2, X_3) &= 0.75 * \mathbb{N}\left(\begin{pmatrix} 0 \\ -5 \end{pmatrix}, \begin{pmatrix} 1 & 10 \\ 10 & 5.1 \end{pmatrix}\right), \\ P(X_1 = \text{T}) * p(X_2, X_3) &= 0.25 * \mathbb{N}\left(\begin{pmatrix} 0 \\ 5 \end{pmatrix}, \begin{pmatrix} 1 & 10 \\ 10 & 5.2 \end{pmatrix}\right).\end{aligned}$$

■

Determining the joint distribution induced by \mathcal{N} requires a series of nontrivial computations. We refer the reader to the next chapter for a brief treatment of inference in LCG Bayesian networks. A detailed treatment of these computations is beyond the scope of this book.

Example 31 (Adapted from Lauritzen, S. L. (1992)) Consider a banker who is monitoring her clients in order to limit future loss from each client account. The task of the banker is to identify clients who may have problems repaying

their loans by predicting potential future loss originating from each individual customer based on demographic information and credit limit.

Figure 3.4 shows a simple LCG Bayesian network model for this scenario. *Loss* is a linear function of variables *Income* (*I*) given variable *WillToPay* (*W*). *CreditLimit* (*C*) is a linear function of *Income* given *Housing* (*H*) and *MaritalStatus* (*M*). In addition *MaritalStatus* is also a causal factor of *Housing* and *WillToPay*, while *Profession* and *Employment* are causal factors of *Income*.

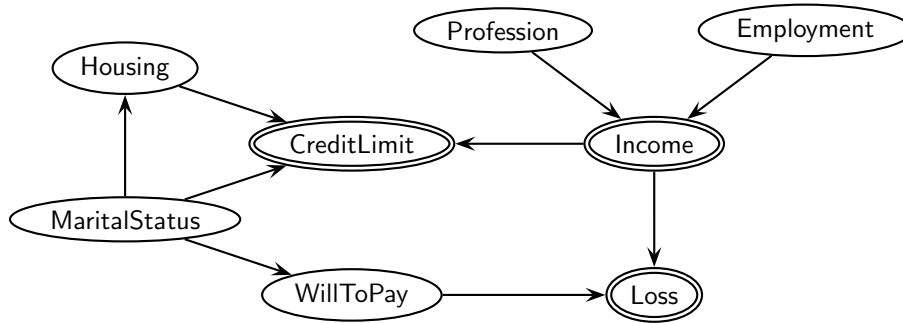


Figure 3.4: LCG Bayesian network for credit account management.

With the model, the banker may enter observations on each client and compute an expected loss for that client. The model may be extended to include various risk indicators and controls in order to facilitate a scenario-based analysis on each client. ■

The reason for restricting our attention to the case of linear conditional Gaussian distributions is that only for this case is exact probabilistic inference feasible by local computations. For most other cases it is necessary to resort to approximate reasoning.

3.2 Decision Making Under Uncertainty

The framework of influence diagrams (Howard & Matheson 1981) is an effective modeling framework for representation and analysis of (Bayesian) decision making under uncertainty. Influence diagrams provide a natural representation for capturing the semantics of decision making with a minimum of clutter and confusion for the decision maker (Shachter & Peot 1992). Solving a decision problem amounts to (i) determining an optimal strategy that maximizes the expected utility for the decision maker and (ii) computing the maximal expected utility of adhering to this strategy.

An influence diagram is a type of causal model that differs from a Bayesian network. A Bayesian network is a model for reasoning under uncertainty, whereas an influence diagram is a probabilistic network for reasoning about

decision making under uncertainty. An influence diagram is a graphical representation of a decision problem involving a sequence of interleaved decisions and observations. Similar to Bayesian networks, an influence diagram is a compact and intuitive probabilistic knowledge representation (a probabilistic network). It consists of a graphical representation describing dependence relations between entities of a problem domain, points in time where a decision is to be made, and a precedence ordering specifying the order on decisions and observations. It also consists of a quantification of the strengths of the dependence relations and the preferences of the decision maker. As such, an influence diagram can be considered as a Bayesian network augmented with decision variables, utility functions specifying the preferences of the decision maker, and a precedence ordering.

As decision makers we are interested in making the best possible decisions given our model of the problem domain. Therefore, we associate utilities with state configurations of the network. These utilities are represented by *utility functions* (also known as *value functions*). Each utility function associates a utility value with each configuration of its domain variables. The objective of decision analysis is to identify the decision options that produce the highest expected utility.

By making decisions, we influence the probabilities of the configurations of the network. To identify the decision option with the highest expected utility, we compute the expected utility of each decision alternative. If A is a decision variable with options a_1, \dots, a_m , H is a hypothesis with states h_1, \dots, h_n , and ε is a set of observations in the form of evidence, then we can compute the utility of each outcome of the hypothesis and the expected utility of each action. The utility of an outcome (a_i, h_j) is $U(a_i, h_j)$ where $U(\cdot)$ is our utility function. The expected utility of performing action a_i is

$$EU(a_i) = \sum_j U(a_i, h_j)P(h_j | \varepsilon),$$

where $P(\cdot)$ represents our belief in H given ε . The utility function $U(\cdot)$ encodes the preferences of the decision maker on a numerical scale.

We shall choose the alternative with the highest expected utility; this is known as the maximum expected utility principle. Choosing the action, which maximizes the expected utility amounts to selecting an option a^* such that

$$a^* = \arg \max_{a \in A} EU(a).$$

There is an important difference between observations and actions. An observation of an event is passive in the sense that we assume that an observation does not effect the state of the world whereas the decision on an action is active in the sense that an action enforces a certain event. The event enforced by a decision may or may not be included in the model depending on whether or not the event is relevant for the reasoning. If the event enforced by an action A is represented in our model, then A is referred to as an intervening action, otherwise it is referred to as a non-intervening action.

3.2.1 Discrete Influence Diagrams

A (discrete) influence diagram $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P}, \mathcal{U})$ is a four-tuple consisting of a set, \mathcal{X} , of discrete random variables and discrete decision variables, an acyclic, directed graph \mathcal{G} , a set of conditional probability distributions \mathcal{P} , and a set of utility functions \mathcal{U} . The acyclic, directed graph, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, contains nodes representing random variables, decision variables, and utility functions (also known as value or utility nodes).

Each decision variable, D , represents a specific point in time under the model of the problem domain where the decision maker has to make a decision. The decision options or alternatives are the states (d_1, \dots, d_n) of D where $n = \|\mathcal{D}\|$. The decision options are mutually exclusive and exhaustive. The usefulness of each decision option is measured by the local utility functions associated with D or one of its descendants in \mathcal{G} . Each local utility function $u(X_{\text{pa}(v)}) \in \mathcal{U}$, where $v \in \mathcal{V}_U$ is a utility node, represents an additive contribution to the total utility function $u(\mathcal{X})$ in \mathcal{N} . Thus, the total utility function is the sum of all the utility functions in the influence diagram, i.e., $u(\mathcal{X}) = \sum_{v \in \mathcal{V}_U} u(X_{\text{pa}(v)})$.

Definition 4 A (discrete) influence diagram $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P}, \mathcal{U})$ consists of

- A DAG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with nodes, \mathcal{V} , and directed links, \mathcal{E} , encoding dependence relations and information precedence including a total order on decisions.
- A set of discrete random variables, \mathcal{X}_C , and discrete decision variables, \mathcal{X}_D , such that $\mathcal{X} = \mathcal{X}_C \cup \mathcal{X}_D$ represented by nodes of \mathcal{G} .
- A set of conditional probability distributions, \mathcal{P} , containing one distribution, $P(X_v | X_{\text{pa}(v)})$, for each discrete random variable X_v .
- A set of utility functions, \mathcal{U} , containing one utility function, $u(X_{\text{pa}(v)})$, for each node v in the subset $\mathcal{V}_U \subset \mathcal{V}$ of utility nodes.

An influence diagram supports the representation and solution of sequential decision problems with multiple local utility functions under the *no-forgetting assumption* (i.e., perfect recall is assumed of all observations and decisions made in the past).

An influence diagram, $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P}, \mathcal{U})$, should be constructed such that one can determine exactly which variables are known prior to making each decision. If the state of a variable $X_v \in \mathcal{X}_C$ will be known at the time of making a decision $D_w \in \mathcal{X}_D$, this will (probably) have an impact on the choice of alternative at D . An observation on X_v made prior to decision D_w is represented in \mathcal{N} by making v a parent of w in \mathcal{G} . If v is a parent of w in $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ (i.e., $(v, w) \in \mathcal{E}$, implying $X_v \in X_{\text{pa}(w)}$), then it is assumed that X_v is observed prior to making the decision represented by D_w . The link (v, w) is then referred to as an *informational link*.

In an influence diagram there must also be a total order on the decision variables $\mathcal{X}_D = \{D_1, \dots, D_n\} \subseteq \mathcal{X}$. That is, there can be only one sequence in

which the decisions are made. We add informational links to specify a total order (D_1, \dots, D_n) on $\mathcal{X}_D = \{D_1, \dots, D_n\}$. There need only be a directed path from one decision variable to the next one in the decision sequence in order to enforce a total order on the decisions.

In short, a link, (w, v) , into a node representing a random variable, X_v , denotes a possible probabilistic dependence relation of X_v on X_w while a link from a node representing a variable, X , into a node representing a decision variable, D , denotes that the state of X is known when decision D is to be made. A link, (w, v) , into a node representing a local utility function, u , denotes functional dependence of u on $X_w \in \mathcal{X}$.

The chain rule of influence diagrams is

$$EU(\mathcal{X}) = \prod_{X_v \in \mathcal{X}_c} P(X_v | X_{pa(v)}) \sum_{w \in \mathcal{V}_u} u(X_{pa(w)}).$$

An influence diagram is a compact representation of a joint expected utility function.

In the graphical representation of an influence diagram, utility functions are represented by rhombuses (diamond shaped nodes), whereas decision variables are represented as rectangles.

Example 32 (Oil Wildcatter (Raiffa 1968)) Consider the fictitious example of an Oil Wildcatter about to decide whether or not to drill for oil at a specific site. The situation of the Oil Wildcatter is the following.

An oil wildcatter must decide either to *drill* or *not to drill*. He is uncertain whether the *hole* will be dry, wet, or soaking. The wildcatter could *take seismic soundings* that will help determine the *geological structure* of the site. The soundings will give a closed reflection pattern (indication of much oil), an open pattern (indication of some oil), or a diffuse pattern (almost no hope of oil).

The qualitative domain knowledge extracted from the above description can be formulated as the DAG shown in Figure 3.5. The state spaces of the variables are as follows $\text{dom}(\text{Drill}) = \{\text{no}, \text{yes}\}$, $\text{dom}(\text{Oil}) = \{\text{dry}, \text{wet}, \text{soaking}\}$, $\text{dom}(\text{Seismic}) = \{\text{closed}, \text{open}, \text{diffuse}\}$, and $\text{dom}(\text{Test}) = \{\text{no}, \text{yes}\}$.

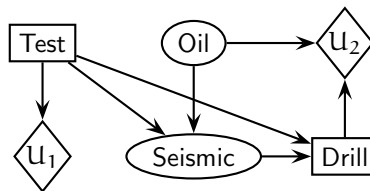


Figure 3.5: The Oil Wildcatter network.

Figure 3.5 shows how the qualitative knowledge of the example can be compactly specified in the structure of an influence diagram $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P}, \mathcal{U})$.

The quantitative probabilistic knowledge as defined by the structure of \mathcal{G} consists of $P(\text{Oil})$ and $P(\text{Seismic}|\text{Oil}, \text{Test})$, while the quantitative utility knowledge consists of $U_1(\text{Test})$ and $U_2(\text{Drill}, \text{Oil})$.

The cost of testing is 10k whereas the cost of drilling is 70k. The utility of drilling is 0k, 120k, and 270k for a dry, wet, and soaking hole, respectively. Hence, $U_1(\text{Test}) = (0, -10)$ and $U_2(\text{Drill} = \text{yes}, \text{Oil}) = (-70, 50, 200)$. The test result *Seismic* depends on the amount of oil *Oil* as specified in Table 3.5. The prior belief of the Oil Wildcatter on the amount of oil at the site is $P(\text{Oil}) = (0.5, 0.3, 0.2)$.

Oil	Seismic		
	diffuse	open	closed
dry	0.6	0.3	0.1
wet	0.3	0.4	0.3
soaking	0.1	0.4	0.5

Table 3.5: The conditional probability distribution $P(\text{Seismic}|\text{Oil}, \text{Test} = \text{yes})$.

This produces a completely specified influence diagram representation of the Oil Wildcatter decision problem. The decision strategy of the Oil Wildcatter will be considered in Example 34 on page 69. ■

As a consequence of the total order on decisions and the set of informational links, the set of discrete random variables and decision variables are subjected to a partial ordering. The random variables are partitioned into disjoint information sets $\mathcal{J}_0, \dots, \mathcal{J}_n$ (i.e., $\mathcal{J}_i \cap \mathcal{J}_j = \emptyset$ for $i \neq j$) relative to the decision variables specifying the precedence order. The partition induces a partial ordering \prec , on the variables \mathcal{X} . The set of variables observed between decisions D_i and D_{i+1} precedes D_{i+1} and succeeds D_i in the ordering

$$\mathcal{J}_0 \prec D_1 \prec \mathcal{J}_1 \prec \dots \prec D_n \prec \mathcal{J}_n,$$

where \mathcal{J}_0 is the set of discrete random variables observed before the first decision, \mathcal{J}_i is the set of discrete random variables observed after making decision D_i and before making decision D_{i+1} , for all $i = 1, \dots, n - 1$, and \mathcal{J}_n is the set of discrete random variables never observed or observed after the last decision D_n has been made. If the influence diagram is not constructed or used according to this constraint, the computed expected utilities will (of course) not be correct.

Example 33 The total order on decisions and the informational links of Example 32 on the facing page induce the following partial order:

$$\{\} \prec \text{Test} \prec \{\text{Seismic}\} \prec \text{Drill} \prec \{\text{Oil}\}.$$

This partial order turns out to be a total order. In general, this is not the case. The total order specifies the flow of information in the decision problem. No

observations are made prior to the decision on whether or not to Test. After testing and before deciding on whether or not to Drill, the oil wildcatter will make an observation on Seismic, i.e. the test result is available before the Drill decision. After drilling Oil is observed. ■

To solve an influence diagram $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P}, \mathcal{U})$ with decision variables, \mathcal{X}_D , is to identify an optimal strategy, $\hat{\Delta}$, over \mathcal{X}_D maximizing the expected utility for the decision maker and to compute the *maximum expected utility* $MEU(\hat{\Delta})$ of $\hat{\Delta}$. A *strategy*, Δ , is an ordered set of decision policies $\Delta = (\delta_1, \dots, \delta_n)$ including one decision policy for each decision $D \in \mathcal{X}_D$. An *optimal strategy* $\hat{\Delta} = (\hat{\delta}_1, \dots, \hat{\delta}_n)$, maximizes the expected utility over all possible strategies, i.e., it satisfies

$$EU(\hat{\Delta}) \geq EU(\Delta),$$

for all strategies Δ .

The *decision history* of D_i , denoted $\mathcal{H}(D_i)$, is the set of previous decisions and their parent variables

$$\mathcal{H}(D_i) = \bigcup_{j=1}^{i-1} (\{D_j\} \cup \mathcal{X}_{\text{pa}(v_j)}) = \{D_1, \dots, D_{i-1}\} \cup \bigcup_{j=0}^{i-2} \mathcal{J}_j,$$

where v_j is denoting the node that represents D_j .

The *decision past* of D_j , denoted $\mathcal{J}(D_i)$, is the set of its parent variables and the decision history $\mathcal{H}(D_i)$

$$\begin{aligned} \mathcal{J}(D_i) &= \mathcal{X}_{\text{pa}(v_i)} \cup \mathcal{H}(D_i) \\ &= \mathcal{X}_{\text{pa}(v_i)} \cup \bigcup_{j=1}^{i-1} (\{D_j\} \cup \mathcal{X}_{\text{pa}(v_j)}) \\ &= \{D_1, \dots, D_{i-1}\} \cup \bigcup_{j=1}^{i-1} \mathcal{J}_j. \end{aligned}$$

Hence, $\mathcal{J}(D_i) \setminus \mathcal{H}(D_i) = \mathcal{J}_{i-1}$ are the variables observed between D_{i-1} and D_i .

The *decision future* of D_i , denoted $\mathcal{F}(D_i)$ is the set of its descendant variables

$$\begin{aligned} \mathcal{F}(D_i) &= \mathcal{J}_i \cup \left(\bigcup_{j=i+1}^n (\{D_j\} \cup \mathcal{X}_{\text{pa}(v_j)}) \right) \\ &= \{D_{i+1}, \dots, D_n\} \cup \bigcup_{j=i}^n \mathcal{J}_j. \end{aligned}$$

A *policy* δ_i is a mapping from the information set $\mathcal{J}(D_i)$ of D_i to the state space $\text{dom}(D_i)$ of D_i such that $\delta_i : \mathcal{J}(D_i) \rightarrow \text{dom}(D_i)$. A policy for decision D specifies the optimal action for the decision maker for all possible observations made prior to making decision D .

It is only necessary to consider δ_i as a function from relevant observations on $\mathcal{J}(D_i)$ to $\text{dom}(D_i)$, i.e., observations with an unblocked path to a utility descendant of D_i . Relevance of an observation with respect to a decision is defined in Section 3.2.3 on page 78.

Example 34 After solving the influence diagram, we obtain an optimal strategy $\hat{\Delta} = \{\hat{\delta}_{\text{Test}}, \hat{\delta}_{\text{Drill}}\}$. Hence, the optimal strategy $\hat{\Delta}$ (we show how to identify the optimal strategy for this example in Example 54 on page 107) consists of a policy $\hat{\delta}_{\text{Test}}$ for Test and a policy $\hat{\delta}_{\text{Drill}}$ for Drill given Test and Seismic

$$\hat{\delta}_{\text{Test}} = \text{yes}$$

$$\hat{\delta}_{\text{Drill}}(\text{Seismic}, \text{Test}) = \begin{cases} \text{yes} & \text{Seismic} = \text{closed}, \text{Test} = \text{no} \\ \text{yes} & \text{Seismic} = \text{open}, \text{Test} = \text{no} \\ \text{yes} & \text{Seismic} = \text{diffuse}, \text{Test} = \text{no} \\ \text{yes} & \text{Seismic} = \text{closed}, \text{Test} = \text{yes} \\ \text{yes} & \text{Seismic} = \text{open}, \text{Test} = \text{yes} \\ \text{no} & \text{Seismic} = \text{diffuse}, \text{Test} = \text{yes} \end{cases}$$

The policy for Test says that we should always test, while the policy for Drill says that we should not drill only when the test produces a diffuse pattern indicating almost no hope of oil. ■

An intervening decision D of an influence diagram is a decision that may impact the value of another variable X represented in the model. In order for D to potentially impact the value of X , X must be a descendant of D in G . This can be realized by considering the d -separation criterion (consider the information blocking properties of the converging connection) and the set of evidence available when making the decision D . Consider, for instance, the influence diagram shown in Figure 3.5. The decision **Test** is an intervening decision as it impacts the value of **Seismic**. It cannot, however, impact the value of **Oil** as **Oil** is a non-descendant of **Test** and we have no *down-stream* evidence when making the decision on **Test**. Since decision D may only have a potential impact on its descendants, the usefulness of D can only be measured by the utility descendants of D .

A total ordering on the decision variables is usually assumed. This assumption can, however, be relaxed. Nielsen & Jensen (1999) describe when decision problems with only a partial ordering on the decision variables are *well-defined*. In addition, the limited memory influence diagram (Lauritzen & Nilsson 2001) and the unconstrained influence diagram (Vomlelová & Jensen 2002) support the use of unordered decision variables.

Example 35 (Apple Jack) We consider once again the problems of Apple Jack from Example 28 on page 56. A Bayesian network for reasoning about the causes of the apple tree loosing its leaves was shown in Figure 3.1 on page 56.

We continue the example by assuming that Apple Jack wants to decide whether or not to invest resources in giving the tree some treatment against a

possible disease. Since this involves a decision through time, we have to extend the Bayesian network to capture the impact of the treatment on the development of the disease. We first add three variables similar to those already in the network. The new variables Sick^* , Dry^* , and Looses^* correspond to the original variables, except that they represent the situation at the time of harvest. These variables have been added in Figure 3.6.

