Probabilistic Reasoning

with Bayesian Networks

Fall 2023

authors: Linda van der Gaag
Silja Renooij

Universiteit Utrecht

Preface

In artificial-intelligence research, the Bayesian network framework for automated reasoning with uncertainty has rapidly gained popularity since its introduction in the late 1980s. Bayesian networks belong to the family of *Probabilistic Graphical Models*, providing a powerful formalism for representing a joint probability distribution on a set of random variables. In addition, the Bayesian network framework offers algorithms for efficient probabilistic inference. Over the years, numerous decision-support systems employing the framework have been developed for various domains of application, ranging from probabilistic information retrieval to medical diagnosis. This syllabus provides a tutorial introduction to the Bayesian network framework and highlights the basics underlying ongoing research in applying the framework for problem solving in real-life domains. Each chapter includes examples as well as a number of exercises; solutions, answers or hints to exercises (indicated by a *) are provided in the final chapter.

This syllabus was first written in the late 1990s by L.C. van der Gaag and has been continuously under development ever since. Since 2001, adaptations and extensions have been made mostly by S. Renooij. The syllabus is by no means devoid from imperfections and any useful comments on its contents are greatly appreciated.

Changes to the previous editions (Fall 2021 and Fall 2022) consist mostly of introducing notation boxes and further extension and clarifications of exercises and their answers.

Linda van der Gaag
Silja Renooij
Utrecht University
August 2023

Contents

Chapter 1

Introduction

Over the past decades interest in the results of artificial-intelligence research has been growing rapidly, culminating in big data and (deep) machine learning hypes. Steady attention throughout the years has been attracted by especially the area of *intelligent* or *knowledge-based systems*. Traditionally knowledge-based systems are considered to be part of 'symbolic AI' which refers to AI systems that use some symbolic representation (e.g. logic) to incorporate and reason with human knowledge [Lucas & Van der Gaag, 1991, Jackson, 1990]. Until the 1980s, symbolic AI was the dominant AI paradigm; since then 'sub-symbolic AI' started to get a foothold. Sub-symbolic AI systems, such as neural networks, typically capture associations learned from data rather than human knowledge; their 'reasoning' is based on the manipulation of numbers rather than symbols. The need for eXplainable AI (XAI) has resulted in more and more research into combining symbolic and sub-symbolic systems [Ilkou & Koutraki, 2020]. Probabilistic graphical models, such as Bayesian networks, combine characteristics of both symbolic and sub-symbolic systems in a single framework.

Knowledge-Based Systems and Uncertainty

Knowledge-based systems are typically designed to deal with real-life problems that require considerable human knowledge and expertise for their solution; in the early days they were therefore called *expert systems*. Examples of such systems range from medical diagnosis and technical trouble shooting to financial advice and product design. It is their ability to capture and reason with (specialised) human knowledge that allows knowledge-based systems to arrive at a performance comparable to that of human experts. Recognising that complex decisions require more than the capability to reason alone, knowledge-based systems are also referred to as *decision-support systems*. These systems by now have found their way from academic laboratories to the industrial world and are being integrated into conventional software environments.

As more and more decision-support systems were being developed for a large variety of problems, it became apparent that the knowledge required to solve these problems often is not precisely defined but instead is of an imprecise nature. In fact, many real-life problem domains are fraught with uncertainty. Human experts in these domains typically are able to form judgements and take decisions based on uncertain, incomplete, and sometimes even contradictory information. To be of practical use, a decision-support system has to deal with such information at least equally well. For this purpose, a decision-support system employs a formalism for representing uncertainty and an associated algorithm for manipulating uncertain information. The major research topic in artificial intelligence of *reasoning with uncertainty*, or *plausible reasoning*, addresses the design of such formalisms and algorithms [Shafer & Pearl, 1990].

Early Probabilistic Reasoning

As probability theory is a mathematically well-founded theory about uncertainty, having a long and outstanding tradition of research and experience, it is not surprising that this theory takes a prominent place in research on reasoning with uncertainty in decision-support systems. Unfortunately, applying probability theory in a knowledge-based context is not as easy as it may seem at first sight. Straightforward application of the basic concepts from probability theory leads to insuperable problems of computational complexity: explicit representation of a joint probability distribution requires exponential space (exponential in the number of variables discerned), and even if the distribution could be represented more economically, computing probabilities of interest by the basic rules of marginalisation and conditioning would have an exponential time complexity. The rich history of applying probability theory for reasoning with uncertainty in decision-support systems shows various attempts to settle these problems.

In this chapter, we sketch the historical background of applying probability theory in a knowledge-based system. We would like to note that our intention is not to be complete, but merely to give an impression of the problems encountered by researchers pioneering in automated probabilistic inference. In our sketch, we focus on the task of (medical) diagnosis. For a given problem domain, we discern a set of possible *hypotheses* $\boldsymbol{H} = \{h_1, \ldots, h_n\}$, $n \geq 1$, and a set of pieces of *evidence* $\boldsymbol{E} = \{e_1, \ldots, e_m\}$, $m \geq 1$, that may be observed in relation with these hypotheses. For ease of exposition, we assume that each of the hypotheses is either *true* or *false*; equally, we assume that each of the pieces of evidence is either *true* or *false*. A *diagnostic problem* in this domain now is a set of pieces of evidence $\boldsymbol{e} \subseteq \boldsymbol{E}$ that is actually observed and needs to be explained in terms of the hypotheses discerned. A *diagnosis* for a problem $\boldsymbol{e}$ under consideration is a set of hypotheses $\boldsymbol{h} \subseteq \boldsymbol{H}$ that best explains $\boldsymbol{e}$.

As early as in the 1960s several research efforts on automated reasoning with uncertainty for diagnostic applications were undertaken [Warner *et al.*, 1961, Gorry & Barnett, 1968, De Dombal *et al.*, 1972]. The systems constructed in this period were based to a large extent on application of Bayes' Theorem; in the sequel, we will refer to the approach taken in these early systems as the *naive-Bayesian approach*. In this approach, the basic idea of computing a diagnosis for a set of actually observed pieces of evidence $\boldsymbol{e} \subseteq \boldsymbol{E}$ is to compute for all sets of hypotheses $\boldsymbol{h} \subseteq \boldsymbol{H}$ the conditional probability $\Pr(\boldsymbol{h} \mid \boldsymbol{e})$ from the distribution Pr on the domain at hand, and then select a set $\boldsymbol{h}' \subseteq \boldsymbol{H}$ with highest probability. Since for real-life applications the conditional probabilities $\Pr(\boldsymbol{e} \mid \boldsymbol{h})$ often are easier to come by than the conditional probabilities $\Pr(\boldsymbol{h} \mid \boldsymbol{e})$, generally Bayes' Theorem is used for computing the required probabilities:

$$\Pr(\boldsymbol{h} \mid \boldsymbol{e}) \quad = \quad \frac{\Pr(\boldsymbol{e} \mid \boldsymbol{h}) \cdot \Pr(\boldsymbol{h})}{\Pr(\boldsymbol{e})}$$

Evidently, this approach is quite expensive from a computational point of view: because a diagnosis may be composed of several different hypotheses, the number of probabilities to be computed equals $2^n - 1$. To overcome this problem of computational *time complexity*, a simplifying assumption is made: it is assumed that all hypotheses are *mutually exclusive* and *collectively exhaustive*. With this assumption only the n singleton hypothesis sets $\{h_i\}$ have to be considered as possible diagnoses. So, only the probabilities $\Pr(h_i \mid \boldsymbol{e})$ (writing h_i instead of $\{h_i\}$) for all $h_i \in \boldsymbol{H}$ have to be computed. To this end, once more Bayes' theorem is used:

$$\Pr(h_i \mid \boldsymbol{e}) \quad = \quad \frac{\Pr(\boldsymbol{e} \mid h_i) \cdot \Pr(h_i)}{\Pr(\boldsymbol{e})} \quad = \quad \frac{\Pr(\boldsymbol{e} \mid h_i) \cdot \Pr(h_i)}{\sum_{k=1}^{n} \Pr(\boldsymbol{e} \mid h_k) \cdot \Pr(h_k)}$$

For automated application of Bayes' theorem in this form, several probabilities are required from the joint probability distribution Pr on the domain at hand. In fact, conditional probabilities $\Pr(\boldsymbol{e} \mid h_k)$, $k = 1, \ldots, n$, for *every* combination of pieces of evidence $\boldsymbol{e} \subseteq \boldsymbol{E}$, have to be available. Apart from the fact that it is hardly likely that these probabilities will be readily available in a real-life problem domain, this means storing exponentially many probabilities. To overcome this problem of computational *space complexity*, a second simplifying assumption is made: it is assumed that all pieces of evidence are *conditionally independent* given any of the hypotheses discerned.

The two simplifying assumptions taken together allow for computing the probabilities $\Pr(h_i \mid \boldsymbol{e})$ for all $h_i \in \boldsymbol{H}$ given observed evidence $\boldsymbol{e} = \{e_{j_1}, \ldots, e_{j_p}\}$, $1 \leq p \leq m$, from

$$\begin{aligned}
\Pr(h_i \mid e_{j_1} \wedge \cdots \wedge e_{j_p}) &= \frac{\Pr(e_{j_1} \wedge \cdots \wedge e_{j_p} \mid h_i) \cdot \Pr(h_i)}{\sum_{k=1}^{n} \Pr(e_{j_1} \wedge \cdots \wedge e_{j_p} \mid h_k) \cdot \Pr(h_k)} = \\
&= \frac{\Pr(e_{j_1} \mid h_i) \cdots \Pr(e_{j_p} \mid h_i) \cdot \Pr(h_i)}{\sum_{k=1}^{n} \Pr(e_{j_1} \mid h_k) \cdots \Pr(e_{j_p} \mid h_k) \cdot \Pr(h_k)}
\end{aligned}$$

It will be evident that for any diagnostic problem $\boldsymbol{e}$ now only $n - 1$ probabilities have to be computed, and that for this purpose only $m \cdot n$ conditional probabilities and $n - 1$ prior ones have to be stored.

Quasi-Probabilistic Reasoning

The systems for automated reasoning with uncertainty constructed in the 1960s were rather small-scaled: they were devised for clear-cut problem domains with only a small number of hypotheses and restricted evidence. For these small systems, all probabilities necessary for applying Bayes' Theorem could be acquired from statistical analysis of empirical data[1]. Despite the underlying (over-)simplifying assumptions, these systems performed considerably well [De Dombal *et al.*, 1974]. Nevertheless, interest in this naive Bayesian approach to reasoning with uncertainty faded in the late 1960s and early 1970s. One of the reasons for this decline in interest is that the approach was feasible only for highly restricted problem domains. For larger or more complex domains, the above-mentioned simplifying assumptions often were seriously violated, causing degeneration of system behaviour. In addition, for larger domains the approach inevitably became demanding, either computationally or from an assessment point of view.

At this stage, the first diagnostic knowledge-based systems began to emerge from artificial-intelligence research. These systems mostly use *production rules* for representing human (experiential) knowledge in a modular form closely resembling logical implications — production rules are expressions of the form **if** ⟨*condition*⟩ **then** ⟨*conclusion*⟩. These so-called *rule-based expert systems* exhibit 'intelligent' reasoning behaviour by employing a heuristic reasoning algorithm that use the production rules for selective gathering of evidence and for pruning the search space of possible diagnoses. It is this pruning behaviour that renders the rule-based expert systems capable of dealing with larger and complexer problem domains than the early naive-Bayesian systems are. The best-known rule-based expert system developed in the 1970s is the MYCIN system for assisting physicians in the diagnosis and treatment of bacterial infections [Buchanan & Shortliffe, 1984].

In the context of rule-based expert systems, the naive Bayesian approach to reasoning with uncertainty is no longer feasible due to the large number of probabilities to be computed. Since in a rule-based system during problem solving the search space of possible diagnoses is pruned by heuristic as well as probabilistic criteria, it is necessary to compute probabilities for all intermediate results derived by the production rules in addition to the probabilities of the separate hypotheses. To allow for efficient computation of all these

[1]Empirical data were available from (medical) research studies, but not collected or recorded to the extent we are used to today, and certainly not in digital form!

probabilities, a set of computation rules has been designed. These computation rules provide for computing the probability of an (intermediate) result from probabilities associated with the production rules that are used in its derivation; to this end, each production rule is assigned the conditional probability of its conclusion given its condition. Unfortunately, these computation rules do not always accord with the axioms of probability theory and can not even be considered approximation rules for computing probabilities. In the sequel, we will use the phrase *quasi-probabilistic* to refer to this approach. The most well-known illustration of the quasi-probabilistic approach is the *certainty-factor model*, designed originally for dealing with uncertainty in the MYCIN system [Shortliffe & Buchanan, 1984]. The certainty-factor model enjoys widespread use in rule-based expert systems built after MYCIN, even though by now it is widely known that the model is mathematically flawed. The relative success of the model can however be accounted for by its satisfactory behaviour in most applications and by its conceptual and computational simplicity [Van der Gaag, 1994].

To Do or Not to Do: Using Probability Theory

Although the quasi-probabilistic approach to reasoning with uncertainty in knowledge-based systems on the one hand met with considerable success in the artificial-intelligence community, it was criticised severely on the other hand because of its ad-hoc character. The incorrectness of the approach from a mathematical point of view even led to a world-wide debate concerning the appropriateness of probability theory for handling uncertainty in a knowledge-based context.