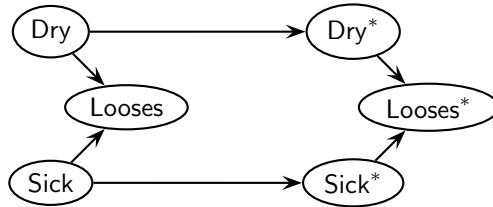


Figure 3.6: We model the system at two different points in time (before and after a decision) by replicating the structure.

The additional variables have the same states as the original variables: Sick^* , Dry^* , and Looses^* all have states *no* and *yes*. In the extended model, we expect a causal influence from the original Sick variable on the Sick^* variable and from the original Dry variable on the Dry^* variable. The reason is the following. If, for example, we expect the tree to be sick now, then this is also very likely to be the case in the future and especially at the time of harvest. Of course, the strength of the influence depends on how far out in the future we look. Perhaps one could also have a causal influence from Looses on Looses^* , but we have chosen not to model such a possible dependence relation in this model.

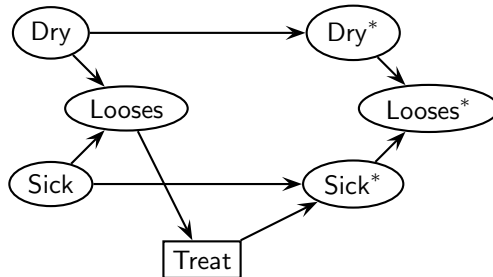


Figure 3.7: Addition of a decision variable for treatment to the Bayesian network in Figure 3.6.

Apple Jack may try to heal the tree with a treatment to get rid of the possible disease. If he expects that the loss of leaves is caused by drought, he might save his money and just wait for rain. The action of giving the tree a treatment is

Treat	Sick	Sick*	
		no	yes
no	no	0.98	0.02
no	yes	0.01	0.99
yes	no	0.99	0.01
yes	yes	0.8	0.2

Table 3.6: The conditional probability distribution $P(\text{Sick}^* | \text{Treat}, \text{Sick})$.

now added as a decision variable to the Bayesian network, which will then no longer be a Bayesian network. Instead it becomes the influence diagram shown in Figure 3.7 on the facing page.

The treat decision variable has the states *no* and *yes*. There is a causal link ($\text{Treat}, \text{Sick}^*$) from the decision *Treat* to Sick^* as we expect the treatment to have a causal impact on the future health of the tree. There is an informational link from *Looses* to *Treat* as we expect Apple Jack to observe whether or not the apple tree is losing its leaves prior to making the decision on treatment.

We need to specify the utility functions enabling us to compute the expected utility of the decision options. This is done by adding utility functions to the influence diagram. Each utility function will represent a term of an additively decomposing utility function and each term will contribute to the total utility. The utility functions are shown in Figure 3.8.

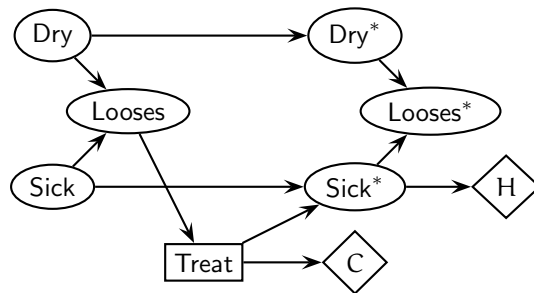


Figure 3.8: A complete qualitative representation of the influence diagram used for decision making in Apple Jack's orchard.

The utility function *C* specifies the cost of the treatment while utility function *H* specifies the gain of the harvest. The latter depends on the state of Sick^* , indicating that the production of apples depends on the health of the tree.

Figure 3.8 shows the complete qualitative representation of the influence diagram $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P}, \mathcal{U})$. To complete the quantitative representation as well, we need to specify the conditional probability distributions, \mathcal{P} , and utility func-

Dry	Dry*	
	no	yes
no	0.95	0.05
yes	0.4	0.6

Table 3.7: The conditional probability distribution $P(\text{Dry}^* | \text{Dry})$.

Dry*	Sick*	Looses*	
		no	yes
no	no	0.98	0.02
no	yes	0.1	0.9
yes	no	0.15	0.85
yes	yes	0.05	0.95

Table 3.8: The conditional probability distribution $P(\text{Looses}^* | \text{Dry}^*, \text{Sick}^*)$.

tions, \mathcal{U} , of \mathcal{N} . Recall that a decision variable does not have any distribution. The appropriate probability distributions are specified in Tables 3.6 – 3.8.

If we have a healthy tree (Sick^* is in state no), then Apple Jack will get an income of EUR 200, while if the tree is sick (Sick^* is in state yes) his income is only EUR 30, i.e., $H(\text{Sick}^*) = (200, 30)$. To treat the tree, he has to spend EUR 80, i.e., $C(\text{Treat}) = (0, -80)$.

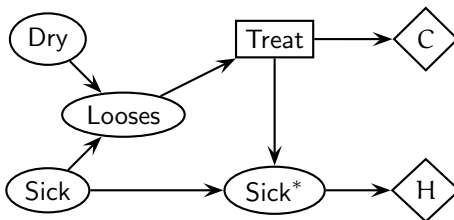


Figure 3.9: A simplified influence diagram for the decision problem of Apple Jack.

Since Dry^* and Looses^* are not relevant for the decision on whether or not to treat and since we do not care about their distribution, we remove them from our model producing the final model shown in Figure 3.9. Variables Dry^* and Looses^* are in fact barren variables, see Section 2.3.4 on page 39. In an influence diagram a variable is a barren variable when none of its descendants are utility nodes and none of its descendants are ever observed.

The purpose of our influence diagram is to be able to determine the optimal strategy for Apple Jack. After solving \mathcal{N} , we obtain the following policy (δ_{Treat} :

Looses \rightarrow dom(Treat) for Treat

$$\delta_{\text{Treat}}(\text{Looses}) = \begin{cases} \text{no} & \text{Looses} = \text{no} \\ \text{yes} & \text{Looses} = \text{yes} \end{cases}$$

Hence, we should only treat the tree when it loses its leaves. In Section 4.2, we describe how to solve an influence diagram. ■

Notice that since a policy is a mapping from all possible observations to decision options, it is sufficient to solve an influence diagram once. Hence, the computed strategy can be used by the decision maker each time she or he is faced with the decision problem.

Implications of Perfect Recall

As mentioned above, using influence diagrams to represent decision problems we assume perfect recall. This assumption states that at the time of any decision, the decision maker remembers all past decisions and all previously known information (as enforced by the informational links). This implies that a decision variable and all of its parent variables are informational parents of all subsequent decision variables. Due to this assumption it is not necessary to include no-forgetting links in the DAG of the influence diagram as they — if missing — will implicitly be assumed present.

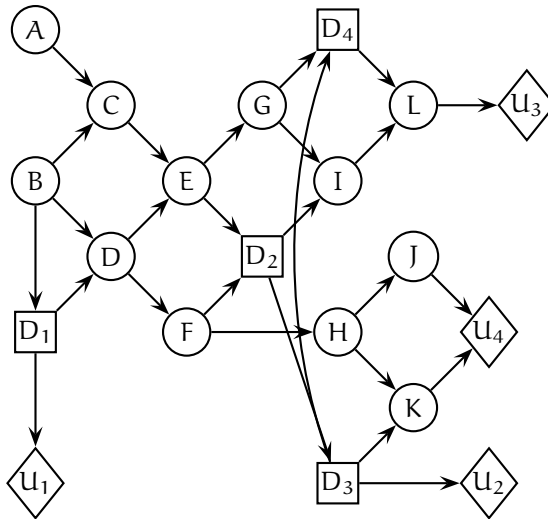


Figure 3.10: An influence diagram representing the sequence of decisions D_1, D_2, D_3, D_4 .

Example 36 (Jensen, Jensen & Dittmer 1994) Let \mathcal{N} be the influence diagram in Figure 3.10 on the page before. This influence diagram represents a decision problem involving four decisions D_1 , D_2 , D_3 , and D_4 in that order.

From the structure of \mathcal{N} , the following partial ordering on the random and decision variables can be read

$$\{B\} \prec D_1 \prec \{E, F\} \prec D_2 \prec \{\} \prec D_3 \prec \{G\} \prec D_4 \prec \{A, C, D, H, I, J, K, L\}.$$

This partial ordering specifies the flow of information in the decision problem represented by \mathcal{N} . Thus, the initial (relevant) information available to the decision maker is an observation of B . After making a decision on D_1 , the decision maker observes E and F . After the observations of E and F a decision on D_2 is made, and so on.

Notice that no-forgetting links have been left out, e.g., there are no links from B to D_2 , D_3 , or D_4 . These links are included in Figure 3.11. The difference in complexity of reading the graph is apparent.

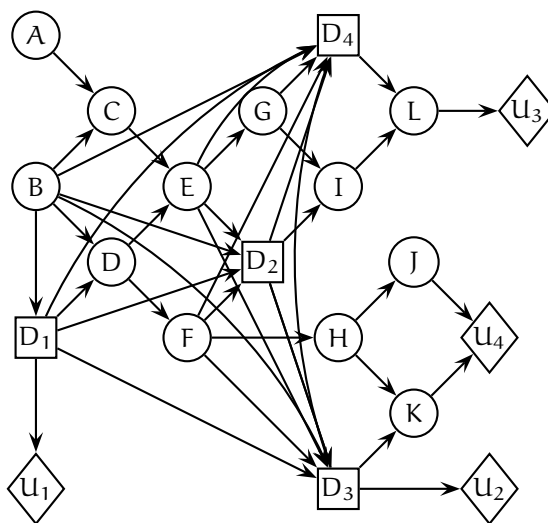


Figure 3.11: The influence diagram of Figure 3.10 on the preceding page with no-forgetting links.

As this example shows a rather informative analysis can be performed by reading only the structure of the graph of \mathcal{N} . ■

3.2.2 Linear-Quadratic CG Influence Diagrams

Linear-quadratic conditional Gaussian influence diagrams combine linear-conditional Gaussian Bayesian networks, discrete influence diagrams, and quadratic

utility functions into a single framework supporting decision making under uncertainty with both continuous and discrete variables (Madsen & Jensen 2005).

Definition 5 An LQCG influence diagram $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P}, \mathcal{F}, \mathcal{U})$ consists of

- A DAG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with nodes, \mathcal{V} , and directed links, \mathcal{E} , encoding dependence relations and information precedence including a total order on decisions.
- A set of random variables, \mathcal{X}_C , and decision variables, \mathcal{X}_D , such that $\mathcal{X} = \mathcal{X}_C \cup \mathcal{X}_D$ represented by nodes of \mathcal{G} .
- A set of conditional probability distributions, \mathcal{P} , containing one distribution, $P(X_v | X_{\text{pa}(v)})$, for each discrete random variable X_v .
- A set of linear-conditional Gaussian probability density functions, \mathcal{F} , containing one density function, $p(Y_w | X_{\text{pa}(w)})$, for each continuous random variable Y_w .
- A set of linear-quadratic utility functions, \mathcal{U} , containing one utility function, $u(X_{\text{pa}(v)})$, for each node v in the subset $\mathcal{V}_U \subset \mathcal{V}$ of utility nodes.

We refer to a linear-quadratic conditional Gaussian influence diagram as an LQCG influence diagram. The chain rule of LQCG influence diagrams is

$$\begin{aligned} EU(\mathcal{X}_\Delta = \mathbf{i}, \mathcal{X}_\Gamma) &= P(\mathcal{X}_\Delta = \mathbf{i}) * \mathbb{N}_{|\mathcal{X}_\Gamma|}(\boldsymbol{\mu}(\mathbf{i}), \boldsymbol{\sigma}^2(\mathbf{i})) * \sum_{z \in \mathcal{V}_U} u(\mathcal{X}_{\text{pa}(z)}) \\ &= \prod_{v \in \mathcal{V}_\Delta} P(i_v | \mathbf{i}_{\text{pa}(v)}) * \prod_{w \in \mathcal{V}_\Gamma} p(y_w | \mathcal{X}_{\text{pa}(w)}) * \sum_{z \in \mathcal{V}_U} u(\mathcal{X}_{\text{pa}(z)}), \end{aligned}$$

for each configuration \mathbf{i} of \mathcal{X}_Δ .

In the graphical representation of an LQCG influence diagram, continuous utility functions are represented by double rhombuses and continuous decision variables as double rectangles.

An LQCG influence diagram is a compact representation of a joint expected utility function over continuous and discrete variables, where continuous variables are assumed to follow a linear Gaussian distribution conditional on a subset of discrete variables while utility functions are assumed to be linear-quadratic in the continuous variables (and constant in the discrete). This may seem as a severe assumption which could be limiting to the usefulness of the LQCG influence diagram. The assumption seems to indicate that all local utility functions specified in an LQCG influence diagram should be linear-quadratic in the continuous variables. This is not the case, however, as the following examples show. We will consider the assumption in more detail in Section 4.2 on solving decision models.

Example 37 (Guessing Game (Madsen & Jensen 2005)) Figure 3.12 on the next page illustrates an LQCG influence diagram, \mathcal{N} , representation of a simple guessing game with two decisions.

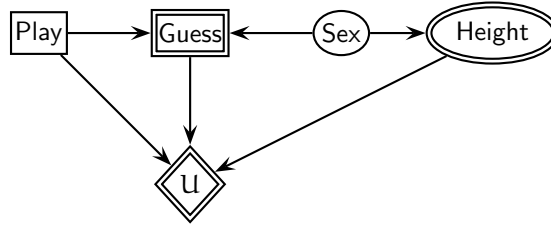


Figure 3.12: An LQCG influence diagram for a simple guessing game.

The first decision, represented by the discrete decision variable Play with states reward and play , is to either accept an immediate reward or to play a game where you will receive a payoff determined by how good you are at guessing the height of a person, represented by the continuous random variable Height , based on knowledge about the sex of the person, represented by the discrete random variable Sex with states female and male . The second decision, represented by the real-valued decision variable Guess , is your guess on the height of the person given knowledge about the sex of the person.

The payoff is a constant (higher than the reward) minus the distance of your guess from the true height of the person measured as height minus guess squared.

To quantify \mathcal{N} , we need to specify a prior probability distribution for Sex , a conditional Gaussian distribution for Height and a utility function over Play , Guess , and Height . Assume the prior distribution on Sex is $P(\text{Sex}) = (0.5, 0.5)$ whereas the distribution for Height is

$$\begin{aligned}\mathcal{L}(\text{Height}|\text{female}) &= \mathcal{N}(170, 400) \\ \mathcal{L}(\text{Height}|\text{male}) &= \mathcal{N}(180, 100).\end{aligned}$$

We assume the average height of a female to be 170 cm with a standard deviation of 20 cm and average height of a male to be 180 cm with a standard deviation of 10 cm. The utility function over Play , Guess , Height is

$$\begin{aligned}u(\text{play}, d_2, h) &= 150 - (h - d_2)^2 \\ u(\text{reward}, d_2, h) &= 100.\end{aligned}$$

We assume the immediate reward is 100. After solving \mathcal{N} , we obtain an optimal strategy $\Delta = \{\delta_{\text{Play}}, \delta_{\text{Guess}}\}$

$$\begin{aligned}\delta_{\text{Play}} &= \text{play} \\ \delta_{\text{Guess}}(\text{play}, \text{female}) &= 170 \\ \delta_{\text{Guess}}(\text{play}, \text{male}) &= 180.\end{aligned}$$

The optimal strategy is to guess that the height of a female person is 170 cm and the height of a male person is 180 cm.

In this example the policy for **Guess** reduces to a constant for each configuration of its parent variables. In the general case, the policy for a continuous decision variable is a multi-linear function in its continuous parent variables given the discrete parent variables. ■

As another example of an LQCG influence diagram consider a revised extension of the Oil Wildcatter problem of Raiffa (1968) (Example 32 on page 66). The revised Oil Wildcatter problem, which is further revised here, is due to Cobb & Shenoy (2004).

Example 38 (Oil Wildcatter (Madsen & Jensen 2005)) The network of the revised version of the Oil Wildcatter problem is shown in Figure 3.13. First, the decision maker makes a decision on whether or not to perform a test **Test** of the geological structure of the site under consideration. When performed, this test will produce a test result, **Seismic** depending on the amount of oil **Oil**. Next, a decision **Drill** on whether or not to drill is made. There is a cost **Cost** associated with drilling, while the revenue is a function of oil volume **Volume** and oil price **Price**.

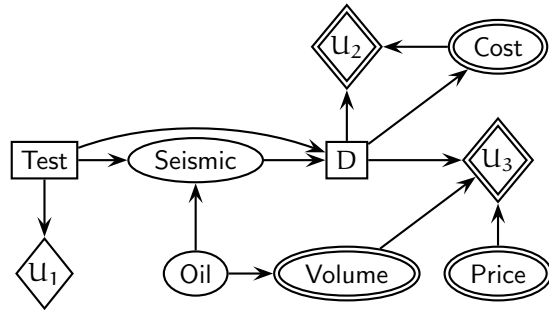


Figure 3.13: A revised version of the Oil Wildcatter problem.

We assume the continuous random variables (i.e., cost of drilling, oil price, and oil volume) to follow (conditional) Gaussian distributions. The utility function can be stated in thousands of EURs as $U_1(\text{Test} = \text{yes}) = -10$, $U_2(\text{Cost} = c, \text{Drill} = \text{yes}) = -c$, $U_3(\text{Volume} = v, \text{Price} = p, \text{Drill} = \text{yes}) = v * p$, and zero for the no drill and no test situations.

If the hole is **dry**, then no oil is extracted: $\mathcal{L}(\text{Volume} | \text{Oil} = \text{dry}) = \mathcal{N}(0, 0)$. If the hole is **wet**, then some oil is extracted: $\mathcal{L}(\text{Volume} | \text{Oil} = \text{wet}) = \mathcal{N}(6, 1)$. If the hole is **soaking** with oil, then a lot of oil is extracted: $\mathcal{L}(\text{Volume} | \text{Oil} = \text{soaking}) = \mathcal{N}(13.5, 4)$. The unit is a thousand barrels. The cost of drilling follows a Gaussian distribution $\mathcal{L}(\text{Cost} | \text{Drill} = \text{yes}) = \mathcal{N}(70, 100)$. We assume that the price of oil **Price** also follows a Gaussian distribution $\mathcal{L}(\text{Price}) = \mathcal{N}(20, 4)$.

Notice that the continuous utility functions U_2 and U_3 are not linear-quadratic in their continuous domain variables. ■

3.2.3 Limited Memory Influence Diagrams

The framework of influence diagrams offers compact and intuitive models for reasoning about decision making under uncertainty. Two of the fundamental assumptions of the influence diagram representation are the no-forgetting assumption implying perfect recall of the past and the assumption of a total order on the decisions. The limited memory influence diagram framework (LIMID) (Lauritzen & Nilsson 2001) relaxes both of these fundamental assumptions.

Relaxing the no-forgetting and the total order (on decisions) assumptions largely increases the class of multistage decision problems that can be modeled. LIMIDs allow us to model more types of decision problems than the ordinary influence diagrams.

The graphical difference between the LIMID representation and the ordinary influence diagram representation is that the latter representation (as presented in this book) assumes some informational links to be implicitly present in the graph. This assumption is not made in the LIMID representation. For this reason it is necessary to explicitly represent all information available to the decision maker at each decision.

The definition of a limited memory influence diagram is as follows.

Definition 6 A LIMID $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P}, \mathcal{U})$ consists of

- A DAG $\mathcal{G} = (V, E)$ with nodes V and directed links E encoding dependence relations and information precedence.
- A set of random variables, \mathcal{X}_C , and discrete decision variables, \mathcal{X}_D , such that $\mathcal{X} = \mathcal{X}_C \cup \mathcal{X}_D$ represented by nodes of \mathcal{G} .
- A set of conditional probability distributions, \mathcal{P} , containing one distribution, $P(X_v | X_{pa(v)})$, for each discrete random variable X_v .
- A set of utility functions, \mathcal{U} , containing one utility function, $u(X_{pa(v)})$, for each node v in the subset $V_U \subset V$ of utility nodes.

Using the LIMID representation it is possible to model multistage decision problems with unordered sequences of decisions and decision problems in which perfect recall cannot be assumed or may not be appropriate. This makes the LIMID framework a good candidate for modeling large and complex domains using appropriate assumption of forgetfulness of the decision maker. Notice that all decision problems that can be represented as an ordinary influence diagram can also be represented as a LIMID.