The *adversaries* of probability theory argue that the theory is not expressive enough to cope with the different kinds of uncertainty that are encountered in real-life problem domains and therefore have to be dealt with in decision-support systems. As a consequence several other (more or less) mathematical models have been proposed for reasoning with uncertainty. A major trend in plausible reasoning has arisen from the claim that probability theory is not able to capture imprecision or vagueness, notions of uncertainty which are salient in natural language representations. The name of L.A. Zadeh is inseparable from this trend: he was the first to propose *fuzzy set theory* as the point of departure for the development of methods that are able to cope with vague information. *Dempster-Shafer theory* lies at the basis of another major trend in plausible reasoning. The theory was developed by G. Shafer, building on earlier work by A.P. Dempster [Shafer, 1976]. It was motivated by the observation that probability theory is not able to discern between uncertainty and ignorance due to incompleteness of information.

The *advocates* of probability theory, on the other hand, claim that it is provable that probability theory is the only correct way of dealing with uncertainty and that anything that can be done with non-probabilistic methods, can be done equally well using a probability-based method. For this claim often an argument by R.T. Cox is cited [Cox, 1979]: Cox states a simple set of intuitive properties a measure of uncertainty has to satisfy and subsequently shows that the basic axioms of probability theory follow. Here, we will not enter into the debate concerning the appropriateness of probability theory for reasoning with uncertainty in decision-support systems; for a wide range of diverging opinions, the reader is referred to [Cheeseman, 1988] with its ensuing discussions.

Probabilistic Reasoning: Luctor et Emergo

Although the above-mentioned debate was not in the least subdued, in the mid-1980s the *probabilistic network framework* was introduced as a novel approach to applying probability theory for reasoning with uncertainty in knowledge-based systems [Pearl, 1988]. The probabilistic network framework is characterised by a powerful formalism for representing domain knowledge and the uncertainties that go with it — more in specific, the formalism provides for a concise representation of a multi-variate joint probability distribution on a set of random variables by combining a graph with local conditional probability distributions over small sets of variables. Associated with this formalism are algorithms for efficiently computing probabilities of interest and for processing evidence; these algorithms constitute the basic building blocks for reasoning with knowledge represented in the formalism. Probabilistic networks can both be handcrafted, using knowledge of the domain, and learned from data.

When compared to the naive-Bayesian approach on the one hand and the quasi-probabilistic approach on the other hand, the *probabilistic network approach* offers advantages over both. In contrast with the quasi-probabilistic approach, the probabilistic network approach has a firm mathematical foundation in probability theory. Contrasting the naive-Bayesian approach, the probabilistic network approach circumvents the need for simplifying assumptions by capturing and reasoning about actual independences among variables.

Focus of this Syllabus

Since the early 2000s, the general family of graph-based probabilistic networks is known by the phrase *Probabilistic Graphical models* (PGMs); the family member that assumes discrete variables and a directed graph is the topic of this syllabus and mostly referred to as a *Bayesian Network*[2]. Since its introduction, the Bayesian network has rapidly gained in popularity and by now illustrates its worth in complex problem domains: practical applications have for example been developed for medical diagnosis and prognosis [Andreassen *et al.*, 1987, Heckerman *et al.*, 1992, Blanco *et al.*, 2005], information retrieval [Bruza & Van der Gaag, 1994], in computer vision [Jensen *et al.*, 1990], forensic science [Taroni *et al.*, 2006] and various other domains [Pourret, Naim & Marcot, 2008].

Traditionally, Bayesian networks have been used as a *probabilistic modelling tool* that allows for modelling and reasoning under uncertainty in a complex problem domain. The graph associated with a Bayesian network is generally considered to be an intuitive representation of the problem domain [Dal *et al.*, 2018, p. 97], facilitating the manual construction as well as explanation of such a network for a real-life application. Indeed, earlier applications of Bayesian networks are mostly handcrafted with the help of domain experts. The increasing availability of large data sets has made it much easier to construct applications directly from data [Neapolitan, 2003]. Even large data sets, however, usually do not contain sufficient reliable information to construct reliable networks of general topology. As a result, network engineers still have to rely on domain expertise to complete the network [Druzdzel & Van der Gaag, 2000], or accept a less accurate model by resorting to the use of various types of classifier [Friedman *et al.*, 1997]. From the perspective of explainable AI, classifiers with restricted topology that violates certain independence properties from the domain are less preferable, even if their performance is acceptable.

With the growing interest in machine learning and data mining, Bayesian networks, and PGMs more in general, have also become popular as *statistical data analysis tool*: learning from data can provide insight in the relations between measured variables. The Probabilistic Reasoning course for which this syllabus was written has a focus on Bayesian networks as probabilistic modelling tool. For those interested in their use as statistical analysis tool, we suggest to consider more data-science or data-mining oriented courses.

This syllabus provides a tutorial introduction to the Bayesian network framework and highlights the basics of ongoing research in applying the framework for real-life problem

[2]Other names for Bayesian networks include (Bayesian) belief network, probabilistic network and causal network.

solving. It is organised as follows. Chapter 2 provides some preliminaries from graph theory and from probability theory. In Chapter 3, we discuss the representation of probabilistic independence in graphical models. Chapter 4 introduces the Bayesian network framework: it details the Bayesian network formalism and outlines one of its associated algorithms. In Chapter 5 we address building Bayesian networks for real-life problem domains. Analysis of and problem solving with Bayesian networks is the topic of Chapter 6. The syllabus is concluded with some discussion in Chapter 7 and answers to exercises in Chapter 8.

Exercises

Exercise 1.1

In the naive Bayesian approach a score $s = 10 \cdot \ln O(h \mid \boldsymbol{e})$ *is computed for hypothesis* h *given evidence* $\boldsymbol{e}$*. Show that from this score the posterior probability* $\Pr(h \mid \boldsymbol{e})$ *can be computed as follows:*

$$\Pr(h \mid \boldsymbol{e}) = 1/\big(1 + e^{-(s/10)}\big),$$

where e is Euler's number: the base of the natural logarithm $\ln = {}^{e}\log$.

Exercise 1.2

Consider a set of n *discrete, binary-valued, random variables* V_i*,* $i = 1, \ldots, n$*. Assume you have no knowledge about possible independences among these variables.*

a. *How many probabilities (or: model parameters) are required to specify the full joint probability distribution over these* n *variables?*

b. *How many of these model parameters are* free[3]*?*

* c. *Suppose you want to compute from the joint distribution a probability over only* $m < n$ *variables. How many summations are required for this computation?*

[3]If a discrete distribution requires k model parameters that together sum to one, then $k-1$ of these are *free* in the sense that they can be arbitrarily chosen as long as their sum does not exceed one; the value of the single remaining model parameter is used to ensure that the total mass becomes one and is therefore dictated by the values of the other model parameters.