Example 39 Figure 3.14 on the facing page shows an example of a LIMID representation $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P}, \mathcal{U})$ of a decision scenario with two unordered decisions. Prior to decision D_i observations on the values of A and C are made, while prior to decision D_j an observation on the value of E is made. Notice that the observations on A and C made prior to decision D_i are not available at decision D_j and vice versa for the observation on E . ■

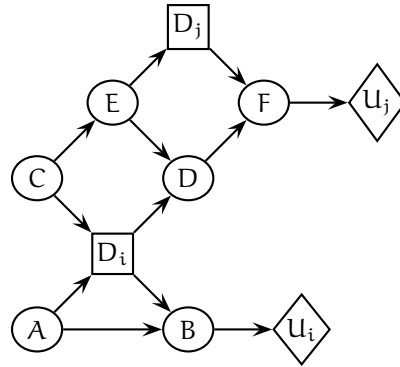


Figure 3.14: A LIMID representation of a decision scenario with two unordered decisions.

Example 40 (Breeding Pigs (Lauritzen & Nilsson 2001)) A pig farmer is growing pigs for a period of four months and subsequently selling them. During this period the pigs may or may not develop a certain disease. If a pig has the disease at the time it must be sold for slaughtering, its expected market price is EUR 40. If it is disease free, its expected market price as a breeding animal is EUR 135.

Once a month, a veterinarian inspects each pig and makes a test for presence of the disease. If a pig is ill, the test will indicate this with probability 0.80, and if the pig is healthy, the test will indicate this with probability 0.90. At each monthly visit, the doctor may or may not treat a pig for the disease by injecting a certain drug. The cost of an injection is EUR 13.

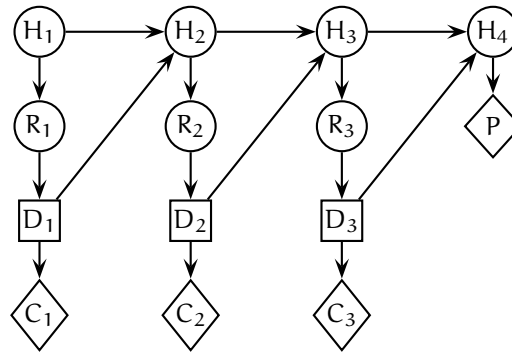


Figure 3.15: Three test-and-treat cycles are performed prior to selling a pig.

A pig has the disease in the first month with probability 0.10. A healthy pig develops the disease in the following month with probability 0.20 without

injection, whereas a healthy and treated pig develops the disease with probability 0.10, so the injection has some preventive effect. An untreated pig that is unhealthy will remain so in the following month with probability 0.90, whereas the similar probability is 0.50 for an unhealthy pig that is treated. Thus, spontaneous cure is possible, but treatment is beneficial on average.

The qualitative structure of the LIMID representation of this decision problem is shown in Figure 3.15 on the page before. Notice that we make the assumption that the test result R_i is only available for decision D_i . This implies that the test result is not taken into account for future decisions as it is either forgotten or ignored. ■

The above example could be modeled as a standard influence diagram, but if more test-and-treat cycles must be performed, the state space size of the past renders decision making intractable. Therefore, it is appropriate to make the decision on whether or not to treat based on the current test result (and not considering past test results and possible treatments) – in this case, individual records for the pigs need not be kept. In short, the example illustrates a situation where instead of keeping track of all past observations and decisions, some of these are deliberately ignored (in order to maintain tractability of the task of computing policies).

3.3 Object-Oriented Probabilistic Networks

As large and complex systems are often composed of collections of identical or similar components, models of such systems will naturally contain repetitive patterns. A complex system will typically be composed of a large number of similar or even identical components. This composition of the system should be reflected in models of the system to support model construction, maintenance, and reconfiguration. For instance, a diagnosis model for diagnosing car start problems could reflect the natural decomposition of a car into its engine, electrical system, fuel system, etc.

To support this approach to model development, the framework of object-oriented probabilistic networks has been developed, see e.g. (Koller & Pfeffer 1997, Laskey & Mahoney 1997, Neil, Fenton & Nielsen 2000). Object-orientation may be defined in the following way

$$\text{object-orientation} = \text{objects} + \text{inheritance},$$

where objects are instances of classes and inheritance defines a relationship between classes. Thus, we need to introduce the notion of objects and classes. In this section, we introduce the notion of *object-oriented probabilistic networks (OOPNs)*.

The basic OOPN mechanisms described below support a type of object-oriented specification of probabilistic networks, which makes it simple to reuse models, to encapsulate sub-models (providing a means for hierarchical model specification), and to perform model construction in a top-down fashion, a

bottom-up fashion, or a mixture of the two (allowing repeated changes of level of abstraction).

An object-oriented modeling paradigm provides support for working with different levels of abstraction in constructing network models. Repeated changes of focus are partly due to the fact that humans naturally think about systems in terms of hierarchies of abstractions and partly due to lack of ability to mentally capture all details of a complex system simultaneously. Specifying a model in a hierarchical fashion often makes the model less cluttered, and thus provides a better means of communicating ideas among knowledge engineers, domain experts, and users.

In the OOPN paradigm we present, an *instance* or *object* has a set of variables and related functions (i.e., probability distributions, probability densities, utility functions, and precedence constraints). This implies that in addition to the usual types of nodes, the graph of an OOPN model may contain nodes representing instances of other networks encapsulated in the model. A node that does not represent an instance of a network class is said to represent a *basic* variable.

An instance represents an instantiation of a network class within another network class. A network class is a blueprint for an instance. As such, a *network class* is a named and self-contained description of a probabilistic network, characterized by its name, interface, and hidden part. As instances can be nested, an object-oriented network can be viewed as a hierarchical description of a problem domain. In this way, an instance \mathcal{M} is the instantiation (or realization) of a network class $C_{\mathcal{M}}$ within another network class $C_{\mathcal{N}}$, see Figure 3.16.

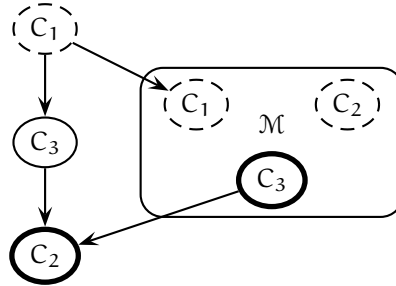


Figure 3.16: \mathcal{M} is an instance of a network class $C_{\mathcal{M}}$ within another network class $C_{\mathcal{N}}$.

An instance connects to other variables via some of its (basic) variables. These variables are known as its *interface* variables. As we wish to support information hiding, the interface variables usually only constitute a subset of the variables in the network class.

Let us be more precise. A network class C is a DAG over three pairwise disjoint sets of nodes $\mathcal{I}(C)$, $\mathcal{H}(C)$, $\mathcal{O}(C)$ where $\mathcal{I}(C)$ are the input nodes, $\mathcal{H}(C)$ are the hidden nodes, and $\mathcal{O}(C)$ are the output nodes of C . The set $\mathcal{I}(C) \cup \mathcal{O}(C)$

is the interface of C . Interface nodes may represent either decision or random variables, whereas hidden nodes may be instances of network classes, decision variables, random variables, and utility functions.

Definition 7 An OOPN network class $C = (\mathcal{N}, \mathcal{I}, \mathcal{O})$ consists of

- A probabilistic network \mathcal{N} over variables \mathcal{X} with DAG \mathcal{G} .
- A set of basic variables $\mathcal{I} \subseteq \mathcal{X}$ specified as input variables and a set of basic variables $\mathcal{O} \subseteq \mathcal{X}$ specified as output variables such that $\mathcal{I} \cap \mathcal{O} = \emptyset$ and $\mathcal{H} = \mathcal{X} \setminus (\mathcal{I} \cup \mathcal{O})$.

In the graphical representation of an OOPN instances are represented as rectangles with arc-shaped corners whereas input variables are represented as dashed ovals and output variables are represented as bold ovals. If the interface variables of a network instance are not shown, then the instance is collapsed. Otherwise it is expanded.

Since an OOPN implements information hiding through encapsulation, we need to be clear on scope rules. First, we define the notations of simple and qualified names. If X is a variable of a network instance \mathcal{N} , then X is the *simple name* of the variable, whereas $\mathcal{N}.X$ is the *qualified name* (also known as the long name) of the variable. The scope $\mathbb{S}(X)$ of a variable X (i.e., a basic variable or an instance) is defined as the part of a model in which the declaration of X can be referred to by its simple name.

The (internal) scope $\mathbb{S}(C)$ of a network class C is the set of variables and instances which can be referred to by their simple names inside C . For instance, the internal scope of the network $C_{\mathcal{N}}$ in Figure 3.16 on the page before is $\mathbb{S}(C_{\mathcal{N}}) = \{C_1, C_3, C_2, \mathcal{M}\}$. The scope of an instance \mathcal{M} of a network class $C_{\mathcal{M}}$, i.e., $\text{class}(\mathcal{M}) = C_{\mathcal{M}}$, is defined in a similar manner.

The interface variables $\mathcal{I}(C) \cup \mathcal{O}(C)$ of C are used to enlarge the visibility of basic variables in the instantiations of C . The visibility of a variable X can be enlarged by specifying it as either an input or an output variable of its class.

An input variable X of an instance \mathcal{M} is a placeholder for a variable (the parent of X) in the encapsulating class of \mathcal{M} . Therefore, an input variable has at most one parent. An output variable X of an instance \mathcal{M} , on the other hand, enlarges the visibility of X to include the encapsulating network class of \mathcal{M} .

Notice that the scope of a variable is distinct from visibility of the variable. In Figure 3.16 on the preceding page, the scope of output variable C_3 is \mathcal{M} whereas its visibility is enlarged to include \mathcal{N} by defining it as an output variable of \mathcal{M} .

An input variable I of an instance \mathcal{M} of network class C is *bound* if it has a parent X in the network class encapsulating \mathcal{M} . Each input random variable I of a class C is assigned a default prior probability distribution $P(I)$, which becomes the probability distribution of the variable I in all instances of C where I is an unbound input variable. A link into a node representing an input variable may be referred to as a *binding link*.

Let \mathcal{M} be an instance of network class C . Each input variable $I \in \mathcal{I}(C)$ has no parent in C , no children outside C , and the corresponding variable of \mathcal{M} has at

most one parent in the encapsulating class of \mathcal{M} . Each output variable $O \in \mathcal{O}(C)$ may only have parents in $\mathcal{I}(C) \cup \mathcal{H}(C)$. The children and parents of $H \in \mathcal{H}(C)$ are subsets of the variables of C .

Example 41 Figure 3.16 on page 81 shows a class instance \mathcal{M} of a network class $C_{\mathcal{M}}$ instantiated within another network class $C_{\mathcal{N}}$. Network class $C_{\mathcal{N}}$ has input variable C_1 , hidden variables C_3 and \mathcal{M} , and output variable C_2 . The network class $C_{\mathcal{M}}$ has input variables C_1 and C_2 , output variable C_3 , and unknown hidden variables. The input variable C_1 of instance \mathcal{M} is bound to C_1 of $C_{\mathcal{N}}$ whereas C_2 is unbound.

Since $C_1 \in \mathcal{I}(C_{\mathcal{N}})$ is bound to $C_1 \in \mathcal{I}(\mathcal{M})$, the visibility of $C_1 \in \mathcal{I}(C_{\mathcal{N}})$ is extended to include the internal scope of \mathcal{M} . Hence, when we refer to $C_1 \in \mathcal{I}(C_{\mathcal{M}})$ inside $C_{\mathcal{M}}$, we are in fact referring to $C_1 \in \mathcal{I}(C_{\mathcal{N}})$ as $C_1 \in \mathcal{I}(C_{\mathcal{M}})$ in instance \mathcal{M} is a placeholder for $C_1 \in \mathcal{I}(C_{\mathcal{N}})$ (i.e., you may think of $C_1 \in \mathcal{I}(C_{\mathcal{M}})$ as the formal parameter of $C_{\mathcal{M}}$ and $C_1 \in \mathcal{I}(C_{\mathcal{N}})$ as the actual parameter of \mathcal{M}). ■

Since an input variable $I \in \mathcal{I}(\mathcal{M})$ of an instance \mathcal{M} is a placeholder for a variable Y in the internal scope of the encapsulating instance of \mathcal{M} , type checking becomes important when the variable Y is bound to I . The variable I enlarges the visibility of Y to include the internal scope of \mathcal{M} and it should therefore be equivalent to Y . We define two variables Y and X to be equivalent as follows.

Definition 8 Two variables X and Y are *equivalent* if and only if they are of the same kind, category, and subtype with the same state labels in the case of discrete variables.

This approach to type checking is referred as *strong type checking*.

If a model contains a lot of repetitive structure, its construction may be tiresome and the resulting model may even be rather cluttered. Both issues are solved when using object-oriented models. Another key feature of object-oriented models is modularity. Modularity allows knowledge engineers to work on different parts of the model independently once an appropriate interface has been defined. The following example will illustrate this point.

Example 42 (Apple Jack's Garden) Let us assume that Apple Jack from Example 28 on page 56 has a garden of three apple trees (including his finest apple tree). He may want to reason about the sickness of each tree given observations on whether or not some of the trees in the garden are losing their leaves.

Figure 3.17 shows the Apple Tree network class. The prior of each tree being sick will be the same while the dryness of a tree is caused by a drought. The drought is an input variable of the Apple Tree network class. If there is a drought this will impact the dryness of all trees. The prior on drought is $P(\text{Drought}) = (0.9, 0.1)$ while the conditional distribution of Dry conditional on Drought is shown in Table 3.9 on the next page.

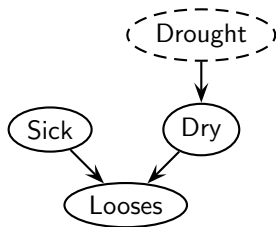


Figure 3.17: The Apple Tree network class.

Drought	Dry	
	no	yes
no	0.85	0.15
yes	0.35	0.65

Table 3.9: The conditional probability distribution $P(\text{Drought}|\text{Dry})$.

Figure 3.18 shows the network class of the Apple Garden. The input variable **Drought** of each of the instances of the Apple Tree network class is bound to the **Drought** variable in the Apple Garden network class. This enlarges the visibility of the **Drought** variable (in the Apple Garden network class) to the internal scope defined by each instance.

The two instances Tree_1 and Tree_2 are collapsed (i.e., not showing the interface variables) while the instance Tree_3 is expanded (i.e., not collapsed) illustrating the interface of the network class.

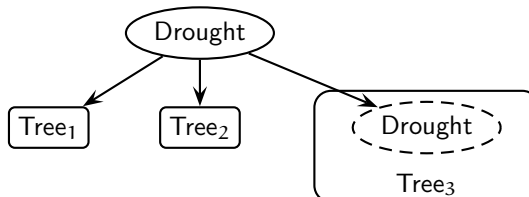


Figure 3.18: The Apple Garden network consisting of three instantiations of the Apple Tree network.

The **Drought** variable could be an input variable of the Apple Garden network class as well as it is determined by other complex factors. For the sake of simplicity of the example, we have made it a hidden variable of the Apple Garden network class. ■

As mentioned above, a default prior distribution $P(X)$ is assigned to each input variable $X \in \mathcal{I}(C)$ of the class $C = (\mathcal{N}, \mathcal{O}, \mathcal{I})$. Assigning a default potential to each input variable X implies that any network class is a valid probabilistic network model.

3.3.1 Chain Rule

It should be clear from the above discussion that each OOPN encodes either a probability distribution or an expected utility function. For simplicity we will discuss only the chain rule for object-oriented (discrete) Bayesian networks. The chain rule of an object-oriented Bayesian network reflects the hierarchical structure of the model.

An instance \mathcal{M} of network class C encapsulates a conditional probability distribution over its random variables given its unbound input nodes. For further simplicity, let $C = (\mathcal{N}, \mathcal{I}, \mathcal{O})$ be a network class over basic discrete random variables only (i.e., no instances, no decisions, and no utilities) with $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P})$ where $X \in \mathcal{X}$ is the only input variable, i.e., $X \in \mathcal{I}$ and $|\mathcal{I}| = 1$. Since X has a default prior distribution, \mathcal{N} is a valid model representing the joint probability distribution

$$P(\mathcal{X}) = P(X) \prod_{Y_v \neq X} P(Y_v | X_{\text{pa}(v)}).$$

In general, an instance \mathcal{M} is a representation of the conditional probability distribution $P(\mathcal{O} | \mathcal{I}')$ where $\mathcal{I}' \subseteq \mathcal{I}$ is the subset of bound input variables of \mathcal{M}

$$P(\mathcal{O} | \mathcal{I}') = \prod_{X \in \mathcal{I} \setminus \mathcal{I}'} P(X) \prod_{Y_v \notin \mathcal{I}} P(Y_v | X_{\text{pa}(v)}).$$

3.3.2 Unfolded OOPNs

An object-oriented network \mathcal{N} has an equivalent *flat* or *unfolded* network model representation \mathcal{M} . The unfolded network model of an object-oriented network \mathcal{N} is obtained by recursively unfolding the instance nodes of \mathcal{N} . The unfolded network representation of a network class is important as it is the structure used for inference.

The joint distribution of an object-oriented Bayesian network model is equivalent to the joint distribution of its unfolded network model

$$P(\mathcal{X}) = \prod_{X_v \in \mathcal{X}_{\mathcal{M}}} P(X_v | X_{\text{pa}(v)}),$$

where $\mathcal{M} = (\mathcal{X}, \mathcal{G}, \mathcal{P})$ is the unfolded network.

3.3.3 Instance Trees

An object-oriented model is a hierarchical model representation. The *instance tree* T of an object-oriented model \mathcal{N} is a tree over the set of instances of classes

in \mathcal{N} . Two nodes v_i and v_j in T (with v_i closer to the root of T than v_j) are connected by an undirected link if and only if the instance represented by v_i contains the instance represented by v_j . The root of an instance tree is the top level network class not instantiated in any other network class within the model. Notice that an instance tree is unique.

In addition to the notion of default potentials there is the notion of the *default instance*. Let C be a network class with instance tree T . Each non-root node v of T represents an instance of a class C_v whereas the root node r of T represents an instance of the unique class C_r , which has not been instantiated in any class. This instance is referred to as the default instance of C_r .

Example 43 Figure 3.19 shows the instance tree of a network class \mathcal{N} where the root is the default instance of \mathcal{N} .

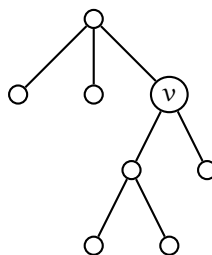


Figure 3.19: An instance tree.

Each node v of T represents an instance \mathcal{M} and the children of v in T represents instances in \mathcal{M} . ■

3.3.4 Inheritance

Another important concept of the OOPN framework is inheritance. For simplicity, we define inheritance as the ability of an instance to take its interface definition from another instance. Let C_1 be a network class with input variables $I(C_1)$ and output variables $O(C_1)$, i.e., $C_1 = (\mathcal{N}_1, \mathcal{I}_1, \mathcal{O}_1)$. A network class $C_2 = (\mathcal{N}_2, \mathcal{I}_2, \mathcal{O}_2)$ may be specified as a subclass of C_1 if and only if $\mathcal{I}_1 \subseteq \mathcal{I}_2$ and $\mathcal{O}_1 \subseteq \mathcal{O}_2$. Hence, subclasses may enlarge the interface.

3.4 Summary

In this chapter we have introduced probabilistic networks for reasoning and decision making under uncertainty. A probabilistic network represents and processes probabilistic knowledge. The qualitative component of a probabilistic network encodes a set of (conditional) dependence and independence statements among a set of random variables, informational precedence, and preference relations

whereas its quantitative component. The quantitative component specifies the strengths of dependence relations using probability theory and preference relations using utility theory.

We have introduced discrete Bayesian network and LCG Bayesian network models for reasoning under uncertainty. A discrete Bayesian network supports the use of discrete random variables whereas a LCG Bayesian network supports the use of a mixture of continuous and discrete random variables. The continuous variables are constrained to be linear-conditional Gaussian variables. The chapter contains a number of examples that illustrates the use of Bayesian networks for reasoning under uncertainty.