Chapter 2

Preliminaries

In this section, some concepts from graph theory and from probability theory are reviewed. Our review is tailored to probabilistic graphical models and is not meant to be exhaustive; for further information, any introductory textbooks on graph theory and probability theory, respectively, will suffice.

Notation To enable unambiguous descriptions of all concepts and theory we require notations. The literature on probabilistic graphical models uses a variety of notations and very little conventions. For the material in this syllabus we try to match notations in the original literature as much as possible, while at the same time using consistent and contemporary notation. Boxes are used to summarize general properties of notations:

Boldfaced capital letters, e.g. $\boldsymbol{V}$, are typically used to indicate sets; an element of such a set is also denoted by an upper case letter, e.g. $V \in \boldsymbol{V}$.

Specific notations can be tracked back to their definition through the index at the back of this syllabus.

2.1 Graph Theory

Generally two types of graph are discerned: undirected and directed ones.

Definition 2.1.1 *An* undirected graph *G is a pair $G = (\boldsymbol{V}_G, \boldsymbol{E}_G)$ where $\boldsymbol{V}_G$ is a finite set of* vertices *(also called nodes) and $\boldsymbol{E}_G$ is a set of unordered pairs (V_i, V_j), $V_i, V_j \in \boldsymbol{V}_G$, called* edges.

A directed graph*, or* digraph *for short, is a pair $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ where $\boldsymbol{V}_G$ is a finite set of vertices and $\boldsymbol{A}_G$ is a set of ordered pairs (V_i, V_j), $V_i, V_j \in \boldsymbol{V}_G$, called* arcs*. An arc (V_i, V_j) is often written as $V_i \rightarrow V_j$ or $V_j \leftarrow V_i$.*

For a vertex in a graph, different sets of related vertices can be identified.

Definition 2.1.2 *In a digraph G, vertex V_j is called a* predecessor *(or parent) of vertex V_i if $(V_j, V_i) \in \boldsymbol{A}_G$; the set of all predecessors of vertex V_i in G is denoted as $\boldsymbol{\rho}_G(V_i)$. Likewise, vertex V_j is called a* successor *(or child) of vertex V_i if $(V_i, V_j) \in \boldsymbol{A}_G$; the set of all successors of vertex V_i in G is denoted as $\boldsymbol{\sigma}_G(V_i)$. The reflexive transitive closure*[1] *of V_i under the predecessor relation is denoted as $\boldsymbol{\rho}^*_G(V_i)$; an element from $\boldsymbol{\rho}^*_G(V_i)$ is called an* ancestor *of V_i. Similarly, $\boldsymbol{\sigma}^*_G(V_i)$ denotes the* descendants *of V_i.*

[1]The reflexive closure of set $\boldsymbol{A}$ under r is $r^0(\boldsymbol{A}) = \boldsymbol{A}$, the transitive closure is $r^+(\boldsymbol{A}) = r(\boldsymbol{A}) \cup r^+(r(\boldsymbol{A}))$, and both combined gives $r^*(\boldsymbol{A}) = r^0(\boldsymbol{A}) \cup r^+(\boldsymbol{A})$.

The set of neighbours *of vertex V_i is defined as*

$$\boldsymbol{\nu}_G(V_i) = \begin{cases} \boldsymbol{\sigma}_G(V_i) \cup \boldsymbol{\rho}_G(V_i) & \text{if } G \text{ is directed;} \\ \{V_j \mid (V_i, V_j) \in \boldsymbol{E}_G\} & \text{if } G \text{ is undirected} \end{cases}$$

The size of the neighbour-set of a vertex is called its degree*. In case of a vertex in a digraph, we in addition define the* in-degree *to be its number of predecessors and the* out-degree *to be the number of its successors; the incoming and outgoing arcs together are called its* incident arcs*.*

We often drop the subscript G from $\boldsymbol{\rho}_G$ etc. as long as ambiguity cannot occur. The following definition introduces several types of vertex sequence for undirected graphs.

Definition 2.1.3 *Let $G = (\boldsymbol{V}_G, \boldsymbol{E}_G)$ be an undirected graph. A* path *from vertex V_0 to vertex V_k in G is a sequence of vertices $V_0, \ldots, V_k$, $k \geq 0$, with distinct edges $(V_{i-1}, V_i) \in \boldsymbol{E}_G$, $i = 1, \ldots, k$, between them; k is called the* length *of the path. A path is called* simple *if all vertices are distinct. A* cycle *is a path $V_0, \ldots, V_k, V_0$ from V_0 to V_0 of non-zero length. The graph G is called* cyclic *if it contains at least one cycle; otherwise, it is called* acyclic*.*

In undirected graphs, self-loops (an edge (V_0, V_0)) are generally not allowed. The concepts of path and cycle introduced for undirected graphs directly apply to directed graphs by considering arcs rather than edges. Unless stated otherwise, we typically assume paths to be simple.

We now introduce the concept of *underlying graph*. This concept associates an undirected graph with a directed one. We thereby assume that directed graphs do not contain self-loops either, although this is not a convention.

Definition 2.1.4 *Let $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ be a digraph. The* underlying graph *H of G is the undirected graph $H = (\boldsymbol{V}_H, \boldsymbol{E}_H)$ where $\boldsymbol{V}_H = \boldsymbol{V}_G$ and $\boldsymbol{E}_H$ is obtained from $\boldsymbol{A}_G$ by replacing each arc $(V_i, V_j) \in \boldsymbol{A}_G$ by an edge (V_i, V_j).*

Related to a digraph's underlying graph, we introduce two additional types of vertex sequence for digraphs.

Definition 2.1.5 *Let G be a digraph and let H be its underlying graph. A* chain *from vertex V_0 to vertex V_k in G is a sequence of vertices $V_0, \ldots, V_k$, $k \geq 0$, that is a path in the underlying graph H of G; k is called the* length *of the chain. A* loop *in G is a sequence of vertices that is a cycle in the underlying graph H of G.*

Note that, in a digraph, the concept of path takes the directions of the arcs into account, while the concept of chain does not. A digraph is therefore *acyclic* if it contains no directed cycles; an acyclic digraph (or DAG) can contain loops, however.

In a directed graph, two vertices may be connected by a chain. If this property holds for any two vertices in a digraph, we say that the graph is *connected*.

Definition 2.1.6 *A digraph G is* connected *if there exists at least one chain between any two vertices in G; otherwise, it is called* unconnected*.*

We have introduced the concept of connectedness to apply to directed graphs; the concept, however, is easily extended to apply to undirected graphs.

We now distinguish between several types of digraph.

Definition 2.1.7 *A digraph G is called* singly connected *if it does not contain any loops; otherwise, it is called* multiply connected*. A singly connected digraph G is called a* directed tree *if each vertex in G has at most one predecessor.*

Note that in a singly connected digraph, there is at most one chain between any two vertices; this property does not hold for multiply connected digraphs.

To conclude, we introduce the concept of a *subgraph.* The concept is introduced for undirected graphs, but is extended straightforwardly to apply to digraphs.

Definition 2.1.8 *Let* $G = (\boldsymbol{V}_G, \boldsymbol{E}_G)$ *be an undirected graph. The* subgraph H induced *by* $\boldsymbol{V} \subseteq \boldsymbol{V}_G$ *is the undirected graph* $H = (\boldsymbol{V}, (\boldsymbol{V} \times \boldsymbol{V}) \cap \boldsymbol{E}_G)$.

Note that a subgraph induced by a set of vertices $\boldsymbol{V}$ takes from the original graph *all* edges existent among the vertices from $\boldsymbol{V}$.

2.2 Probability Theory

2.2.1 Set-Theoretic View

In (introductory) literature, probability theory is often approached from a *set-theoretic* point of view. Probability distributions are then defined on *sets* of elements that represent events. All possible outcomes of an experiment (for example, the possible outcomes of rolling a die) are given by the sample space Ω and each event A is a subset of Ω. A probability measure/function/distribution then is a function from events to the $[0, 1]$ interval. As events are sets, combinations of events can be expressed using operations on sets such as union ($\cup$) and intersection ($\cap$). Outcomes of an experiment are often *coded* by using a *random variable* (also: *statistical/ stochastic variable*) which is a function from the sample space to another space (such as reals). By writing probability distributions on random variables, the notation suppresses references to the actual sample space. However, as a statement about a random variable defines an event, there is no actual difference.

2.2.2 Algebraic View

In probabilistic graphical models we typically assume domain and range of random variables to be the same. For a variable V defined on outcomes *heads* and *tails*, for example, we therefore have $V(\textit{heads}) = \textit{heads}$ and $V(\textit{tails}) = \textit{tails}$. We now simply say that V can have or take on one of the *values heads* and *tails*, in which case we write $V = \textit{heads}$ or $V = \textit{tails}$ as possible *value-assignments*. Note that each value-assignment can be seen as an event that either does or does not occur; alternatively, it can be viewed as a *logical proposition* that is either true or false. The PGM community now approaches probability theory from an algebraic point of view by associating probabilities with logical propositions instead of with sets.

We use lower case letters to denote outcomes of random variables. A value-assignment of an outcome to a variable is often abbreviated to just the outcome, if no confusion is possible. For example, $V = \textit{heads}$ may be abbreviated to just *heads*.
An *arbitrary* value-assignment to variable V is denoted by c_V. This notation reflects that a (joint) value-assignment to one or more variables is often called a *configuration*.
For a *binary-valued* variable V, taking on either the value *true* or the value *false*, the two configurations $V = \textit{true}$ and $V = \textit{false}$ are typically denoted v and $\neg v$, respectively.
An arbitrary configuration for a set of variables $\boldsymbol{V}$ is denoted $c_{\boldsymbol{V}}$.A boldface lower case $\boldsymbol{v}$ is sometimes used to indicate a specific configuration for the set $\boldsymbol{V}$.

In this syllabus, we consider a set of *random variables* $\boldsymbol{V} = \{V_1, \ldots, V_n\}$, $n \geq 1$. For ease of exposition, we will often restrict the discussion to binary-valued variables; the generalisation to variables with more than two discrete values, however, is rather straightforward. The set of variables $\boldsymbol{V}$ may be looked upon as spanning a *Boolean algebra of propositions* $\mathcal{V}$. Informally speaking, this algebra comprises all logical propositions that are built from value assignments to the variables discerned. More formally, the Boolean algebra of propositions $\mathcal{V}$ spanned by $\boldsymbol{V}$ is the set of logical propositions consisting of the constant propositions True and False[2], the atomic proposition v_i for all $V_i \in \boldsymbol{V}$, and all compound propositions that are constructed from these by applying the binary operators $\wedge$ (*conjunction*), $\vee$ (*disjunction*), and the unary operator $\neg$ (*negation*); the elements of the algebra $\mathcal{V}$ adhere to the usual axioms of propositional logic.

We now define a *joint probability distribution* as a function on a Boolean algebra of propositions that is spanned by a set of random variables.

Definition 2.2.1 *Let $\boldsymbol{V}$ be a set of random variables and let $\mathcal{V}$ be the Boolean algebra of propositions spanned by $\boldsymbol{V}$. Let $\Pr : \mathcal{V} \to [0,1]$ be a function such that*

- $\Pr(x) \geq 0$*, for all $x \in \mathcal{V}$, and* $\Pr(\mathsf{False}) = 0$*, more in specific;*
- $\Pr(\mathsf{True}) = 1$*;*
- $\Pr(x \vee y) = \Pr(x) + \Pr(y)$*, for all $x, y \in \mathcal{V}$ such that $x \wedge y \equiv \mathsf{False}$.*

Then, $\Pr$ *is called a* joint probability distribution *on $\boldsymbol{V}$. For each $x \in \mathcal{V}$, the function value* $\Pr(x)$ *is termed the* probability *of x.*

A probability $\Pr(x)$ for a logical proposition x expresses the amount of certainty concerning the truth of x. Note that in the previous definition we have indeed associated probabilities with logical propositions instead of with sets. It can easily be shown, however, that the probability of an event (a set of outcomes) is equivalent to the probability of the truth of the proposition asserting the occurrence of the event [Finetti, 1970].

Example 2.2.2 *Suppose X and Y are random variables representing a coin toss. Let A be the event that X = heads and Y = tails then the probability of this event is*

- *from a set-theoretic point of view: the probability of event A, written* $\Pr(A)$*;*
- *from an algebraic point of view: the probability that X = heads and Y = tails are both true propositions, written* $\Pr(X = \mathit{heads} \wedge Y = \mathit{tails})$

If X = heads and Y = tails were considered two separate events A and B, then this would make no difference from the algebraic point of view, but in the set-theoretic approach we should now write $\Pr(A \cap B)$*.*

In the sequel, we will want to single out *strictly positive* joint probability distributions as these have some interesting properties. Strictly positive distributions for example are well-known for their not embedding any functional or logical relationships among their variables.

Definition 2.2.3 *Let $\boldsymbol{V}$ be a set of random variables and let $\mathcal{V}$ be the Boolean algebra of propositions spanned by $\boldsymbol{V}$. Let $\Pr$ be a joint probability distribution on $\boldsymbol{V}$.* $\Pr$ *is* strictly positive *if* $\Pr(x) = 0$ *implies $x \equiv \mathsf{False}$.*

We now introduce the concept of conditional probability.