Discrete influence diagrams, LQCG influence diagrams, and limited memory influence diagrams were introduced as models for reasoning and decision making under uncertainty. An influence diagram is a Bayesian network augmented with decision variables, informational precedence relations, and preference relations. A discrete influence diagram supports the use of discrete random and decision variables with an additively decomposing utility function. An LQCG influence diagram supports the use of a mixture of continuous and discrete variables. The continuous random variables are constrained to be linear-conditional Gaussian variables while the utility function is constrained to be linear-quadratic. A limited memory influence diagram is an extension of the discrete influence diagram where the assumptions of no-forgetting and a total order on the decisions are relaxed. This allows us to model a large set of decision problems that cannot be modeled using the traditional influence diagram representation. The chapter contains a number of examples that illustrates the use of influence diagrams for decision making under uncertainty.

Finally, we have introduced OOPNs. The basic OOPN mechanisms introduced support a type of object-oriented specification of probabilistic networks, which makes it simple to reuse models, to encapsulate sub-models, and to perform model construction at different levels of abstraction. The chapter contains a number of examples that illustrates the use of the basic OOPN mechanisms in the model development process.

In Chapter 4 we discuss techniques for solving probabilistic networks.

Chapter 4

Solving Probabilistic Networks

We build knowledge bases in order to formulate our knowledge about a certain problem domain in a structured way. The purpose of the knowledge base is to support our reasoning about events and decisions in a domain with inherent uncertainty. The fundamental idea of solving a probabilistic network is to exploit the structure of the knowledge base to reason efficiently about the events and decisions of the domain taking the inherent uncertainty into account.

An expert system consists of a knowledge base and an inference engine. The inference engine is used to solve queries against the knowledge base. In the case of probabilistic networks, we have a clear distinction between the knowledge base and the inference engine. The knowledge base is the Bayesian network or influence diagram, whereas the inference engine is a set of generic methods that applies the knowledge formulated in the knowledge base on task-specific data sets, known as evidence, to compute solutions to queries against the knowledge base. The knowledge base alone is of limited use if it cannot be applied to update our belief about the state of the world or to identify (optimal) decisions in the light of new knowledge.

As we saw in the previous chapter, the knowledge bases we consider are probabilistic networks. A probabilistic network may be an efficient representation of a joint probability distribution or a joint expected utility function. In the former case the model is a Bayesian network, while in the latter case it is an influence diagram.

In this chapter we consider the process of solving probabilistic networks. As the exact nature of solving a query against a probabilistic network depends on the type of model, the solution process of Bayesian networks and influence diagrams are considered separately in the following sections.

Section 4.1 considers probabilistic inference in Bayesian networks as the task of computing posterior beliefs in the light of evidence. A number of different approaches to inference are considered. We consider variable elimination,

query-based inference, arc-reversal, and message passing in junction trees. The inference process in discrete Bayesian networks is treated in detail, while the inference process in LCG Bayesian networks is outlined. In Section 4.2 we consider the task of solving decision models. Solving a decision model assumes to computing maximum expected utilities. We derive a generic method for solving influence diagrams and LIMIDs.

4.1 Probabilistic Inference

We build Bayesian network models in order to support efficient reasoning under uncertainty in a given domain. Reasoning under uncertainty is the task of computing our updated beliefs in (unobserved) events given observations on other events, i.e., evidence.

4.1.1 Inference in Discrete Bayesian Networks

One particular type of probabilistic inference task in Bayesian networks is the task of computing the posterior marginal of an unobserved variable Y given a (possibly empty) set of evidence ε , i.e., $P(Y|\varepsilon)$. Let $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P})$ be a Bayesian network over the set of discrete random variables $\mathcal{X} = \{X_1, \dots, X_n\}$, and assume that $\varepsilon = \emptyset$. Exploiting the chain rule for Bayesian networks (see e.g. (3.1) on page 55), for variable $Y \in \mathcal{X}$, we may compute

$$\begin{aligned} P(Y) &= \sum_{\mathcal{X} \in \mathcal{X} \setminus \{Y\}} P(\mathcal{X}) \\ &= \sum_{\mathcal{X} \in \mathcal{X} \setminus \{Y\}} \prod_{X_v \in \mathcal{X}} P(X_v | X_{\text{pa}(v)}). \end{aligned} \quad (4.1)$$

This is the prior marginal distribution $P(Y)$ of Y . The prior marginal of all variables may be computed by repetition for each variable.

Example 44 Given the example of Apple Jack (Example 28 on page 56), we may consider the task of computing the prior marginal distribution $P(L)$ over the events that the tree does loose its leaves and that the tree does not loose its leaves. The distribution $P(L)$ may be computed as

$$P(L) = \sum_S \sum_D P(S)P(L|S, D)P(D).$$

Using the quantification of the model specified as part of Example 28, we arrive at the prior distribution $P(L) = (0.82, 0.18)$. Hence, *a priori*, there is an 18% probability that the tree will loose its leaves. ■

The above approach does not incorporate evidence into the inference task. In addition, it is a very inefficient approach for non-trivial Bayesian networks

because the joint distribution $P(\mathcal{X})$ over \mathcal{X} is constructed as an intermediate step and because a lot of calculations are repeated.

As we will see, it is possible to develop a more efficient approach to probabilistic inference by exploiting the independence relations induced by the structure of the DAG and the evidence, and by minimizing the repetition of calculations. Having said that, let us turn to the general case of computing the posterior marginal $P(X|\varepsilon)$ of a variable, X , given evidence ε .

Let $\varepsilon = \{\varepsilon_1, \dots, \varepsilon_m\}$ be a non-empty set of evidence over variables $\mathcal{X}(\varepsilon)$. For a (non-observed) variable $X_j \in \mathcal{X}$ of \mathcal{N} , the task is to compute the posterior probability distribution $P(X_j|\varepsilon)$. This can be done by exploiting the chain rule factorization of the joint probability distribution induced by \mathcal{N} :

$$\begin{aligned} P(X_j|\varepsilon) &= \frac{P(\varepsilon|X_j)P(X_j)}{P(\varepsilon)} = \frac{P(X_j, \varepsilon)}{P(\varepsilon)} \\ &\propto P(X_j, \varepsilon) \\ &= \sum_{\mathcal{Y} \in \mathcal{U} \setminus \{X_j\}} P(\mathcal{X}, \varepsilon) \\ &= \sum_{\mathcal{Y} \in \mathcal{U} \setminus \{X_j\}} \prod_{X_i \in \mathcal{X}} P(X_i | X_{\text{pa}(v_i)}) \mathcal{E}_\varepsilon \\ &= \sum_{\mathcal{Y} \in \mathcal{U} \setminus \{X_j\}} \prod_{X_i \in \mathcal{X}} P(X_i | X_{\text{pa}(v_i)}) \prod_{X \in \mathcal{X}(\varepsilon)} \mathcal{E}_X \end{aligned}$$

for each $X_j \notin \mathcal{X}(\varepsilon)$, where \mathcal{E}_X is the evidence function for $X \in \mathcal{X}(\varepsilon)$ and v_i is the node representing X_i . Notice that

$$L(X_j|\varepsilon) = P(\varepsilon|X_j) = \sum_{\mathcal{Y} \in \mathcal{X} \setminus \{X_j\}} \prod_{i \neq j} P(X_{v_i} | X_{\text{pa}(v_i)}) \prod_{X \in \mathcal{X}(\varepsilon)} \mathcal{E}_X \quad (4.2)$$

is the likelihood function of X_j given ε . Since $P(X_j)$ may be obtained by inference over the empty set of evidence, we can — using Bayes' rule — compute

$$P(X_j|\varepsilon) \propto L(X_j|\varepsilon)P(X_j).$$

The proportionality factor is the normalization constant $\alpha = P(\varepsilon)$, which is easily computed from $P(\mathcal{X}, \varepsilon)$ by summation over \mathcal{X} as $\alpha = \sum_{\mathcal{X}} P(\mathcal{X}, \varepsilon)$.

Example 45 One evening when Apple Jack is making his usual after-dinner walk in the garden, he observes his finest apple tree to be losing its leaves. Given that he knows that this may be an indication of the tree being sick, he starts wondering whether or not the tree is sick.

Apple Jack is interested in the probability of the tree being sick given the observation on the tree losing its leaves

$$\begin{aligned} P(S|\varepsilon) &= \frac{P(S, \varepsilon)}{P(\varepsilon)} \\ &= \frac{\sum_S \sum_D P(S)P(L|S, D)P(D)\mathcal{E}_L}{P(\varepsilon)} \\ &\propto (0.0927, 0.0905), \end{aligned}$$

where $\mathcal{E}_L = (0, 1)$ is the evidence function reflecting the tree losing its leaves. The normalization constant is $\alpha = P(\epsilon) = P(S = \text{no}|\epsilon) + P(S = \text{yes}|\epsilon) = 0.0927 + 0.0905 = 0.1832$. This produces the posterior distribution $P(S|\epsilon) = (0.506, 0.494)$ over the tree losing its leaves. Hence, there is an increased probability that the tree is sick when it has been observed to lose its leaves. The prior distribution on the tree being sick is $P(S) = (0.9, 0.1)$. ■

In general, probabilistic inference is an NP-hard task (Cooper 1990). Even approximate probabilistic inference is NP-hard (Dagum & Luby 1993). For certain classes of Bayesian network models the complexity of probabilistic inference is polynomial or even linear in the number of variables in the network. The complexity is polynomial when the graph of the Bayesian network is a polytree (Kim & Pearl 1983, Pearl 1988) (a directed graph \mathcal{G} is called a *polytree*, if its underlying undirected graph is singly connected), while it is linear when the graph of the Bayesian network is a tree.

The most critical problem related to the efficiency of the inference process is that of finding the optimal order in which to perform the computations. The inference task is, in principle, solved by performing a sequence of multiplications and additions.

Query-Based Inference

One approach to inference is to consider the inference task as the task of computing the posterior distribution of a set of variables. This is referred to as query based inference. We define the notion of a query, Q , against a Bayesian network model \mathcal{N} as follows.

Definition 9 (Query) Let $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P})$ be a Bayesian network model. A query Q is a three-tuple $Q = (\mathcal{N}, \mathcal{T}, \epsilon)$ where $\mathcal{T} \subseteq \mathcal{X}$ is the target set and ϵ is the evidence set.

The solution of a query, Q , is the posterior distribution over the target, i.e., $P(\mathcal{T}|\epsilon)$. A variable X is a target variable if $X \in \mathcal{T}$. Notice that computing all posterior marginals of a Bayesian network $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P})$ corresponds to solving $|\mathcal{X}|$ queries, i.e., $Q = (\mathcal{N}, \{X\}, \epsilon)$ for each $X \in \mathcal{X}$.

Prior to solving the query Q , the graph \mathcal{G} of \mathcal{N} may be pruned to include only variables relevant for the query. One class of variables which may be pruned from the graph without any computation is the class of barren variables, see Section 2.3.4 on page 39 for an example. Here we give a formal definition of a barren variable.

Definition 10 (Barren Variable) Let $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P})$ be a Bayesian network and let $Q = (\mathcal{N}, \mathcal{T} \subseteq \mathcal{X}, \epsilon)$ be a query against \mathcal{N} . A variable X is a *barren variable* with respect to Q , if $X \notin \mathcal{T}$, $X \notin \epsilon$, and all descendants, $de(X)$, of X are barren.

When a variable X is classified as a barren variable, it is always relative to a target and given a set of evidence. A barren variable does not add any information to the inference process. It is computationally irrelevant to Q .

Once all barren variables with respect to Q have been pruned from the graph \mathcal{G} , the inference task can be solved by variable elimination as described in the previous section.

In addition to the concept of a barren variable, there is the concept of a *nuisance variable*.

Definition 11 (Nuisance Variable) Let $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P})$ be a Bayesian network and let $Q = (\mathcal{N}, \mathcal{T} \subseteq \mathcal{X}, \varepsilon)$ be a query against \mathcal{N} . A non-barren variable X is a *nuisance variable* with respect to Q , if $X \notin \mathcal{T}$, $X \notin \varepsilon$, and X is not on a path between any pair of variables $Y \in \mathcal{T}$ and $Z \in \varepsilon$.

Notice that a nuisance variable is computationally relevant for a query Q , but it is not on a path between any pair of evidence and query variables. Given a query and a set of evidence variables, the contribution from a nuisance variable does not depend on the observed values of the evidence variables. Hence, if a query is to be solved with respect to multiple instantiations over the evidence variables, then the nuisance variables (and barren variables) may be eliminated in a preprocessing step to obtain the *relevant network* (Lin & Druzdzel 1997). The relevant network consists of target variables, evidence variables, and variables on paths between target and evidence variables only.

Example 46 Returning to the Chest Clinic example (Example 29 on page 58), we may consider the task of computing the probability of each disease given the observations that the patient is a smoker and has a positive X-ray result. That is, we need to compute $P(Y|\varepsilon)$ for $Y \in \{T, L, B\}$ and $\varepsilon = \{S = \text{yes}, X = \text{yes}\}$.

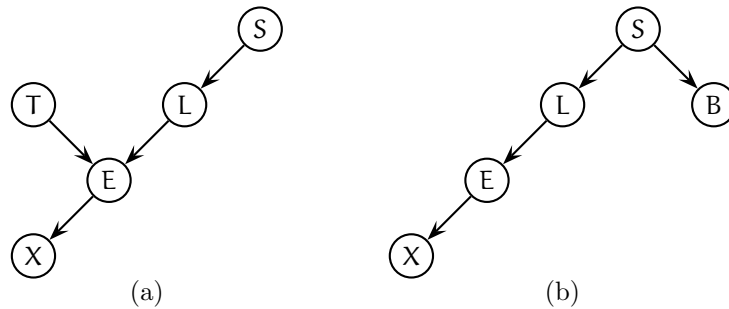


Figure 4.1: The relevant networks for computing (a) $P(T|\varepsilon)$ and $P(L|\varepsilon)$, and (b) $P(B|\varepsilon)$.

The variables $\{A, T\}$ are nuisance variables with respect to posteriors for B and L . The variable D is a barren variable with respect to the posteriors for B , T , and L , whereas B is a barren variable with respect to the posteriors for T and L . Figure 4.1 shows the relevant networks for (a) computing $P(T|\varepsilon)$ and $P(L|\varepsilon)$, and for (b) computing $P(B|\varepsilon)$. ■

The approach to inference outlined above may be referred to as a *direct approach*. Arc-reversal is a specific type of direct approach to inference (Olmsted 1983, Shachter 1986).

Arc Reversal

In Section 2.4.1 on page 42 we illustrated how application of Bayes' rule can be given a graphical interpretation as arc reversal. We mentioned that Olmsted (1983) and Shachter (1986) have exploited this view of inference in their arc reversal algorithms for inference in probabilistic networks. Here we consider the process in more detail.

Let \mathcal{G} be the DAG of a Bayesian network $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P})$ and assume a query $Q = (\mathcal{N}, \{Z\}, \emptyset)$ against \mathcal{N} . The inference task is to compute $P(Z)$ by eliminating all variables $\mathcal{X} \setminus \{Z\}$.

The inference process on \mathcal{G} has a natural graphical interpretation as a sequence of arc reversals and barren variable eliminations. The fundamental idea is to adjust the structure of \mathcal{G} such that all variables except Z are pruned as barren variables while maintaining the underlying properties of the joint probability distributions over the remaining variables. The structure of \mathcal{G} is adjusted through a sequence of arc reversal operations.

Assume X_w is the next variable to be eliminated as a barren variable. Let X_w have parents $X_{pa(w)} = X_i \cup X_j$ and X_v have parents $X_{pa(v)} = \{X_w\} \cup X_j \cup X_k$ where $X_i \cap X_j = X_i \cap X_k = X_j \cap X_k = \emptyset$ such that $X_i = X_{pa(w)} \setminus X_{pa(v)}$ are the parents specific for X_w , $X_j = X_{pa(w)} \cap X_{pa(v)}$ are the common parents, and $X_k = X_{pa(v)} \setminus X_{fa(w)}$ are the parents specific for X_v .

The reversal of arc (w, v) proceeds by setting $X_{pa(w)} = X_i \cup X_j \cup X_k \cup \{X_v\}$ and $X_{pa(v)} = X_i \cup X_j \cup X_k$ as well as performing the computations specified below, see Figure 4.2 for a graphical representation

$$P(X_v | X_i, X_j, X_k) = \sum_{X_w} P(X_w | X_i, X_j) P(X_v | X_w, X_j, X_k) \quad (4.3)$$

$$P(X_w | X_v, X_i, X_j, X_k) = \frac{P(X_w | X_i, X_j) P(X_v | X_w, X_j, X_k)}{P(X_v | X_i, X_j, X_k)} \quad (4.4)$$

The operation of reversing an arc changes the structure of \mathcal{G} without changing the underlying joint probability distribution over \mathcal{X} induced by \mathcal{N} .

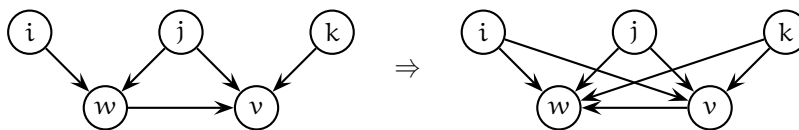


Figure 4.2: An illustration of reversal of the arc (w, v) .

Once the arc (w, v) has been reversed, the variable X_w is a barren variable relative to the other variables (given the empty set of evidence), and can be pruned from \mathcal{G} without further computations.

The basic idea of the inference process known as arc reversal is to perform a sequence of arc reversals and barren variable eliminations on the DAG \mathcal{G} until a desired marginal or conditional is obtained. In this process a valid Bayesian network structure is maintained throughout the inference process.

Example 47 We may compute the prior probability distribution $P(L)$ in the Apple Jack example (see Example 28 on page 56) using a sequence of arc reversals and barren variable eliminations as indicated in Figure 4.3.

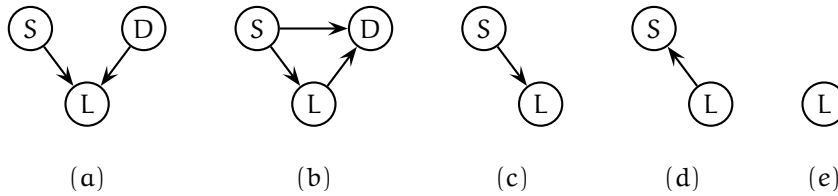


Figure 4.3: Computing $P(L)$ by arc reversal.

Notice that the arc reversal method does not have worse complexity than variable elimination. ■

Arc reversal is not a local computation algorithm in the sense that when reversing an arc (w, v) , it is necessary to test for existence of a directed path from w to v not containing (w, v) . If such a path exists, then the arc (w, v) cannot be reversed until one or more other arcs have been reversed as reversing (w, v) would otherwise create a directed path.

Graphical Representation of Inference

We may define the task of solving a Bayesian network model $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P})$ as the problem of computing the posterior marginal $P(X|\varepsilon)$ given a set of evidence ε for all variables $X \in \mathcal{X}$.

When defining the task of probabilistic inference as the task of computing the posterior marginals $P(X|\varepsilon)$ for all X given evidence ε , the most common approach is to use a secondary computational structure. Performing inference in a secondary computational structure aims at reusing calculations solving all queries simultaneously.

From (4.1) on page 90 we should notice the direct correspondence between the acyclic, directed graph \mathcal{G} and the factorization of the joint probability distribution $P(\mathcal{X})$ over \mathcal{X} . The domain of each factor in the factorization corresponds to a node and its parents. The head of the factor is the child node whereas the tail consists of the parents. Furthermore, if we drop the distinction between head and tail we see that the domain of each factor corresponds to a *clique* (a

clique is a maximal complete subgraph) of \mathcal{G}^m — the moralization of \mathcal{G} . This is exploited to build a secondary structure for performing inference.

Assume we are in the process of computing $P(X_i)$. Let Y be the first random variable to eliminate. The elimination process proceeds by local computation in order to maintain efficiency (i.e., we exploit the distributive law to maintain the factorization of the joint probability distribution — see Section 2.3.3 on page 38). The set of probability potentials \mathcal{P} can be divided into two disjoint subsets with respect to Y . Let $\mathcal{P}_Y \subseteq \mathcal{P}$ be the subset of probability potentials including Y in the domain

$$\mathcal{P}_Y = \{P \in \mathcal{P} | Y \in \text{dom}(P)\},$$

where $\text{dom}(P)$ denotes the domain of P (i.e., the set of variables over which it is defined). Then $\mathcal{P} \setminus \mathcal{P}_Y$ is the set of probability potentials not including Y in their domain. Let ϕ_Y be the probability potential obtained by eliminating Y (by summation) from the combination of all probability potentials in \mathcal{P}_Y . Using ϕ_Y as well as a generalized version of the distributive law, we may rewrite Equation 4.1 on page 90 as

$$\begin{aligned} P(X_i) &= \sum_{X \in \mathcal{X} \setminus \{X_i\}} \prod_{X_v \in \mathcal{X}} P(X_v | X_{\text{pa}(v)}) \\ &= \sum_{X \in \mathcal{X} \setminus \{X_i\}} \prod_{\phi \in \mathcal{P} \setminus \mathcal{P}_Y} \phi \prod_{\phi' \in \mathcal{P}_Y} \phi' \\ &= \sum_{X \in \mathcal{X} \setminus \{X_i, Y\}} \prod_{\phi \in \mathcal{P} \setminus \mathcal{P}_Y} \phi \sum_Y \prod_{\phi' \in \mathcal{P}_Y} \phi' \\ &= \sum_{X \in \mathcal{X} \setminus \{X_i, Y\}} \phi_Y \prod_{\phi \in \mathcal{P} \setminus \mathcal{P}_Y} \phi. \end{aligned} \tag{4.5}$$

Equation 4.5 specifies a decomposition of the joint probability distribution over $\mathcal{X} \setminus \{Y\}$. The decomposition has the form of Equation 4.1. The decomposition is the product over the elements of $\mathcal{P} \setminus \mathcal{P}_Y \cup \{\phi_Y\}$. In addition, we have performed the elimination over Y by local computations only involving potentials of which Y is a domain variable. We say that the set

$$\mathcal{P} \setminus \mathcal{P}_Y \cup \{\phi_Y\}$$

is a reduction of \mathcal{P} where Y has been eliminated. The elimination of the next variable to be eliminated may proceed in the same manner on $\mathcal{P} \setminus \mathcal{P}_Y \cup \{\phi_Y\}$. The order in which variables are eliminated is the *elimination order*.

An example of this process may be depicted graphically as shown in Figure 4.4 on the facing page where we assume $\text{dom}(\phi_Y) = \{X_1, X_2\}$. The arrows in the figure are used to illustrate the flow of computations.

The elimination of Y from $\phi(X_1, X_2, Y)$ creates a potential over $\phi(X_1, X_2)$ which is included in the elimination of the next variable X_1 to be eliminated. In this way the process continues until the desired marginals are obtained. Let us consider an even more concrete example.

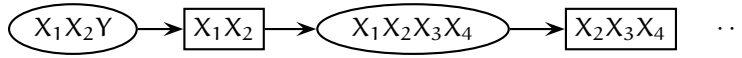


Figure 4.4: A graphical illustration of the process of eliminating Y from $\phi(X_1, X_2, Y)$ and X_1 from $\phi(X_1, X_2, X_3, X_4)$, where the ovals represent the domain of a potential before elimination, and rectangles represent the domain of a potential after elimination.

Example 48 (Burglary or Earthquake on page 10) The Bayesian network shown in Figure 1.6 on page 11, is repeated in Figure 4.5.

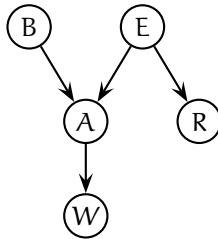


Figure 4.5: The Burglary or Earthquake network.

The prior marginal on A may be computed by elimination of $\{B, E, R, W\}$ as follows

$$P(A) = \sum_E P(E) \sum_B P(B)P(A|B, E) \sum_R P(R|E) \sum_W P(W|A). \quad (4.6)$$

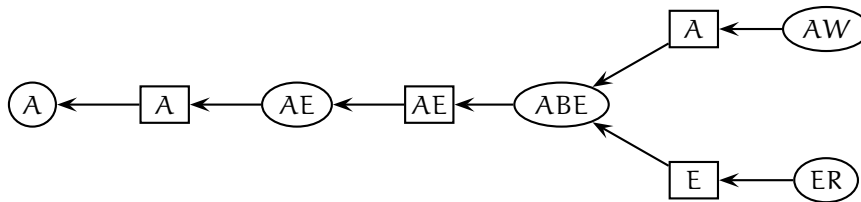


Figure 4.6: A graphical illustration of the process of computing $P(A)$ in (4.6), where the ovals represent the domain of a potential before elimination, and rectangles represent the domain of a potential after elimination.

Figure 4.6 shows a graphical representation of the computations and potentials created the during process of computing $P(A)$.

Similarly, the prior marginal distribution over W may be computed by elimination of $\{A, B, E, R\}$ as follows

$$P(W) = \sum_A P(W|A) \sum_E P(E) \sum_B P(B)P(A|B, E) \sum_R P(R|E). \quad (4.7)$$

Figure 4.7 shows a graphical representation of the computations and potentials created during the process of computing of $P(W)$.

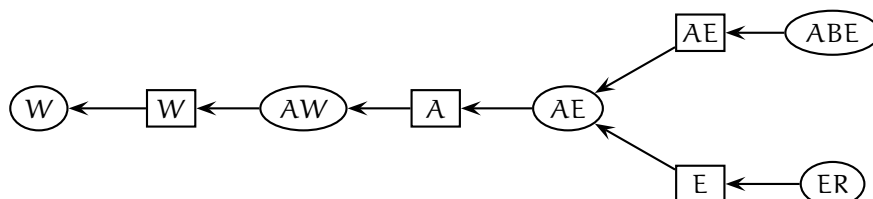


Figure 4.7: A graphical illustration of the process of computing $P(W)$ in (4.7), where the ovals represent the domain of a potential before elimination, and rectangles represent the domain of a potential after elimination.

Notice the similarity between the potentials created in the process of computing $P(A)$ and $P(W)$. There is a significant overlap between the potentials created and therefore the calculations performed. This is no coincidence. ■

Junction Trees

The task of probabilistic inference may be solved efficiently by local procedures operating on a secondary computational structure known as the *junction tree* (also known as a join tree and a Markov tree) representation of a Bayesian network (Jensen & Jensen 1994, Jensen et al. 1994).

The junction tree representation is efficient when solving the inference task for multiple sets of different evidence and target variables. A junction tree representation \mathcal{T} of a Bayesian network $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P})$ is a pair $\mathcal{T} = (\mathcal{C}, \mathcal{S})$ where \mathcal{C} is the set of cliques and \mathcal{S} is the set of separators. The cliques \mathcal{C} are the nodes of \mathcal{T} whereas the separators \mathcal{S} annotate the links of the tree. Each clique $C \in \mathcal{C}$ represents a maximal complete subset of pairwise connected variables of \mathcal{X} , i.e., $C \subseteq \mathcal{X}$, of an undirected graph.¹ The link between two neighboring cliques C_i and C_j is annotated with the intersection $S = C_i \cap C_j$, where $S \in \mathcal{S}$.

Example 49 (Chest Clinic) Figure 3.2 on page 58 shows the DAG \mathcal{G} of the Chest Clinic network $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P})$.

¹The undirected graph is constructed from the moral graph \mathcal{G}^m of \mathcal{G} by adding undirected edges until the graph is triangulated. A graph is triangulated if every cycle of length greater than three has a chord.

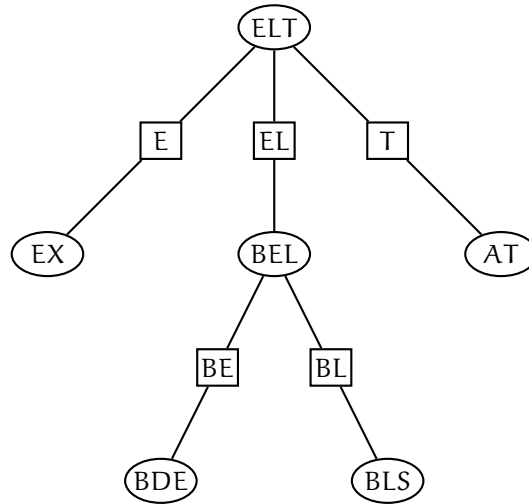


Figure 4.8: A junction tree representation \mathcal{T} for the Chest Clinic network.

Figure 4.8 shows a junction tree representation $\mathcal{T} = (\mathcal{C}, \mathcal{S})$ of the Chest Clinic network. The junction tree consists of cliques

$$\mathcal{C} = \{\{A, T\}, \{B, D, E\}, \{B, E, L\}, \{B, L, S\}, \{E, L, T\}, \{E, X\}\}$$

and separators

$$\mathcal{S} = \{\{B, E\}, \{B, L\}, \{E\}, \{E, L\}, \{T\}\}.$$

The structure of \mathcal{T} is determined from the structure of \mathcal{G} . ■

The process of creating a junction tree representation of a DAG is beyond the scope of this book. Instead we refer the interested reader to the literature, see e.g. Cowell, Dawid, Lauritzen & Spiegelhalter (1999).

The junction tree serves as an excellent control structure for organizing the computations performed during probabilistic inference. Messages are passed between cliques of the junction tree in two sweeps such that a single message is passed between each pair of neighboring cliques in each sweep. This process is referred to as a *propagation of information*.

Once the junction tree $\mathcal{T} = (\mathcal{C}, \mathcal{S})$ has been constructed, a probability potential is associated with each clique $C \in \mathcal{C}$ and each separator $S \in \mathcal{S}$ between two adjacent cliques C_i and C_j where $S = C_i \cap C_j$, see Figure 4.9 on the following page.

Inference involves the following steps:

- (1) Each item of evidence must be incorporated into the junction tree potentials. For each item of evidence, an evidence function is multiplied onto an appropriate clique potential.

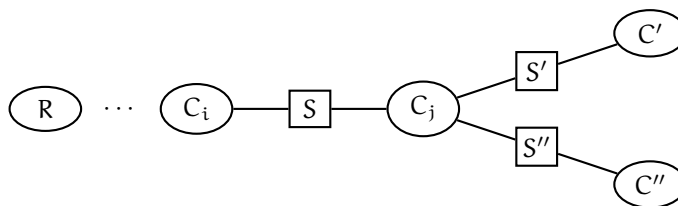


Figure 4.9: When C_j has absorbed information from its other neighbors, C_i can absorb from C_j .

- (2) Some clique $R \in \mathcal{C}$ of \mathcal{T} is selected. This clique is referred to as the *root* of the propagation.
- (3) Then messages are passed toward the selected root. The messages are passed through the separators of the junction tree (i.e., along the links of the tree). These messages cause the potentials of the receiving cliques and separators to be updated. This phase is known as `COLLECTINFORMATION`.
- (4) Now messages are passed in the opposite direction (i.e., from the root toward the leaves of the junction tree). This phase is known as `DISTRIBUTEINFORMATION`.
- (5) At this point, the junction tree is said to be in equilibrium: The probability $P(X|\varepsilon)$ can be computed from any clique or separator containing X — the result will be independent of the chosen clique or separator.

Prior to the initial round of message passing, for each variable $X_v \in \mathcal{X}$ we assign the conditional probability distribution $P(X_v|X_{\text{pa}(v)})$ to a clique C such that $X_{\text{fa}(v)} \subseteq C$. Once all conditional probability distributions have been assigned to cliques, the distributions assigned to each clique are combined to form the initial clique potential.

Example 50 Consider again the junction tree of the Chest Clinic network shown in Figure 4.8 on the page before. Each conditional probability distribution $P \in \mathcal{P}$ is associated with a clique of \mathcal{T} such that $\text{dom}(P) \subseteq C$ for $C \in \mathcal{C}$. Notice that the association of distributions with cliques is unique in this example. ■

The basic inference algorithm is as follows. Each separator holds a single potential over the separator variables, which initially is a unity potential. During propagation of information the separator and clique potentials are updated. Consider two adjacent cliques C_i and C_j as shown in Figure 4.9. When a message is passed from C_j to C_i either during `COLLECTINFORMATION` or `DISTRIBUTEINFORMATION`, C_i absorbs information from C_j . Absorption of information involves performing the following calculations:

- (1) Calculate the updated separator potential:

$$\phi_S^* = \sum_{C_j \setminus S} \phi_{C_j}.$$

- (2) Update the clique potential of C_i :

$$\phi_{C_i} := \phi_{C_i} \frac{\phi_S^*}{\phi_S}.$$

- (3) Associate the updated potential with the separator:

$$\phi_S = \phi_S^*.$$

After a full round of message passing the potential associated with any clique (separator) is the joint probability distribution (up to the same normalization constant) of the variables in the clique (separator) and the evidence. This algorithm is known as the *Hugin algorithm*. Details on the inference process can be found in the literature (Lauritzen & Spiegelhalter 1988, Andersen, Olesen, Jensen & Jensen 1989, Jensen, Lauritzen & Olesen 1990, Dawid 1992, Jensen et al. 1994, Lauritzen & Jensen 2001).

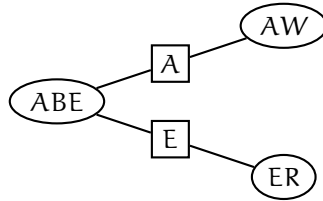


Figure 4.10: A junction tree representation \mathcal{T} of the Bayesian network depicted in Figure 4.5 on page 97.

Example 51 Figure 4.10 shows a junction tree representation $\mathcal{T} = (\mathcal{C}, \mathcal{S})$ of the Bayesian network depicted in Figure 4.5 on page 97 with cliques:

$$\mathcal{C} = \{\{A, B, E\}, \{E, R\}, \{A, W\}\}$$

and separators:

$$\mathcal{S} = \{\{E\}, \{A\}\}.$$

Notice the similarity between Figure 4.10 and Figures 4.6 and 4.7. The nodes of Figures 4.6 and 4.7 are clusters (i.e., subsets of variables) whereas the nodes of Figure 4.10 are cliques (i.e., maximal subsets of pairwise connected variables) of undirected graphs.

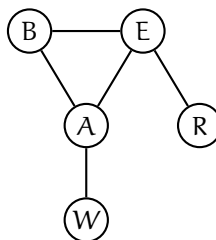
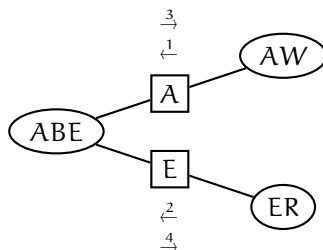


Figure 4.11: The undirected graph corresponding to Figures 4.6, 4.7 and 4.10.

Figure 4.12: Message passing in \mathcal{J} .

The undirected graph corresponding to a junction tree is obtained by adding undirected edges between each pair of variables contained in the same clique or cluster. Figure 4.11 is the undirected graph corresponding to Figures 4.6, 4.7 and 4.10.

Figure 4.12 shows how messages are passed over \mathcal{J} relative to the root ABE.

Underlying any approach to inference is the junction tree representation, although its presence may be implicit. Figure 4.6 shows the cluster tree representation underlying the computation of $P(A)$ whereas Figure 4.7 shows the cluster tree representation underlying the computation of $P(W)$. Figures 4.6 and 4.7 are not junction trees, but cluster trees. The cliques of a junction tree are maximal complete subsets of pairwise connected variables, whereas clusters are not necessarily maximal. ■

The quality of the junction tree $\mathcal{J} = (\mathcal{C}, \mathcal{S})$ determines the efficiency of inference. A common score or criterion to use when considering the optimality of a junction tree is the maximum state space size over all cliques in \mathcal{J} , i.e., $\max_{C \in \mathcal{C}} \|C\|$. Another similar score is the sum over all cliques in \mathcal{J} , i.e., $\sum_{C \in \mathcal{C}} \|C\|$.

All types of probabilistic networks considered in this book may be solved by message passing in a junction tree representation. However, we will restrict ourselves from a detailed treatment of this topic for all models presented as it is beyond the scope of this book.

The approach to inference outlined above may be referred to as an *indirect approach*.

4.1.2 Inference in LCG Bayesian Networks

Let $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P}, \mathcal{F})$ be an LCG Bayesian network with continuous random variables, \mathcal{X}_Γ , and discrete random variables, \mathcal{X}_Δ , such that $\mathcal{X} = \mathcal{X}_\Gamma \cup \mathcal{X}_\Delta$. To solve the probabilistic inference task on \mathcal{N} is to compute the marginal for each $X \in \mathcal{X}$. Since \mathcal{N} is an LCG Bayesian network the task of performing inference becomes more subtle than in the case of a pure discrete Bayesian network.

The prior distribution, $P(X)$, of a discrete variable $X \in \mathcal{X}_\Delta$ is equal to the distribution of X in the discrete network $\mathcal{N}' = (\mathcal{X}_\Delta, \mathcal{P})$ obtained by removing all continuous variables from the model (all continuous variables are barren variables with respect to the joint over the discrete variables). The prior density of a continuous variable Y , on the other hand, will, in general, be a mixture of Gaussian distributions where the mixing factors are joint probabilities over configurations of discrete variables $I \subseteq \mathcal{X}_\Delta$. For each configuration \mathbf{i} of I with non-zero probability, i.e., $p(\mathbf{i}) > 0$, the joint distribution of I and X has the form

$$P(I = \mathbf{i}) * \mathcal{N}(\boldsymbol{\mu}(\mathbf{i}), \boldsymbol{\sigma}^2(\mathbf{i})).$$

This implies that the marginal of $X \in \mathcal{X}_\Gamma$ is

$$\mathcal{L}(X) = \sum_{\mathbf{i}: P(I=\mathbf{i}) > 0} P(\mathbf{i}) * \mathcal{N}(\boldsymbol{\mu}(\mathbf{i}), \boldsymbol{\sigma}^2(\mathbf{i})).$$

For each configuration \mathbf{i} of I with $P(\mathbf{i}) = 0$ the mean $\boldsymbol{\mu}(\mathbf{i})$ and variance $\boldsymbol{\sigma}^2(\mathbf{i})$ may be random numbers. Hence, the marginal density function for a continuous variable $X \in \mathcal{X}_\Gamma$ is, in general, a mixture of Gaussian distributions

$$f(x) = \sum_{\mathbf{i}=0}^n \alpha_{\mathbf{i}} f_{\mathbf{i}}(x),$$

where each component $f_{\mathbf{i}}$ is a one-dimensional Gaussian density function in X and each coefficient $\alpha_{\mathbf{i}}$ is the probability of a configuration of discrete variables. This implies that the marginal density function of $X \in \mathcal{X}_\Gamma$ is not necessarily an LCG distribution with the indicated mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\sigma}^2$. That is, the result of a marginalization of an LCG distribution over continuous variables is an LCG distribution whereas the result of a marginalization of an LCG distribution over discrete variables, in general, is not. The first type of marginal is referred to as a *strong marginal*, whereas the latter is referred to as a *weak marginal*. The marginal is strong as we compute the mean $\boldsymbol{\mu}$ and the variance $\boldsymbol{\sigma}^2$, and we know the distribution is an LCG distribution.

Probabilistic inference is the task of updating our belief about the state of the world in light of evidence. Evidence on discrete variables, be it hard or soft evidence, is treated as in the case of discrete Bayesian networks. Evidence on

a continuous variable, on the other hand, is restricted to be hard evidence, i.e., instantiations.

In the general case where evidence ε is available, the marginal for a discrete variable $X \in \mathcal{X}_\Delta$ is a probability distribution $P(X|\varepsilon)$ conditional on the evidence ε , whereas the marginal for a continuous variable $X \in \mathcal{X}_\Gamma$ is a density function $f(x|\varepsilon)$ conditional on ε with a mean μ and a variance σ^2 .

Example 52 Example 30 on page 62 shows an example of a simple LCG Bayesian network. Computing the prior probability density in X_3 amounts to eliminating the variables X_1 and X_2 . With the quantification specified in Example 30 this produces the following mixture

$$\mathcal{L}(X_3) = 0.75 * \mathbb{N}(-5, 5.1) + 0.25 * \mathbb{N}(5, 5.2)$$

with mean $\mu = -2.5$ and variance $\sigma^2 = 23.88$. Notice that the density for X_3 is not the density for the Gaussian distribution with mean $\mu = -2.5$ and variance $\sigma^2 = 23.88$. The density function is shown in Figure 4.13.

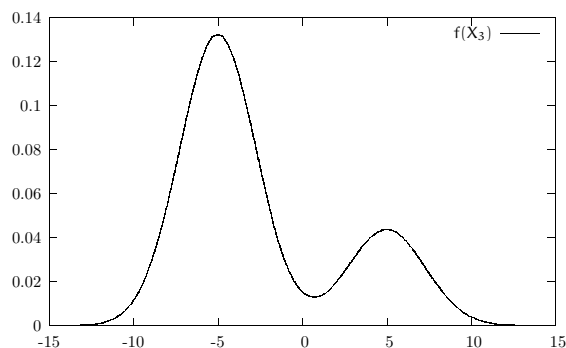


Figure 4.13: The density function for X_3 .