[2]Note the difference between these *propositions* and the afore mentioned *outcomes/values*!

Definition 2.2.4 *Let $\boldsymbol{V}$ be a set of random variables and let $\mathcal{V}$ be the Boolean algebra of propositions spanned by $\boldsymbol{V}$. Let* Pr *be a joint probability distribution on $\boldsymbol{V}$. For each $x, y \in \mathcal{V}$ with $\Pr(y) > 0$, the* conditional probability *of x given y, denoted as $\Pr(x \mid y)$, is defined as*

$$\Pr(x \mid y) = \frac{\Pr(x \wedge y)}{\Pr(y)}$$

The conditional probability $\Pr(x \mid y)$ expresses the amount of certainty concerning the truth of x *given* that the information y is *known with certainty*. Note that a conditional probability $\Pr(x \mid y) = p$ does not mean that whenever y is known to be true, the probability of x equals p: it means that the probability of x equals p if y is known to be true and *nothing else is known that may affect the certainty concerning the truth of x*. In the sequel, we will assume that all conditional probabilities specified are properly defined, that is, for each conditional probability $\Pr(x \mid y)$, we will *implicitly assume that* $\Pr(y) > 0$. We further state without proof that for a given element $y \in \mathcal{V}$, the conditional probabilities $\Pr(x \mid y)$ for all $x \in \mathcal{V}$ once more constitute a probability distribution on $\boldsymbol{V}$; this probability distribution is called the *conditional probability distribution given y* and will sometimes be denoted as $\Pr^y$. A conditional probability distribution is sometimes referred to as a *posterior* probability distribution; the joint probability distribution it is obtained from then in contrast is referred to as the *prior* distribution.

The following definition introduces the concept of independence of propositions.

Definition 2.2.5 *Let $\boldsymbol{V}$ be a set of random variables and let $\mathcal{V}$ be the Boolean algebra of propositions spanned by $\boldsymbol{V}$. Let* Pr *be a joint probability distribution on $\boldsymbol{V}$. Then, two propositions $x, y \in \mathcal{V}$ are called* (mutually) independent *in* Pr *if*

$$\Pr(x \wedge y) = \Pr(x) \cdot \Pr(y)$$

otherwise, x and y are called dependent *in* Pr*. Two propositions $x, y \in \mathcal{V}$ are called* conditionally independent *given the proposition $z \in \mathcal{V}$ in* Pr *if*

$$\Pr(x \wedge y \mid z) = \Pr(x \mid z) \cdot \Pr(y \mid z)$$

otherwise, x and y are called conditionally dependent *given z in* Pr*.*

In the sequel, we will make extensive use of various well-known properties of joint probability distributions. Before stating these properties, we provide some additional concepts and notational conventions.

Recall that so far we have built on the Boolean algebra of propositions $\mathcal{V}$ spanned by some set of random variables $\boldsymbol{V}$. In the sequel we will consider propositions from the Boolean algebra spanned by $\boldsymbol{V}$ without explicitly referring to $\mathcal{V}$. More specifically, we will typically consider (arbitrary) configurations $c_{\boldsymbol{V}}$, i.e. conjunctions of value assignments, to all variables from a set $\boldsymbol{V}$. Rather than describing a joint probability distribution on $\boldsymbol{V}$ in terms of $\Pr(x)$ for any proposition $x \in \mathcal{V}$, we write $\Pr(\boldsymbol{V})$ to denote the collection of probabilities $\Pr(c_{\boldsymbol{V}})$. In this notation, the upper case letter $\boldsymbol{V}$ is used as a *template* to represent all possible configurations; the distinction between e.g. a set and a template should be clear from the context. Any statement involving a configuration template $\boldsymbol{V}$ is thus taken to hold for any configuration $c_{\boldsymbol{V}}$. If a configuration $c_{\boldsymbol{V}}$ concerns the empty set $\boldsymbol{V} = \varnothing$, then—by convention—$c_{\boldsymbol{V}} \equiv \mathsf{True}$ (no restrictions, anything is possible).

Example 2.2.6 *Consider the set of two binary-valued random variables $\boldsymbol{V} = \{X, Y\}$. The joint probability distribution on $\boldsymbol{V}$ is written as $\Pr(\boldsymbol{V})$, or $\Pr(X \wedge Y)$ where $\boldsymbol{V}$ and $X \wedge Y$ are actually templates that represent all 4 possible configurations $c_{\boldsymbol{V}} \equiv c_X \wedge c_Y$: $x \wedge y$, $x \wedge \neg y$, $\neg x \wedge y$ and $\neg x \wedge \neg y$. The joint distributions are completely defined by the probabilities $\Pr(c_{\boldsymbol{V}}) = \Pr(c_X \wedge c_Y)$. Let $v \equiv x \wedge \neg y$ be a specific configuration, then $\Pr(v) = \Pr(x \wedge \neg y)$.*

The conditional distribution on X given $Y = false$ is written as $\Pr(X \mid \neg y)$ where again X is a template representing the two configurations c_X. $\Pr(X \mid Y)$ indicates a set *of conditional distributions on X: one for each configuration c_Y of Y.*

We now state the various properties that we will use in the sequel. We would like to note that in the literature on probability theory these properties are introduced in many different appearances; we have chosen the form that suits our purposes best. The property stated in the following proposition is known as the *chain rule.*

Proposition 2.2.7 *Let $\boldsymbol{V} = \{V_1, \ldots, V_n\}$, $n \geq 1$, be a set of random variables and let* Pr *be a joint probability distribution on $\boldsymbol{V}$. Then,*

$$\Pr(\boldsymbol{V}) = \Pr(V_1 \wedge \cdots \wedge V_n) = \Pr(V_n \mid V_1 \wedge \cdots \wedge V_{n-1}) \cdot \ldots \cdot \Pr(V_2 \mid V_1) \cdot \Pr(V_1)$$

Note that the expression stated in the previous proposition actually uses the afore-mentioned template notation to capture a whole collection of equalities: the conjunctions only form proper propositions once the variables are assigned a value ($V_1 \wedge V_2$ is not a proposition since V_1 and V_2 are not propositions, but $V_1 = \textit{red} \wedge V_2 = \textit{yellow}$ is). For example, if all variables are binary-valued, the expression represents 2^n equalities, one for each configuration of the set of variables $\boldsymbol{V}$.

The property stated in the following proposition is termed the *marginalisation property.*

Proposition 2.2.8 *Let $\boldsymbol{V}$ be a set of random variables and let* Pr *be a joint probability distribution on $\boldsymbol{V}$. Then,*

$$\Pr(\boldsymbol{X}) = \sum_{c_{\boldsymbol{Y}}} \Pr(\boldsymbol{X} \wedge c_{\boldsymbol{Y}})$$

for all sets of variables $\boldsymbol{X}, \boldsymbol{Y} \subseteq \boldsymbol{V}$.