The prior probability density for X_2 and the prior probability distribution for X_1 are trivial to compute as $\{X_2, X_3\}$ are barren with respect to the prior for X_1 and similarly $\{X_1, X_3\}$ are barren with respect to the prior for X_2 . ■

The above examples illustrates that the class of LCG distributions is not closed under the operation of discrete variable elimination. The weak marginal distribution $\mathbb{N}(\mu, \sigma^2)$ may, however, be used as an approximation of the *true* marginal. The weak marginal is the closest non-mixture to the true marginal in terms of the Kullback-Leibler distance (Lauritzen 1996).

Example 53 Consider again the LCG Bayesian network \mathcal{N} from Example 30 on page 62. Figure 4.13 shows the density function for X_3 . Figure 4.14 shows both the density function $f(X_3)$ and the weak marginal $g(X_3)$ for X_3 . It is obvious that the weak marginal is only an approximation of the exact density function. ■

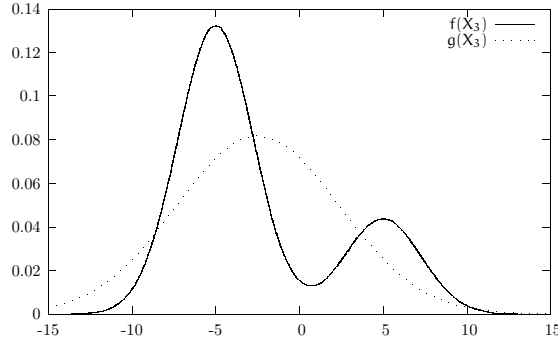


Figure 4.14: The density function $f(X_3)$ for X_3 and its weak marginal $g(X_3)$.

Since the LCG distribution is not closed under the operation of discrete variable elimination and since the operation of discrete variable elimination is not defined when continuous variables are in the domain of the potential to be marginalized, it is required that continuous variables are eliminated before discrete variables. For this reason, when marginalizing over both continuous and discrete variables, we first marginalize over the continuous variables and then over the discrete variables (Lauritzen 1992).

This implies that the (exact) solution method for inference in LCG Bayesian networks induce the partial order $\mathcal{X}_\Delta \prec \mathcal{X}_\Gamma$ on the elimination order. Hence, the continuous variables \mathcal{X}_Γ should be eliminated before the discrete variables \mathcal{X}_Δ . A variable elimination order, which is restricted to induce a certain (partial) order, is referred to as a *strong* elimination order. Hence, we use a strong elimination order to solve LCG Bayesian network by variable elimination. For this reason, inference in an LCG Bayesian network may be more resource intensive than inference in corresponding Bayesian network with the same structure, but consisting only of continuous random variables. Notice that due to independence relations induced by the structure of $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ of an LCG Bayesian network and the structure of the evidence ε , it may in some situations be possible to eliminate discrete variables before continuous variables.

In the special case where the ancestors of $v \in \mathcal{V}$ are all representing continuous variables (i.e., $\text{an}(v) \subseteq \mathcal{V}_\Gamma$) for $X_v \in \mathcal{X}$, the posterior marginal for X_v is a strong marginal. Otherwise, it is a weak marginal. If the posterior for X_v is a weak marginal, the density function of X_v is an unknown mixture of Gaussians, which needs to be computed as part of probabilistic inference.

The normalization constant α computed as part of probabilistic inference is proportional to the density at the observed values of the continuous variables. The proportionality constant is $P(\varepsilon(\Delta) | \varepsilon(\Gamma))$, where $\varepsilon(\Delta)$ is the evidence on discrete variables and $\varepsilon(\Gamma)$ is the evidence on continuous variables. In general, α is scale-dependent and does not make much sense. For instance, the value

of α will depend on whether height is measured in meters or centimeters. If ε only contains discrete variables, then α is the probability of ε .

The presence of both continuous and discrete variables make the operations required for performing probabilistic inference in LCG Bayesian networks more complicated than those required for performing probabilistic inference in discrete Bayesian networks. For a detailed treatment on inference in LCG Bayesian networks, see for example Lauritzen (1992) and Lauritzen & Jensen (2001).

4.2 Solving Decision Models

We build decision models in order to support efficient reasoning and decision making under uncertainty in a given problem domain. Reasoning under uncertainty is the task of computing our updated beliefs in (unobserved) events given observations on other events whereas decision making under uncertainty is the task of identifying the (optimal) decision strategy for the decision maker given observations.

4.2.1 Solving Discrete Influence Diagrams

Inference in an influence diagram $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P}, \mathcal{U})$ is to determine an optimal strategy $\hat{\Delta} = \{\hat{\delta}_1, \dots, \hat{\delta}_n\}$ for the decision maker and compute the maximum expected utility of adhering to $\hat{\Delta}$.

The influence diagram is a compact representation of a joint expected utility function due to the chain rule

$$EU(\mathcal{X}) = \prod_{X_v \in \mathcal{X}_c} P(X_v | X_{pa(v)}) \sum_{w \in \mathcal{V}_u} u(X_{pa(w)}).$$

Applying the \sum -max- \sum -rule (Jensen 1996) on the joint expected utility function, we may solve \mathcal{N} by eliminating variables in the reverse order of the information precedence order \prec . That is, the precedence relation \prec induce a partial order on the elimination of variables in \mathcal{X} . This implies that we use a strong variable elimination order to solve an influence diagram by variable elimination.

Starting with the last decision D_n , the \sum -max- \sum -rule says that we should average over the unknown random variables \mathcal{J}_n , maximize over the decision D_n , average over the random variables \mathcal{J}_{n-1} known to the decision maker at D_n (but not known to the analyst), maximize over D_{n-1} , and so on. The principle is to average over the unknown random variables, maximize over the decision variable, and finally average over the observed random variables.

The intuition behind the application of the \sum -max- \sum -rule in reverse order of decisions is as follows. When we consider the last decision D_n its past is fully observed and the only unobserved variables are the variables never observed or observed after D_n , i.e., \mathcal{J}_n . Hence, after averaging over \mathcal{J}_n , we can select a maximizing argument of the utility function $u(\mathcal{J}(D_n), D_n)$ for each configuration of $\mathcal{J}(D_n)$ as an optimal decision at D_n . Notice that we select a maximizing

argument for each configuration of the past. In principle, this eliminates \mathcal{J}_n and \mathcal{D}_n from the decision problem and we may consider \mathcal{D}_{n-1} as the last decision. This implies that when we solve for \mathcal{D}_{n-1} we assume the decision maker to act optimally for \mathcal{D}_n .

Notice that the variables are observed at the time of decision, but not (necessarily) at the time of analysis. Whether a random variable is known or unknown is defined from the point of view of the decision maker, and not the analyst. In this way we may solve \mathcal{N} by computing the maximum expected utility $\text{MEU}(\hat{\Delta})$ of the optimal strategy Δ as

$$\begin{aligned} \text{MEU}(\hat{\Delta}) &= \sum_{\mathcal{J}_0} \max_{\mathcal{D}_1} \sum_{\mathcal{J}_1} \max_{\mathcal{D}_2} \cdots \sum_{\mathcal{J}_{n-1}} \max_{\mathcal{D}_n} \sum_{\mathcal{J}_n} \text{EU}(\mathcal{X}) \\ &= \sum_{\mathcal{J}_0} \max_{\mathcal{D}_1} \sum_{\mathcal{J}_1} \max_{\mathcal{D}_2} \cdots \sum_{\mathcal{J}_{n-1}} \max_{\mathcal{D}_n} \\ &\quad \prod_{\mathcal{X}_v \in \mathcal{X}_c} P(\mathcal{X}_v | \mathcal{X}_{\text{pa}(v)}) \sum_{\mathcal{U}_w \in \mathcal{X}_u} u(\mathcal{X}_{\text{pa}(w)}). \end{aligned} \quad (4.8)$$

As part of the process, prior to the elimination of each decision \mathcal{D} , we record the maximizing arguments of \mathcal{D} over the utility potential $\psi(\mathcal{D}, \mathcal{J}(\mathcal{D}))$ from which \mathcal{D} is eliminated for each configuration of $\mathcal{J}(\mathcal{D})$. From $\psi(\mathcal{D}, \mathcal{J}(\mathcal{D}))$ we define the (probabilistic) policy function $\delta(\mathcal{D} | \mathcal{J}(\mathcal{D}))$ for \mathcal{D} as

$$\delta(\mathcal{d} | \mathcal{J}(\mathcal{D}) = \mathbf{i}) = \begin{cases} 1 & \text{if } \mathcal{d} = \arg \max_{\mathcal{d}'} \psi(\mathcal{d}', \mathbf{i}), \\ 0 & \text{otherwise,} \end{cases}$$

where we assume the maximizing argument $\arg \max_{\mathcal{d}'} \psi(\mathcal{d}', \mathbf{i})$ to be unique. If it is not unique, then any maximizing argument may be chosen.

Example 54 (Oil Wildcatter) To solve the decision problem of the Oil Wildcatter of Example 32 on page 66 is to identify the optimal decision policies for the test and drill decisions. From the joint expected utility function, $\text{EU}(\mathcal{X})$, over variables \mathcal{X} of $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P}, \mathcal{U})$, we may compute the maximum expected utility, $\text{MEU}(\hat{\Delta})$, of the optimal strategy, $\hat{\Delta} = \{\hat{\delta}_D(S, T), \hat{\delta}_T()\}$, and in the process determine the optimal strategy as follows

$$\text{MEU}(\hat{\Delta}) = \max_T \sum_S \max_D \sum_O P(O)P(S|O, T)(C(T) + U(D, O)).$$

Table 4.1 shows the expected utility function over \mathcal{D} , \mathcal{S} , \mathcal{T} from which the decision policy $\hat{\delta}_D(S, T)$ is identified as the maximizing argument of \mathcal{D} for each configuration of \mathcal{S} and \mathcal{T} . The oil wildcatter should drill for oil unless he performed the test and obtained a diffuse pattern.

Table 4.2 shows the expected utility function over \mathcal{T} from which the decision policy $\hat{\delta}_T()$ is identified as the maximizing argument of \mathcal{T} . Hence, the test should always be performed.

The decision policies $\hat{\delta}_T()$ and $\hat{\delta}_D(S, T)$ are already known from Example 34 on page 69. The maximum expected utility for the decision problem is 22.5. ■

D	S	T	
no	cl	no	0
yes	cl	no	7
no	op	no	0
yes	op	no	7
no	di	no	0
yes	di	no	7
no	cl	yes	-2.4
yes	cl	yes	18.6
no	op	yes	-3.5
yes	op	yes	8
no	di	yes	-4.1
yes	di	yes	-16.6

Table 4.1: The joint expected utility function $\text{EU}(D, S, T)$.

T	
no	21
yes	22.5

Table 4.2: The expected utility function $\text{EU}(T)$.

Solving an influence diagram by performing the variable eliminations according to (4.8) will be highly inefficient even for simple influence diagrams. Instead we will — as in the case of Bayesian networks — apply a generalized version of the distributive law to increase computational efficiency.

For notational convenience, the generalized marginalization operator \mathbb{M} was introduced by Jensen et al. (1994). The marginalization operator works differently for marginalization of random variables and decision variables:

$$\mathbb{M}_X \rho \triangleq \sum_X \rho \quad \text{and} \quad \mathbb{M}_D \rho \triangleq \max_D \rho,$$

where X is a random variable while D is a decision variable. We will use the generalized marginalization operator to explain the process of solving an influence diagram, see (Madsen & Jensen 1999) for details.

Using a generalized version of the distributive law, the solution of an influence diagram may proceed as follows. Let Y be the first random variable to eliminate. The set of utility potentials \mathcal{U} can be divided into two disjoint subsets with respect to Y . Let $\mathcal{U}_Y \subseteq \mathcal{U}$ be the subset of utility potentials including Y in the domain

$$\mathcal{U}_Y = \{u \in \mathcal{U} \mid Y \in \text{dom}(u)\}.$$

Then $\mathcal{U} \setminus \mathcal{U}_Y$ is the set of utility potentials not including Y in the domain. Similarly, let $\mathcal{P}_Y \subseteq \mathcal{P}$ be the subset of probability distributions including Y in

the domain

$$\mathcal{P}_Y = \{\mathbf{P} \in \mathcal{P} \mid Y \in \text{dom}(\mathbf{P})\}.$$

Then $\mathcal{P} \setminus \mathcal{P}_Y$ is the set of probability potentials not including Y in the domain. The elimination process proceeds by local computation in order to maintain efficiency (i.e., we exploit the distributive law to maintain the factorization of the joint expected utility function). Let ϕ_Y be the probability potential obtained by eliminating Y from the combination of all probability potentials in \mathcal{P}_Y and let ψ_Y be the utility potential obtained by eliminating Y from the combination of all probability and utility potentials in $\mathcal{P}_Y \cup \mathcal{U}_Y$ such that

$$\begin{aligned} \phi_Y &= \bigvee_Y \prod_{\phi \in \mathcal{P}_Y} \phi, \\ \psi_Y &= \bigvee_Y \phi_Y \sum_{\psi \in \mathcal{U}_Y} \psi. \end{aligned} \quad (4.9)$$

The two potentials ϕ_Y and ψ_Y will be used to enforce the factorization of the joint expected utility function over $\mathcal{X} \setminus \{Y\}$. The factorization may be achieved by rewriting (4.8) using ϕ_Y and ψ_Y as well as applying the distributive law

$$\begin{aligned} \text{MEU}(\hat{\Delta}) &= \bigvee_{X \in \mathcal{X}} \left(\prod_{\phi \in \mathcal{P}} \phi \sum_{\psi \in \mathcal{U}} \psi \right) \\ &= \bigvee_{X \in \mathcal{X}} \left[\left(\prod_{\phi \in \mathcal{P} \setminus \mathcal{P}_Y} \phi \prod_{\phi' \in \mathcal{P}_Y} \phi' \right) \left(\sum_{\psi \in \mathcal{U} \setminus \mathcal{U}_Y} \psi + \sum_{\psi' \in \mathcal{U}_Y} \psi' \right) \right] \\ &= \bigvee_{X \in \mathcal{X} \setminus \{Y\}} \left[\left(\prod_{\phi \in \mathcal{P} \setminus \mathcal{P}_Y} \phi \right) \bigvee_Y \left(\prod_{\phi' \in \mathcal{P}_Y} \phi' \right) \left(\sum_{\psi \in \mathcal{U} \setminus \mathcal{U}_Y} \psi + \sum_{\psi' \in \mathcal{U}_Y} \psi' \right) \right] \\ &= \bigvee_{X \in \mathcal{X} \setminus \{Y\}} \left[\left(\prod_{\phi \in \mathcal{P} \setminus \mathcal{P}_Y} \phi \right) \left(\left(\sum_{\psi \in \mathcal{U} \setminus \mathcal{U}_Y} \psi \right) \phi_Y + \psi_Y \right) \right] \end{aligned} \quad (4.10)$$

$$= \bigvee_{X \in \mathcal{X} \setminus \{Y\}} \left[\left(\prod_{\phi \in \mathcal{P} \setminus \mathcal{P}_Y} \phi \right) \phi_Y \left(\sum_{\psi \in \mathcal{U} \setminus \mathcal{U}_Y} \psi + \frac{\psi_Y}{\phi_Y} \right) \right]. \quad (4.11)$$

(4.11) specifies a decomposition of the joint expected utility function over $\mathcal{X} \setminus \{Y\}$, and decomposition has the form of (4.8). The decomposition is the product of the summation over the elements of $\mathcal{U} \setminus \mathcal{U}_Y \cup \left\{ \frac{\psi_Y}{\phi_Y} \right\}$ and the product over the elements of $\mathcal{P} \setminus \mathcal{P}_Y \cup \{\phi_Y\}$. In addition, we have performed the elimination of Y by local computations only involving potentials with Y as a domain variable. We say that the sets

$$\mathcal{P} \setminus \mathcal{P}_Y \cup \{\phi_Y\} \quad \text{and} \quad \mathcal{U} \setminus \mathcal{U}_Y \cup \left\{ \frac{\psi_Y}{\phi_Y} \right\},$$

are a value preserving reduction of \mathcal{P} and \mathcal{U} where Y has been eliminated. The elimination of the next variable may proceed in the same manner on $\mathcal{U} \setminus \mathcal{U}_Y \cup \left\{ \frac{\psi_Y}{\phi_Y} \right\}$ and $\mathcal{P} \setminus \mathcal{P}_Y \cup \{\phi_Y\}$.

The division operation in (4.11) is introduced because the combination of probability potentials and utility potentials is non-associative. Thus, either the division should be performed or the probability potentials have to be distributed over the terms of the utility function as in (4.10).

Example 55 (Oil Wildcatter) Utilizing the local computation approach explained above we may solve the Oil Wildcatter problem as follows

$$\begin{aligned} EU(\hat{\Delta}) &= \max_T \sum_S \max_D \sum_O P(O)P(S|O, T)(C(T) + u(D, O)) \\ &= \max_T (C(T) + \sum_S P(S) \max_D \sum_O \frac{P(O)P(S|O, T)}{P(S)} u(D, O)). \end{aligned}$$

The division by $P(S)$ is necessary in order to obtain the correct conditional expected utility for D . This division does not effect the policy.

The benefit of the local computation approach is more profound on large and more complex influence diagrams. ■

4.2.2 Solving LQCG Influence Diagrams

Inference in an LQCG influence diagram $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P}, \mathcal{F}, \mathcal{U})$ is similar to inference in a discrete influence diagram. The task is to determine an optimal strategy, $\hat{\Delta} = \{\hat{\delta}_1, \dots, \hat{\delta}_n\}$, for the decision maker and compute the maximum expected utility of adhering to $\hat{\Delta}$.

The influence diagram is a compact representation of a joint expected utility function due to the chain rule

$$EU(\mathcal{X}_\Delta = i, \mathcal{X}_\Gamma) = \prod_{v \in V_\Delta} P(i_v | i_{pa(v)}) * \prod_{w \in V_\Gamma} p(y_w | X_{pa(w)}) * \sum_{z \in V_U} u(X_{pa(z)}).$$

The solution process for LQCG influence diagrams follows the same approach as the solution process for discrete influence diagrams. The solution process proceeds by applying an extension of the \sum -max- \sum -rule (Madsen & Jensen 2005). The extension is that we need to eliminate the continuous random variables \mathcal{X}_Γ by integration as opposed to summation. We refer the interested reader to the literature for details on the solution process (Kenley 1986, Shachter & Kenley 1989, Poland 1994, Madsen & Jensen 2005).

The optimal strategy $\hat{\Delta} = \{\hat{\delta}_1, \dots, \hat{\delta}_n\}$ will consist of decision policies for both discrete and continuous decision variables. The decision policy for a discrete decision variable $D_i \in \mathcal{X}_\Delta \cap \mathcal{X}_D$ is a mapping from the configuration of its past $\mathcal{J}(D_i)$ to $\text{dom}(D_i)$, whereas the decision policy for a continuous decision variable $D_j \in \mathcal{X}_\Gamma \cap \mathcal{X}_D$ is a multi-linear function in its continuous past $\mathcal{J}(D_i) \cap \mathcal{X}_\Gamma$ conditional on its discrete past $\mathcal{J}(D_i) \cap \mathcal{X}_\Delta$.

Example 56 (Marketing Budget (Madsen & Jensen 2005)) Consider a company manager has to decide on a unit price, Price, to charge for a certain

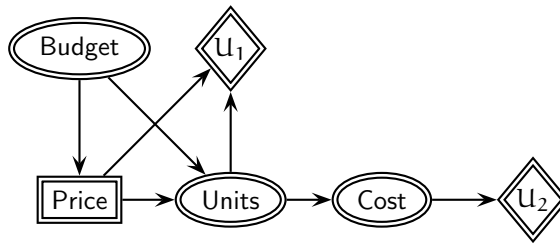


Figure 4.15: Optimization of price given marketing budget size.

item she wants to sell. The number of items sold, **Units**, is a function of the price and marketing budget, **Budget**, whereas the cost of production, **Cost**, is a function of the number of items sold. This scenario can be modeled using the LQCG influence diagram shown in Figure 4.15. Prior to making the decision on price she will be allocated a marketing budget.

The decision problem may be quantified as follows where the unit of utility is thousands of EURs. The distributions of items sold and production cost are

$$\begin{aligned}\mathcal{L}(\text{Units}|\text{Budget} = b, \text{Price} = p) &= \mathcal{N}(20 + 0.2 * b - 0.1 * p, 25) \\ \mathcal{L}(\text{Cost}|\text{Units} = u) &= \mathcal{N}(400 + 10 * u, 2500)\end{aligned}$$

The distribution of marketing budget is

$$\mathcal{L}(\text{Budget}) = \mathcal{N}(100, 400).$$

The cost function is

$$U_2(\text{Cost} = c) = -c$$

and the revenue function is

$$U_1(\text{Price} = p, \text{Units} = u) = u * p.$$

Figure 4.16 on the next page shows the expected utility function as a function of M and P . The optimal decision policy $\delta_P(m)$ for P is a linear function in M : $\delta_P(m) = 105 + m$.

■

4.2.3 Relevance Reasoning

As mentioned in the previous section, a policy δ for D is a mapping from past observations and decisions to the possible decision options at D . When modeling a large and complex decision problem involving a sequence of decisions, the past observations and decisions of a decision may involve a large set of variables. At the same time, it may be that only a small subset of these are essential for the decision. Informally speaking, an observation (or decision) is essential (also

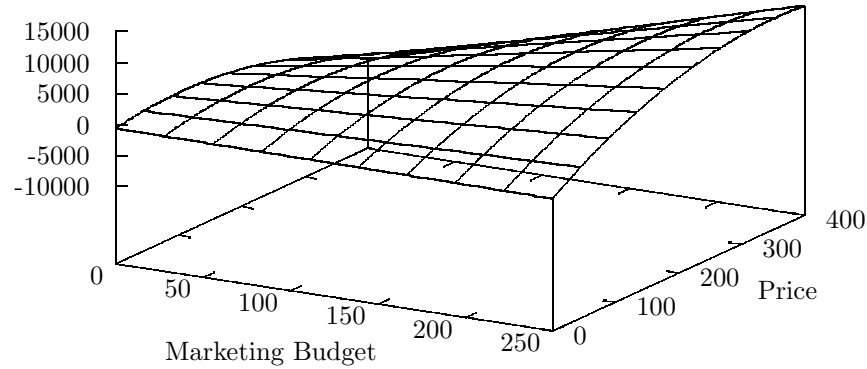


Figure 4.16: Expected utility as a function of price and marketing budget.

known as *requisite*) for a decision, if the outcome of the observation may impact the choice of decision option.

Assume we are able to identify and remove non-requisite parents of each decision. This makes a policy for decision D a function from the requisite past $RP(D)$ to the decision options such that $\delta : RP(D) \rightarrow \text{dom}(D)$. It is not a trivial task to determine the requisite past of a decision D , i.e., the variables observed prior to D , whose values have an impact on the choice of decision option for D (Shachter 1998, Lauritzen & Nilsson 2001, Nielsen 2001).

Definition 12 (Requisite Observation) Let $\mathcal{N} = (\mathcal{X}, \mathcal{G} = (V, E), \mathcal{P}, \mathcal{U})$ be an influence diagram. The observation on variable $Y_v \in \mathcal{J}(D_i)$ is requisite for decision D_i in \mathcal{N} if and only if $v \not\perp_{\mathcal{G}} V_U \cap \text{de}(v_i) \setminus (V_{\mathcal{J}(D_i)} \setminus \{v\})$, where v_i is the node representing D_i .

The solution algorithm will identify some of the non-requisite parents for each decision, but there is no guarantee that all non-requisite parents will be identified and ignored. The implicit identification of non-requisite parents is due to conditional independence properties of the graph.

Similar to the concept of a requisite observation is the concept of a *relevant variable*. The set of variables relevant for a decision, D , is the set of variables observed and the set of decisions made after decision D , which may impact the expected utility of D .

Definition 13 (Relevant Variable) Let $\mathcal{N} = (\mathcal{X}, \mathcal{G} = (V, E), \mathcal{P}, \mathcal{U})$ be an influence diagram. A variable $Y_v \in \mathcal{F}(D_i)$ is relevant for decision D_i if and only if $v \not\perp_{\mathcal{G}} V_{\mathcal{U}} \cap \text{de}(v_i) \mid (V_{\mathcal{J}(D_i)} \setminus \{v\})$, where v_i is the node representing D_i .

Using the concepts of relevant variables and requisite observations it is possible to decompose the structure of an influence diagram $\mathcal{N} = (\mathcal{X}, \mathcal{G} = (V, E), \mathcal{P}, \mathcal{U})$ into a sub-models consisting only of requisite parents and relevant variables for each decision in \mathcal{N} .

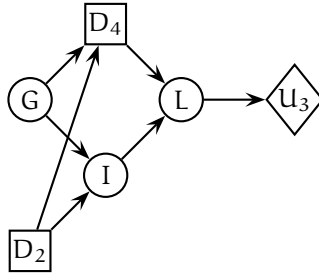


Figure 4.17: The DAG induced by the subset of requisite observations and relevant variables for D_4 .

Example 57 (Nielsen 2001) Consider the influence diagram shown in Figure 3.10 on page 73. Traversing the decision variables in reverse order, we may for each decision variable construct the sub-model consisting of relevant variables and requisite parents only.

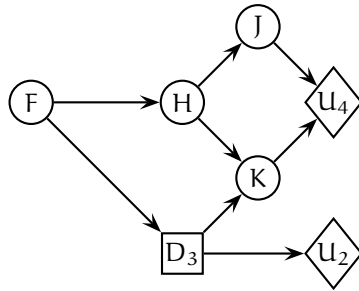


Figure 4.18: The DAG induced by the subset of requisite observations and relevant variables for D_3 .

We consider the decisions in reverse order starting with D_4 . The reasoning proceeds by searching for non-requisite parents of D_4 . By inspection of the diagram it becomes clear that G blocks the flow of information from observations

made prior to D_4 to the only utility descendant U_3 of D_4 . Hence, all other parents are non-requisite. Similarly, we identify the set of relevant variables. Figure 4.17 on the page before shows the DAG induced by the subset of requisite observations and relevant variables for D_4 .

Similarly, Figure 4.18 and Figure 4.19 show the DAGs induced by the subsets of requisite observations and relevant variables for D_3 and D_2 , respectively.

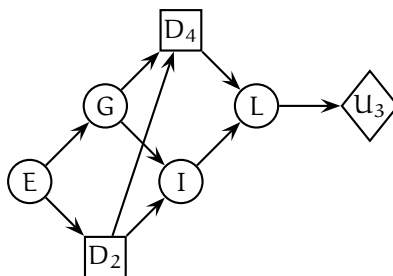


Figure 4.19: The DAG induced by the subset of requisite observations and relevant variables for D_2 .

The DAG induced by the subset of requisite observations and relevant variables for D_1 is equal to the DAG shown in Figure 3.10 on page 73. ■

Decomposing an influence diagram into its sub-models of requisite observations and relevant variables for each decision is very useful for model validation.

4.2.4 Solving LIMIDs

The LIMID representation relaxes the two fundamental assumptions of the influence diagram representation. The assumptions are the total order on decisions and the perfect recall of past decisions and observations. These two assumptions are fundamental to the solution algorithm for influence diagrams described above. Due to the relaxation of these two assumptions, the solution process of LIMIDs becomes more complex than the solution process of influence diagrams.

Let $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P}, \mathcal{U})$ be a LIMID representation of a decision problem. The *Single Policy Updating* (SPU) algorithm is an iterative procedure for identifying (locally) optimal decision policies for the decisions of \mathcal{N} . The basic idea is to start an iterative process from some initial strategy where the policy at each decision is updated while keeping the remaining policies fixed until convergence. For different reasons the starting point is often the uniform strategy where all options are equally likely to be chosen by the decision maker.

As mentioned in the Chapter 3, a decision policy δ_{D_i} is a mapping from the decision past of D_i to the state space $\text{dom}(D_i)$ of D_i such that $\delta_{D_i} : \mathcal{J}(D_i) \rightarrow \text{dom}(D_i)$. This implies that we may use the probabilistic policy func-

tion $\delta'_i(\mathbf{D}_i | \mathcal{J}(\mathbf{D}_i))$ of $\delta_{\mathbf{D}_i}(\mathcal{J}(\mathbf{D}_i))$ introduced in Section 4.2.1 on page 106

$$\delta'_i(\mathbf{d}_i | \mathcal{J}(\mathbf{D}_i) = j) = \begin{cases} 1 & \text{if } \mathbf{d}_i = \delta_{\mathbf{D}_i}(j), \\ 0 & \text{otherwise.} \end{cases}$$

This encoding will play a central role in the process of solving a LIMID.

Let $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P}, \mathcal{U})$ be a LIMID model with chance and decision variables \mathcal{X}_C and \mathcal{X}_D , respectively. A strategy $\Delta = \{\delta_D : D \in \mathcal{X}_D\}$ for \mathcal{N} induces a joint probability distribution $P_\Delta(\mathcal{X})$ over \mathcal{X} as it specifies a probability distribution for each decision variable:

$$P_\Delta(\mathcal{X}) = \prod_{X_v \in \mathcal{X}_C} P(X_v | X_{pa(v)}) \prod_{D_i \in \mathcal{X}_D} \delta'_i. \quad (4.12)$$

The aim of solving \mathcal{N} is to identify a strategy, Δ , maximizing the expected utility

$$EU(\Delta) = \sum_{\mathcal{X} \in \mathcal{X}} P_\Delta(\mathcal{X}) U(\mathcal{X}) = \prod_{X_v \in \mathcal{X}_C} P(X_v | X_{pa(v)}) \prod_{D_i \in \mathcal{X}_D} \delta'_i \sum_{\mathbf{u} \in \mathcal{U}} \mathbf{u}.$$

The SPU algorithm starts with some initial strategy and iteratively updates a single policy until convergence has occurred. Convergence has occurred when no single policy modification can increase the expected utility of the strategy. As mentioned above, a common initial strategy is the uniform strategy $\bar{\Delta} = \{\bar{\delta}'_1, \dots, \bar{\delta}'_n\}$ consisting of uniform policies $\bar{\delta}'_1, \dots, \bar{\delta}'_n$ where $\bar{\delta}'_i(\mathbf{d}) = \frac{1}{\|\mathbf{D}_i\|}$ for each $\mathbf{d} \in \text{dom}(\mathbf{D}_i)$ and each $\mathbf{D}_i \in \mathcal{X}_D$.

Assume Δ is the current strategy and \mathbf{D}_i is the next decision to be considered for a policy update, then SPU proceeds by performing the following steps

Retract Retract the policy δ'_i from Δ to obtain $\Delta_{-i} = \Delta \setminus \{\delta'_i\}$ (i.e., Δ_{-i} is a strategy for all decisions except \mathbf{D}_i).

Update Identify a new policy $\hat{\delta}'_i$ for \mathbf{D}_i by computing

$$\hat{\delta}'_i = \arg \max_{\delta'_i} EU(\Delta_{-i} \cup \{\delta'_i\}).$$

Replace Set $\Delta = \Delta_{-i} \cup \{\hat{\delta}'_i\}$.

SPU may consider the decisions in an arbitrary order. However, if the graph \mathcal{G} specifies a partial order on a subset of decisions $\mathbf{D}_{i_1} \prec \dots \prec \mathbf{D}_{i_j} \prec \dots \prec \mathbf{D}_{i_m}$, then these decisions are processed in reverse order, c.f. the solution process of ordinary influence diagrams.

Example 58 (Solving Breeding Pigs) To solve the breeding pigs decision problem of Example 40 on page 79 is to identify a strategy consisting of one policy for each decision on whether or not to treat each pig for the disease. Using the SPU algorithm described above we may solve the decision problem by iteratively updating each single policy until convergence has occurred.

The uniform strategy will serve as the initial strategy. Hence, we assign a uniform policy $\bar{\delta}_i$ to each decision D_i . As there is a total temporal order on the decisions, we consider them in reverse temporal order.

The SPU algorithm updates the policy of each decision iteratively until convergence. Once convergence has occurred, we have obtained the strategy $\Delta = \{\delta_{D_1}, \delta_{D_2}, \delta_{D_3}\}$, where

$$\delta_{D_1}(R_1) = \begin{cases} \text{no} & R_1 = \text{unhealthy} \\ \text{no} & R_1 = \text{healthy} \end{cases}$$

$$\delta_{D_2}(R_2) = \begin{cases} \text{yes} & R_2 = \text{unhealthy} \\ \text{no} & R_2 = \text{healthy} \end{cases}$$

$$\delta_{D_3}(R_3) = \begin{cases} \text{yes} & R_3 = \text{unhealthy} \\ \text{no} & R_3 = \text{healthy} \end{cases}$$

The strategy is to treat a pig when the test indicates that the pig is unhealthy. Notice that each policy is only a function of the most recent test result. This implies that previous results and decisions are ignored. ■

Probability of Future Decisions

Equation 4.12 on the page before specifies a factorization of the joint probability distribution P_Δ over \mathcal{X} encoded by a strategy Δ . This factorization may be interpreted as a Bayesian network model. With this interpretation we are able to compute the probability of future events under the assumption that the decision maker adheres to the strategy Δ . This property also holds for ordinary influence diagrams.

Example 59 (Breeding Pigs) In Example 58 we identified a strategy $\Delta = \{\delta_{D_1}, \delta_{D_2}, \delta_{D_3}\}$ for the breeding pigs problem. Having identified a strategy, the farmer may be interested in knowing the probability of a pig being healthy when it is sold for slaughtering. This probability may be computed using (4.12 on the preceding page).

The probability of the pig being healthy under strategy Δ is $P_\Delta(H_4 = T) = 67.58$ whereas the probability of the pig being healthy under the uniform strategy $\bar{\Delta}$ is $P_{\bar{\Delta}}(H_4 = T) = 70.55$. The uniform strategy has a lower maximum expected utility though. ■

Minimal LIMIDs

LIMIDs relax the assumption of perfect recall of the decision maker. This implies that the structure of a LIMID defines what information is available to the decision maker at each decision. In addition to specifying what information is

available to the decision maker, we may perform an analysis of which information is relevant to each decision.

It is not always obvious which informational links to include in a LIMID with graph $\mathcal{G} = (V, E)$. Sometimes a link $(v, w) \in E$ from $X_v \in \mathcal{X}_C$ to $D_w \in \mathcal{X}_D$ may be removed from the graph \mathcal{G} without affecting the policies and the expected utility of the computed policies. When this is the case, we say that the link (v, w) (and the parent X_v given by the link) is non-requisite for D_w .

Removing all non-requisite informational links from a LIMID $\mathcal{N} = (\mathcal{X}, \mathcal{G} = (V, E), \mathcal{P}, \mathcal{U})$ produces the minimal reduction $\mathcal{N}^{\min} = (\mathcal{X}, \mathcal{G} = (V, E^*), \mathcal{P}, \mathcal{U})$ of \mathcal{N} . Any LIMID \mathcal{N} has a unique minimal reduction \mathcal{N}^{\min} obtained by iterative removal of informational links from non-requisite parents into decisions.

Since removing a non-requisite parent X from decision D_i may make another previously requisite parent $Y \in X_{\text{pa}(v_i)}$ a non-requisite parent, it is necessary to iteratively process the parent set of each decision until no non-requisite parents are identified. If \mathcal{N} is an ordinary influence diagram, it is sufficient to perform a single pass over the decisions starting with the last decision first. The reason is that we have a total order on the decisions and all decisions are extremal (see Definition 14).

Optimal Strategies

In order to characterize the conditions under which SPU is guaranteed to find an optimal solution we need to define the notion of an *extremal decision*.

Definition 14 (Extremal Decision) Let $\mathcal{N} = (\mathcal{X}, \mathcal{G}, \mathcal{P}, \mathcal{U})$ be a LIMID. A decision variable D_i is extremal if and only if

$$(V_U \cap \text{de}(D_i)) \perp_{\mathcal{G}} \bigcup_{j \neq i} \text{fa}(D_j) \mid \text{fa}(D_i).$$

That is, a decision variable is extremal if all other decisions and their parents are d-separated from the utility descendants of D_i given the family of D_i .

A LIMID is *soluble* if all decisions are extremal. If D_i is extremal in \mathcal{N} , then it has an optimal policy. If all policies in Δ are optimal, then Δ is an optimal strategy.

Example 60 (Breeding Pigs) The Breeding Pigs network In Figure 3.15 on page 79 is not soluble as all decisions are non-extremal. This implies that the local optimal strategy identified is not necessarily a globally optimal strategy.

Similarly, Figure 3.14 of Example 39 on page 78 shows an example of a non-soluble LIMID $\mathcal{N} = (\mathcal{X}, \mathcal{G} = (V, E), \mathcal{P}, \mathcal{U})$. On the other hand, the LIMID $\mathcal{N} = (\mathcal{X}, \mathcal{G} = (V, E \setminus \{(D_i, D)\}), \mathcal{P}, \mathcal{U})$ is soluble as both D_i and D_j are extremal. ■

Notice that since any ordinary influence diagram may be represented as a limited memory influence diagram, the SPU solution process may be used to solve influence diagrams, see e.g. Madsen & Nilsson (2001). Any ordinary influence diagram is a special case of a limited memory influence diagram. The

LIMID representation of an ordinary influence diagram will produce an optimal strategy.

See Lauritzen & Nilsson (2001) for more details on the solution process.

4.3 Solving OOPNs

For the purpose of inference, an object-oriented model is unfolded. The unfolded network is subsequently transformed into the computational structure used for inference. This implies that to solve an object-oriented model is equivalent to solving its unfolded network. Hence, from this point of view of inference there is no difference between an object-oriented network and a flat network.

4.4 Summary

In this chapter we have considered the process of solving probabilistic networks. As the exact nature of solving a query against a probabilistic network depends on the type of model, the solution processes of Bayesian networks and influence diagrams have been considered separately.

We build Bayesian network models in order to support efficient reasoning under uncertainty in a given domain. Reasoning under uncertainty is the task of computing our updated beliefs in (unobserved) events given observations on other events, i.e., evidence.

We have considered the task of computing the posterior marginal of each unobserved variable, Y , given a (possibly empty) set of evidence ε , i.e., $P(Y|\varepsilon)$. The solution process we have focused on computes the posterior marginal for all unobserved variables using a two-phase message passing process on a junction tree structure.

We build decision models in order to support efficient reasoning and decision making under uncertainty in a given problem domain. Reasoning under uncertainty is the task of computing our updated beliefs in (unobserved) events given observations on other events whereas decision making under uncertainty is the task of identifying the (optimal) decision strategy for the decision maker given observations.

We have derived a method for solving influence diagrams by variable elimination. In the process of eliminating variables we are able to identify the decision policy for each decision. The resulting set of policies is the optimal strategy for the influence diagram.

The LIMID representation relaxes the two fundamental assumptions of the influence diagram representation. The assumptions are the total order on decisions and the perfect recall of past decisions and observations. These two assumptions are fundamental to the solution algorithm for influence diagrams described above. Due to the relaxation of these two assumptions, the solution process of LIMIDs becomes more complex than the solution process of influence diagrams.

We have described how the single policy updating algorithm iteratively identifies a set of locally optimal decision policies. A decision policy is globally optimal when the decision is extremal.

Finally, an OOPN is solved by solving its equivalent unfolded network.

Bibliography







- Andersen, S. K., Olesen, K. G., Jensen, F. V. & Jensen, F. (1989). HUGIN — a Shell for Building Bayesian Belief Universes for Expert Systems, *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pp. 1080–1085.
- Cobb, B. R. & Shenoy, P. P. (2004). Decision Making with Hybrid Influence Diagrams Using Mixtures of Truncated Exponentials, *Working paper no. 304*, School of Business, University of Kansas.
- Cooper, G. F. (1990). The computational complexity of probabilistic inference using Bayesian belief networks, *Artificial Intelligence* **42**(2-3): 393–405.
- Cowell, R. G., Dawid, A. P., Lauritzen, S. L. & Spiegelhalter, D. J. (1999). *Probabilistic Networks and Expert Systems*, Springer-Verlag.
- Dagum, P. & Luby, M. (1993). Approximating probabilistic inference in Bayesian belief networks is NP-hard, *Artificial Intelligence* **60**: 141–153.
- Dawid, A. P. (1992). Applications of a general propagation algorithm for probabilistic expert systems, *Statistics and Computing* **2**: 25–36.
- DeGroot, M. (1986). *Probability and Statistics*, second edn, Addison-Wesley. Reprint with corrections, 1989.
- Frydenberg, M. (1989). The chain graph Markov property, *Scandinavian Journal of Statistics* **17**: 333–353.
- Howard, R. A. & Matheson, J. E. (1981). Influence diagrams, in R. A. Howard & J. E. Matheson (eds), *Readings in Decision Analysis*, Strategic Decisions Group, Menlo Park, California, chapter 38, pp. 763–771.
- Jensen, F., Jensen, F. V. & Dittmer, S. (1994). From influence diagrams to junction trees, *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, Morgan Kaufmann Publishers, San Francisco, California, pp. 367–373.
- Jensen, F. V. (1996). *An Introduction to Bayesian Networks*, UCL Press, London.