We state without proof that for any set of variables $\boldsymbol{X} \subseteq \boldsymbol{V}$, the probabilities $\Pr(c_{\boldsymbol{X}})$ for all configurations $c_{\boldsymbol{X}}$ of $\boldsymbol{X}$ once more constitute a joint probability distribution; this probability distribution is termed the *marginal probability distribution* on $\boldsymbol{X}$.

The *conditioning property* is stated in the following proposition.

Proposition 2.2.9 *Let $\boldsymbol{V}$ be a set of random variables and let* Pr *be a joint probability distribution on $\boldsymbol{V}$. Then,*

$$\Pr(\boldsymbol{X}) = \sum_{c_{\boldsymbol{Y}}} \Pr(\boldsymbol{X} \mid c_{\boldsymbol{Y}}) \cdot \Pr(c_{\boldsymbol{Y}})$$

for all sets of variables $\boldsymbol{X}, \boldsymbol{Y} \subseteq \boldsymbol{V}$.

The following theorem is known as *Bayes' Theorem* or *Bayes' rule.*

Theorem 2.2.10 *Let $\boldsymbol{V}$ be a set of random variables and let* Pr *be a joint probability distribution on $\boldsymbol{V}$. Then,*

$$\Pr(\boldsymbol{X} \mid \boldsymbol{Y} \wedge \boldsymbol{Z}) = \frac{\Pr(\boldsymbol{Y} \mid \boldsymbol{X} \wedge \boldsymbol{Z}) \cdot \Pr(\boldsymbol{X} \mid \boldsymbol{Z})}{\Pr(\boldsymbol{Y} \mid \boldsymbol{Z})}$$

for all sets of variables $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z} \subseteq \boldsymbol{V}$.

Note that Bayes' theorem simplifies to the following for $\boldsymbol{Z} = \emptyset$:

$$\Pr(\boldsymbol{X} \mid \boldsymbol{Y}) = \frac{\Pr(\boldsymbol{Y} \mid \boldsymbol{X}) \cdot \Pr(\boldsymbol{X})}{\Pr(\boldsymbol{Y})}$$

To conclude this section, we once more turn to the concept of (conditional) independence. Recall that so far we have taken the concept of independence to apply to *propositions*. We now introduce the concept of independence of *variables*.

Definition 2.2.11 *Let $\boldsymbol{V}$ be a set of random variables and let $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z} \subseteq \boldsymbol{V}$. Let* Pr *be a joint probability distribution on $\boldsymbol{V}$. Then, the set of variables $\boldsymbol{X}$ is called* conditionally independent *of the set of variables $\boldsymbol{Y}$ given the set of variables $\boldsymbol{Z}$ in* Pr *if*

$$\Pr(\boldsymbol{X} \mid \boldsymbol{Y} \wedge \boldsymbol{Z}) = \Pr(\boldsymbol{X} \mid \boldsymbol{Z})$$

otherwise, $\boldsymbol{X}$ is called conditionally dependent *of $\boldsymbol{Y}$ given $\boldsymbol{Z}$ in* Pr*.*

In qualitative terms, the expression $\Pr(\boldsymbol{X} \mid \boldsymbol{Y} \wedge \boldsymbol{Z}) = \Pr(\boldsymbol{X} \mid \boldsymbol{Z})$ indicates that, once information about $\boldsymbol{Z}$ is available, information about $\boldsymbol{Y}$ is irrelevant with respect to $\boldsymbol{X}$. Note that for $\boldsymbol{X}$ and $\boldsymbol{Y}$ to be independent given $\boldsymbol{Z}$, *every* pair of configurations of $\boldsymbol{X}$ and $\boldsymbol{Y}$ has to be independent given *every* configuration of $\boldsymbol{Z}$. Independence of variables therefore implies independence of propositions. The reverse property, however, does *not* hold in general. Also note that the expression from Definition 2.2.11 is asymmetric in $\boldsymbol{X}$ and $\boldsymbol{Y}$. Using Bayes' Theorem, however, it is easily shown that $\Pr(\boldsymbol{X} \mid \boldsymbol{Y} \wedge \boldsymbol{Z}) = \Pr(\boldsymbol{X} \mid \boldsymbol{Z})$ implies $\Pr(\boldsymbol{Y} \mid \boldsymbol{X} \wedge \boldsymbol{Z}) = \Pr(\boldsymbol{Y} \mid \boldsymbol{Z})$.

Exercises

Exercise 2.1

For this exercise, consider the definition of a joint probability distribution (Definition 2.2.1). In addition, consider a binary-valued random variable V with configurations v and $\neg v$, respectively.

a. *Prove that for $x \equiv v$ and $y \equiv \neg v$ the third bullet of the definition follows from the well-known* inclusion/exclusion rule*:*

$$\Pr(x \vee y) = \Pr(x) + \Pr(y) - \Pr(x \wedge y)$$

Note: *the inclusion/exclusion rule for the disjunction of 3 propositions equals*

$$\Pr(x \vee y \vee z) = \Pr(x) + \Pr(y) + \Pr(z) - \Pr(x \wedge y) - \Pr(x \wedge z) - \Pr(y \wedge z) + \Pr(x \wedge y \wedge z)$$

For n propositions this quickly leads to a combinatorial explosion [Comtet, 1974].

b. *Prove that the following property follows from the definition of a joint probability distribution:*

$$\Pr(v) + \Pr(\neg v) = 1$$

*** Exercise 2.2**

Consider a set $\boldsymbol{V}$ of random variables V_i and let $\boldsymbol{E} \subset \boldsymbol{V}$. Match the following notations with their correct semantics:

a.	$\Pr(v_i)$	*I.*	*marginal ('prior') probability*
b.	$\Pr(\boldsymbol{V})$	*II.*	*conditional ('posterior') probability*
c.	$\Pr(V_i)$	*III.*	*joint probability*
d.	$\Pr(\boldsymbol{v})$	*IV.*	*marginal probability distribution*
e.	$\Pr(v_i \mid \boldsymbol{e})$	*V.*	*conditional probability distribution*
f.	$\Pr(V_i \mid \boldsymbol{e})$	*VI.*	*joint probability distribution*
g.	$\Pr(\boldsymbol{V} \mid \boldsymbol{E})$	*VII.*	*set of conditional probability distributions*

***Exercise 2.3**

Prove the following properties for any joint probability distribution, using only definitions and not the properties from this exercise:

a. the chain rule (stated in Proposition 2.2.7);

b. Bayes' Theorem (stated in Theorem 2.2.10);

c. the marginalisation property (stated in Proposition 2.2.8);

d. the conditioning property (stated in Proposition 2.2.9).