- Jensen, F. V. (2001). *Bayesian Networks and Decision Graphs*, Springer-Verlag.
- Jensen, F. V. & Jensen, F. (1994). Optimal junction trees, *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, Morgan Kaufmann Publishers, San Francisco, California, pp. 360–366.
- Jensen, F. V., Lauritzen, S. L. & Olesen, K. G. (1990). Bayesian updating in causal probabilistic networks by local computations, *Computational Statistics Quarterly* **4**: 269–282.
- Kenley, C. R. (1986). *Influence Diagram Models with Continuous Variables*, PhD thesis, Department of Engineering-Economic Systems, Stanford University, California.
- Kim, J. H. & Pearl, J. (1983). A computational model for causal and diagnostic reasoning in inference systems, *Proceedings of the Eighth International Joint Conference on Artificial Intelligence*, pp. 190–193.
- Koller, D. & Pfeffer, A. (1997). Object-oriented Bayesian networks, *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence*, pp. 302–313.
- Laskey, K. B. & Mahoney, S. M. (1997). Network fragments: Representing knowledge for constructing probabilistic models, *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence*, pp. 334–341.
- Lauritzen, S. L. (1992). Credit Valuation Using HUGIN.
- Lauritzen, S. L. (1992). Propagation of probabilities, means and variances in mixed graphical association models, *Journal of the American Statistical Association* **87**(420): 1098–1108.
- Lauritzen, S. L. (1996). *Graphical models*, Oxford Statistical Science Series, Clarendon Press, Oxford, England.
- Lauritzen, S. L. (2001). Causal inference from graphical models, in O. E. Barndorff-Nielsen, D. R. Cox & C. Klüppelberg (eds), *Complex Stochastic Systems*, Chapman and Hall/CRC, London, pp. 63–107.
- Lauritzen, S. L., Dawid, A. P., Larsen, B. N. & Leimer, H.-G. (1990). Independence properties of directed Markov fields, *Networks* **20**(5): 491–505. Special Issue on Influence Diagrams.
- Lauritzen, S. L. & Jensen, F. (2001). Stable Local Computation with Mixed Gaussian Distributions, *Statistics and Computing* **11**(2): 191–203.
- Lauritzen, S. L. & Nilsson, D. (2001). Representing and solving decision problems with limited information, *Management Science* **47**: 1238–1251.

- Lauritzen, S. L. & Spiegelhalter, D. J. (1988). Local computations with probabilities on graphical structures and their application to expert systems, *Journal of the Royal Statistical Society, Series B* **50**(2): 157–224.
- Lin, Y. & Druzdzel, M. J. (1997). Computational advantages of relevance reasoning in Bayesian networks, in D. Geiger & P. P. Shenoy (eds), *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence*, Morgan Kaufmann Publishers, San Francisco, pp. 342–350.
- Madsen, A. & Jensen, F. (2005). Solving linear-quadratic conditional Gaussian influence diagrams, *International Journal of Approximate Reasoning* **38**(3): 263–282.
- Madsen, A. L. & Jensen, F. V. (1999). Lazy propagation: A junction tree inference algorithm based on lazy evaluation, *Artificial Intelligence* **113**(1–2): 203–245.
- Madsen, A. L. & Nielsen, L. (1996). *Deep Green - a bridge playing system based on Bayesian networks*, Master's thesis, Department of Computer Science, Aalborg University, Denmark.
- Madsen, A. L. & Nilsson, D. (2001). Solving Influence Diagrams using HUGIN, Shafer-Shenoy and Lazy Propagation, *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*, pp. 337–345.
- Neil, M., Fenton, N. & Nielsen, L. M. (2000). Building Large-Scale Bayesian Networks, *The Knowledge Engineering Review* **15**(3): 257–284.
- Nielsen, T. D. (2001). Decomposition of Influence Diagrams, *Proceedings of the Sixth European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty*, pp. 144–155.
- Nielsen, T. D. & Jensen, F. V. (1999). Welldefined decision scenarios, *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, pp. 502–511.
- Olmsted, S. (1983). *On representing and solving decision problems*, Phd thesis, Department of Engineering-Economic Systems, Stanford University, Stanford, CA.
- Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Series in Representation and Reasoning, Morgan Kaufmann Publishers, San Mateo, California.
- Pearl, J. (2000). *Causality: models, reasoning, and inference*, Cambridge University Press.
- Poland, W. B. (1994). *Decision Analysis with Continuous and Discrete Variables: A Mixture Distribution Approach*, PhD thesis, Engineering-Economic Systems, Stanford University, Stanford, CA.

- Raiffa, H. (1968). *Decision Analysis*, Addison-Wesley, Reading, MA.
- Shachter, R. (1998). Bayes-Ball: The Rational Pasttime (for Determining Irrelevance and Requisite Information in Belief Networks and Influence Diagrams), *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, pp. 480–487.
- Shachter, R. D. (1986). Evaluating influence diagrams, *Operations Research* **34**(6): 871–882.
- Shachter, R. D. (1990). Evidence absorption and propagation through evidence reversals, in M. Henrion, R. D. Shachter, J. F. Lemmer & L. M. Kanal (eds), *Uncertainty in Artificial Intelligence*, Elsevier Science Publishers B. V. (North-Holland), Amsterdam, pp. 173–190.
- Shachter, R. D. & Kenley, C. R. (1989). Gaussian influence diagrams, *Management Science* **35**(5): 527–549.
- Shachter, R. D. & Peot, M. A. (1992). Decision making using probabilistic inference methods, in D. Dubois, M. P. Wellman, B. D’Ambrosio & P. Smets (eds), *Proceedings of the Eighth Conference on Uncertainty in Artificial Intelligence*, Morgan Kaufmann Publishers, San Mateo, California, pp. 276–283.
- Spirtes, P., Glymour, C. & Scheines, R. (2000). *Causation, Prediction, and Search*, Adaptive Computation and Machine Learning, 2nd edn, MIT Press.
- Vomlelová, M. & Jensen, F. (2002). An Extension of Lazy Evaluation for Influence Diagrams Avoiding Redundant Variables in the Potentials, *First European Workshop on Probabilistic Graphical Models*, pp. 186–193.
- Wermuth, N. & Lauritzen, S. L. (1990). On substantive research hypotheses, conditional independence graphs and graphical chain models, *Journal of the Royal Statistical Society, Series B* **52**(1): 21–50.

List of Symbols and Abbreviations

arg	E.g., $\arg \max_{\mathbf{a} \in A} EU(\mathbf{a})$ denotes the argument of max (i.e., the \mathbf{a}) that maximizes $EU(\mathbf{a})$
BN	Bayesian network
CG	Conditional Gaussian
LCG	Linear conditional Gaussian
LIMID	Limited-memory influence diagram
LQCG	Linear-quadratic conditional Gaussian
CPT	Conditional probability table
ID	Influence diagram
OBN	Object-oriented Bayesian network
OOPN	Object-oriented probabilistic network
	Discrete chance node, representing a discrete random variable
	Continuous chance node, representing a continuous random variable
	Discrete decision node, representing a discrete decision variable
	Continuous decision node, representing a continuous decision variable
	Discrete utility node, representing a discrete utility function
	Continuous utility node, representing a continuous utility function
\triangleq	Defined as
\equiv	Equivalent to

\propto	Proportional to
$\ \cdot\ $	Domain size (i.e., $\ X\ = \text{dom}(X) $)
\perp	Independent
\perp_p	Independent with respect to distribution p
$\not\perp$	Dependent
$\not\perp_p$	Dependent with respect to distribution p
\perp	d-separated
$\perp_{\mathcal{G}}$	d-separated in graph \mathcal{G}
$\not\perp$	d-connected
$\not\perp_{\mathcal{G}}$	d-connected in graph \mathcal{G}
\mathcal{S}	Separator set
\mathcal{E}	Evidence potential
ε	Evidence
η	Normalization operator
\sim	Connected
\rightarrow	Connected by directed link
$—$	Connected by undirected link
$\overset{\mathcal{G}}{\sim}$	Connected in graph \mathcal{G}
$\overset{\mathcal{G}}{\rightarrow}$	Connected by directed link in graph \mathcal{G}
$\overset{\mathcal{G}}{—}$	Connected by undirected link in graph \mathcal{G}
$\not\sim$	Not connected
$\overset{\mathcal{G}}{\not\sim}$	Not connected in graph \mathcal{G}
$\langle \mathbf{u}, \dots, \mathbf{v} \rangle$	Path from \mathbf{u} to \mathbf{v}
dom	Domain (of variable or set of variables)
	“given” (e.g., “ $\mathbf{a} \mathbf{b}$ ” means “ \mathbf{a} given \mathbf{b} ”)
pa	Parents of
fa	Family of

ch	Children of
an	Ancestors of
An	Ancestral set of
de	Descendants of
nd	Non-descendants of
F	Boolean value “false”
T	Boolean value “true”
J	Decision past
\mathcal{F}	Decision future
EU	Expected utility
MEU	Maximum expected utility
\mathbb{N}	Normal (Gaussian) distribution
\mathbb{N}_k	k-dimensional Normal distribution
\mathcal{L}	Law of (e.g., $\mathcal{L}(X) = \mathbb{N}(\mu, \sigma^2)$, also denoted $X \sim \mathbb{N}(\mu, \sigma^2)$)
\mathbb{R}	The set of all real numbers
\mathbb{S}	Scope
\mathcal{I}	Input variables
\mathcal{H}	Private (hidden) variables
\mathcal{O}	Output variables
$\mathcal{X}(\varepsilon)$	Evidence variables (i.e., subset of \mathcal{X} for which their values are known)
\mathcal{E}_ε	Evidence function for $\mathcal{X}(\varepsilon)$
$\mathcal{S}, \mathcal{U}, \mathcal{V}, \mathcal{W}$	Sets of nodes
\mathcal{V}	Set of nodes of a model
\mathcal{V}_Δ	The subset of \mathcal{V} that represent discrete variables
\mathcal{V}_Γ	The subset of \mathcal{V} that represent continuous variables
u, v, w, \dots	Nodes
$\alpha, \beta, \gamma, \dots$	Nodes

X, Y_i, Z_k	Variables or sets of variables
X_W	Subset of variables corresponding to set of nodes W
\mathcal{X}	The set of variables of a model; note that $\mathcal{X} = \mathcal{X}_V$
\mathcal{X}_W	Subset of \mathcal{X} , where $W \subseteq V$
X_u, X_α	Variables corresponding to nodes u and α , respectively
x, y_i, z_k	Configurations/states of (sets of) variables
x_Y	Projection of configuration x to $\text{dom}(Y)$, $X \cap Y \neq \emptyset$
\mathcal{X}_C	The set of chance variables of a model
\mathcal{X}_D	The set of decision variables of a model
\mathcal{X}_Δ	The subset of discrete variables of \mathcal{X}
\mathcal{X}_Γ	The subset of continuous variables of \mathcal{X}
\mathcal{U}	The set of utility functions of a model
V_U	The subset of V representing utility functions
$u(X)$	Utility function $u \in \mathcal{U}$ with the of variables X as domain

Index

- Σ -max- Σ -rule, *see* influence diagram
- \sim , 4
- $\|\cdot\|$, 6
- \rightarrow , *see* link, directed
- \xrightarrow{g} , *see* link, directed
- $\overset{g}{\dashrightarrow}$, *see* link, undirected
- $-$, *see* link, undirected
- abductive reasoning, *see* reasoning, abductive
- acyclic directed graph, *see* graph, acyclic directed
- An, *see* ancestral set
- an, *see* ancestor
- ancestor, 4
- ancestral set, 5
- Apple Jack, 56, 69
- arc, *see* link
- arc reversal, 42
 - example, 43
- Asia, 58
- axioms, *see* probability, axioms
- Balls in An Urn, 30
 - Bayes' factor, 44
 - probability distribution for variables, 31
 - rule of total probability, 32
- barren variable, *see* variable, barren
- Bayes' factor, 44
- Bayes' rule, 40
 - interpretation of, 42
- Bayesian network
 - direct inference approach, 94
 - discrete, 55, 55–59
 - indirect inference approach, 102
 - linear CG, 61, 60–64
- query, 92
- query based inference, 92
- Burglary or Earthquake, 11, 20
 - Bayes' rule, 41
 - conditional independence, 45
 - conditional probability, 28
 - conditional probability table, 33
 - events, 28
- category, *see* variable, category
- causal network, 12–18
 - converging connection in, 16–18
 - diverging connection in, 15–16, 18
 - flow of information in, 12–18
 - serial connection in, 14–15, 18
 - types of connections in, 13
- causal reasoning, *see* reasoning, causal
- causality, 10–12, 56
- ch, *see* child
- chain graph, *see* graph, chain
- chain rule, 48
 - Bayesian networks, 50, 55
 - influence diagrams, 66
 - LCG Bayesian networks, 61
 - LQCG influence diagrams, 75
 - object-oriented Bayesian network models, 85
- Chest Clinic, *see* Asia
 - junction tree, 98
- child, 4
- chord, 98
- clique, 96
- combination, *see* probability potential, combination

- conditional independence, *see* independence, conditional
- conditional probability, 28
- converging connection, *see* causal network, converging connection in
- cycle, 5
 - directed, 5
- D-map, 46
- d-separation, 18, 20, 49
 - example, 20
- DAG, *see* graph, acyclic directed
- de, *see* descendant
- decision future, 68
- decision history, 68
- decision past, 68
- decision problem
 - well-defined, 69
- decision variable, 64
 - informational parents, 73
- decision variables
 - partial order, 67
 - total order, 69
- deductive reasoning, *see* reasoning, deductive
- default prior probability distribution, 83
- dependence and independence
 - conditional, 19–20
- descendant, 4
- diagnostic reasoning, *see* reasoning, diagnostic
- directed acyclic graph, *see* graph, acyclic directed
- directed cycle, *see* cycle, directed
- directed global Markov property, 19, 21, 49
 - example, 21
- directed graph, *see* graph, directed
- distributive law, *see* probability calculus, distributive law of
- diverging connection, *see* causal network, diverging connection in
- division, *see* probability potential, division
 - by zero, 36, 41
- dom, *see* variable, domain
- edge, *see* link
- elimination order, 96
 - strong, 105, 106
- equivalent variables, *see* variable, equivalence
- event, 28
- evidence, 9–10
 - hard, 9
 - likelihood, *see* evidence, soft potential, 35
 - soft, 9
- expected utility, 64
- Expert system, 89
- explaining away, 2, 18
- factorization
 - recursive, *see* probability distribution, recursive factorization of
- flat network, 85
- fundamental rule, 29, 40
- generalized marginalization operator, 108
- graph
 - acyclic directed, 5
 - chain, 1
 - connected, 5
 - directed, 4
 - instances, 81
 - moral, 5
 - moralization, 96
 - skeleton, 5
 - undirected, 4
- head, 33
- head,tail, 95
- head-to-head, 4, 16, 20
- Hugin algorithm, 101
- hypothesis variables, 58
- I-map, 46

- independence, 45
 - conditional, 19–20, 45
 - represented in DAGs, 46
- inference engine, 89
- influence diagram, 63
 - \sum -max- \sum -rule, 106
 - discrete, 65, 64–74
 - information sets, 67
 - limited memory, 78, 77–80
 - minimal reduction, 117
 - soluble, 117
 - linear-quadratic CG, 74, 74–77
 - \sum -max- \sum -rule, 110
 - maximum expected utility, 68
 - no-forgetting links, 73
 - policy, 68
 - policy function, 107
 - strategy, 68
 - optimal, 68
- informational link, *see* link, informational
- inheritance, 86
- instance, 81
- intercausal reasoning, *see* explaining away
- interface variables, 81
- intervening action, 64
- intervening decision, 69

- joint probability distribution, *see* probability distribution
- junction tree, 98
 - COLLECTINFORMATION, 100
 - DISTRIBUTEINFORMATION, 100
 - propagation, 99
 - root, 100

- kind, *see* variable, kind

- LCG Bayesian network, *see* Bayesian network, linear CG
- LCG distribution, *see* linear conditional Gaussian distribution
- likelihood evidence, *see* evidence, soft
- LIMID, *see* influence diagram, limited memory

- linear conditional Gaussian distribution, 61
- linear-quadratic conditional Gaussian influence diagram, *see* influence diagram, linear-quadratic CG
- link, 4
 - directed, 4
 - informational, 65
 - undirected, 4
- LQCG influence diagram, *see* influence diagram, linear-quadratic CG, 75

- marginalization, 32, 37
 - generalized operator, 108
- Markov property, *see* directed global Markov property
- maximum expected utility principle, 64
- MEU
 - maximum expected utility, 107
- minimal reduction, *see* influence diagram, limited memory, minimal reduction
- moral graph, *see* graph, moral
- multivariate Gaussian distribution, 61
- mutually exclusive states, 55, 60

- nd, *see* non-descendants
- network class, 81
 - default instance, 86
 - internal scope, 82
- no-forgetting, 65
- node
 - notation, 9
 - symbols, 8
 - vs. variable, 7
- non-descendants, 5
- non-intervening action, 64
- normalization, *see* probability potential, normalization of
- normalization constant, 91
- notation, 9

- nuisance variable, *see* variable, nuisance
- object, 81
- object-oriented probabilistic network, 80–87
 - instance tree, 86
 - object-orientation
 - classes, 80
 - definition, 80
 - inheritance, 80
 - objects, 80
- observation
 - essential, 111
 - requisite, 112
- Oil Wildcatter, 76
- OoBN, *see* object-oriented probabilistic network
- pa, *see* parent
- parent, 4
- path, 4
 - blocked in DAG, 20
 - blocked in undirected graph, 5
 - directed, 4
- perfect map, 46
- perfect recall, *see* no-forgetting
- polytree, 5, 92
- posterior probability distribution, *see* probability distribution, posterior
- potential calculus, 36
- probability
 - axioms, 29
- probability calculus, 36
 - chain rule of, 48
 - distributive law of, 38
 - fundamental rule of, 29, 40
- probability distribution
 - decomposition, 25
 - for variables, 30
 - graphical representation of conditional, 33
 - marginal, 33
 - posterior, 36
 - recursive factorization of, 25
- probability potential, 34
 - combination, 36
 - division, 36
 - marginalization, *see* marginalization
 - normalization of, 34, 36
 - vacuous, 35
- projection, *see* marginalization
- propagation, *see* junction tree, propagation
- reasoning
 - abductive, 2
 - causal, 2
 - deductive, 2
 - diagnostic, 2
 - intercausal, *see* explaining away
- recursive factorization, *see* probability distribution, recursive factorization of
- relevant network, 93
- relevant variable, 112
- rule of total probability, 31
- serial connection, *see* causal network, serial connection in Single Policy Updating, 114
- singly connected graph, *see* polytree
- skeleton, *see* graph, skeleton
- soluble, *see* influence diagram, limited memory, soluble
- SPU, *see* Single Policy Updating
- subclass, 87
- subtype, *see* variable, subtype
- tail, 33
- topological ordering, 49
- tree, 5
- undirected graph, *see* graph, undirected
- unfolded network, 85
- utility function, 64
- vacuous potential, *see* probability potential, vacuous
- value function, *see* utility function

- value nodes, 65
- variable, 6–9
 - barren, 39, 92
 - basic, 81
 - binding link, 83
 - bound, 83
 - category, 7
 - conditioned variable, 55
 - conditioning variables, 55
 - decision, 7
 - decision future, 68
 - decision history, 68
 - decision past, 68
 - deterministic, 7
 - domain, 6
 - eliminating, 32
 - elimination, 93, 96
 - equivalence, 83
 - extremal decision, 117
 - independence, *see* dependence
 - and independence
 - kind, 7
 - marginalizing out, 32
 - notation, 9
 - nuisance, 93
 - policy, 68
 - qualified name, 82
 - random, 7
 - scope, 82
 - simple name, 82
 - strong marginal, 103
 - strong type checking, 83
 - subtype, 8
 - target, 92
 - taxonomy, 8
 - vs. node, 7
 - weak marginal, 103