Hint: all properties are defined in template form; you can prove them in terms of configurations for the variables as long as the configurations can be considered arbitrary.

***Exercise 2.4**

Let $\boldsymbol{V}$ be a set of random variables and let $\Pr$ *be a joint probability distribution on $\boldsymbol{V}$. Show that*

$$\Pr(\boldsymbol{X} \vee \boldsymbol{Y}) = \Pr(\boldsymbol{X}) + \Pr(\boldsymbol{Y}) - \Pr(\boldsymbol{X} \wedge \boldsymbol{Y})$$

for all sets of variables $\boldsymbol{X}, \boldsymbol{Y} \subseteq \boldsymbol{V}$.

***Exercise 2.5**

Let $\boldsymbol{V}$ be a set of random variables and let $\Pr$ *be a joint probability distribution on $\boldsymbol{V}$. Show that*

$$\Pr(\boldsymbol{X} \mid \boldsymbol{Z}) = \sum_{c_{\boldsymbol{Y}}} \Pr(\boldsymbol{X} \mid c_{\boldsymbol{Y}} \wedge \boldsymbol{Z}) \cdot \Pr(c_{\boldsymbol{Y}} \mid \boldsymbol{Z})$$

for all sets of variables $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z} \subseteq \boldsymbol{V}$.

*** Exercise 2.6**

Let $\boldsymbol{V}$ be a set of random variables and let $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z} \subseteq \boldsymbol{V}$. Let $\Pr$ *be a joint probability distribution on $\boldsymbol{V}$. Show that the set of variables $\boldsymbol{X}$ is conditionally independent of the set of variables $\boldsymbol{Y}$ given the set of variables $\boldsymbol{Z}$ if and only if* $\Pr(\boldsymbol{X} \wedge \boldsymbol{Y} \mid \boldsymbol{Z}) = \Pr(\boldsymbol{X} \mid \boldsymbol{Z}) \cdot \Pr(\boldsymbol{Y} \mid \boldsymbol{Z})$ *and* $\Pr(\boldsymbol{Y} \wedge \boldsymbol{Z}) > 0$.

Part I

PGMs: definition and inference

Chapter 3

Independences and Graphical Representations

The historical background to the framework of Bayesian networks shows various attempts to handle the computational complexity of applying probability theory for reasoning with uncertainty in knowledge-based systems. The concept of (conditional) independence plays a key role in these attempts as knowledge about independences allows for simplifying computations. In this chapter, we address formalisms that allow for a concise representation of an independence relation for effective use in a decision-support system. Current research into independence relations (beyond the scope of this syllabus) is focussed on defining small generating sets [Waal & Van der Gaag, 2005, Bolt & Van der Gaag, 2019], efficient closure computations [Van der Gaag *et al.*, 2018], and on automated construction of graphical representations from them [Baioletti *et al.*, 2011].

3.1 The Concept of Independence Revisited

In most introductory literature on probability theory, the concept of (conditional) independence is introduced in terms of numerical quantities: the independence relation of a joint probability distribution is taken to be implicitly embedded in the probabilities involved. Recall for example that in the previous chapter we have defined two sets of variables $\boldsymbol{X}$ and $\boldsymbol{Y}$ to be conditionally independent given a third set of variables $\boldsymbol{Z}$ if $\Pr(\boldsymbol{X} \mid \boldsymbol{Y} \wedge \boldsymbol{Z}) = \Pr(\boldsymbol{X} \mid \boldsymbol{Z})$. A definition of independence in terms of numbers suggests that, in order to determine whether two sets of variables are (conditionally) independent, several conditional probabilities have to be computed and several equalities have to be tested; moreover, such a definition suggests that for determining independence a joint probability distribution has to be explicitly available for the variables discerned. In contrast, humans tend to be able to state directly, with conviction and consistency, whether or not two sets of variables are independent. Such statements of independence typically are issued qualitatively, without any reference to numerical manipulation of exact probabilities. Based on these observations, we cannot but conclude that the concept of independence is far more basic to human reasoning than its numerical definition suggests. In fact, the definition of independence in terms of probabilities may be looked upon as a *quantitative* way of capturing the basic concept which is *qualitative* in nature. To formalise properties of the qualitative concept of independence, J. Pearl and his co-researchers have designed an *axiomatic system for independence* [Pearl & Paz, 1985, Pearl & Verma, 1987, Geiger & Pearl, 1988]. In this section, we review this axiomatic system.

3.1.1 Pearl's Axiomatic System for Independence

We begin our review of Pearl's axiomatic system for independence by introducing some new terminology and notational convention.

Definition 3.1.1 *Let $\boldsymbol{V}$ be a set of random variables and let* Pr *be a joint probability distribution on $\boldsymbol{V}$. Then, the* independence relation $I_{\mathrm{Pr}} \subseteq \mathcal{P}(\boldsymbol{V}) \times \mathcal{P}(\boldsymbol{V}) \times \mathcal{P}(\boldsymbol{V})$ *of* Pr *is defined by $(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y}) \in I_{\mathrm{Pr}}$ if and only if* $\Pr(\boldsymbol{X} \mid \boldsymbol{Y} \wedge \boldsymbol{Z}) = \Pr(\boldsymbol{X} \mid \boldsymbol{Z})$ *for all sets of variables $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z} \subseteq \boldsymbol{V}$.*

In the sequel, we will write $I_{\mathrm{Pr}}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y})$ to denote $(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y}) \in I_{\mathrm{Pr}}$ and $\neg I_{\mathrm{Pr}}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y})$ to denote $(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y}) \notin I_{\mathrm{Pr}}$. A statement $I_{\mathrm{Pr}}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y})$ of a joint probability distribution's independence relation I_{Pr} is termed an *independence statement*. In qualitative terms, an independence statement $I_{\mathrm{Pr}}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y})$ expresses that in the context of information about $\boldsymbol{Z}$, information about $\boldsymbol{Y}$ is irrelevant with respect to $\boldsymbol{X}$. The above definition allows for stating some trivial but convenient independence statements, such as $I_{\mathrm{Pr}}(\boldsymbol{X}, \boldsymbol{X}, \boldsymbol{Y})$, which holds iff $\Pr(\boldsymbol{X} \mid \boldsymbol{Y} \wedge \boldsymbol{X}) = \Pr(\boldsymbol{X} \mid \boldsymbol{X})$, i.e. $1 = 1$; its symmetric version $I_{\mathrm{Pr}}(\boldsymbol{Y}, \boldsymbol{X}, \boldsymbol{X})$ is also trivially true since $I_{\mathrm{Pr}}(\boldsymbol{Y}, \boldsymbol{X}, \boldsymbol{X})$ iff $\Pr(\boldsymbol{Y} \mid \boldsymbol{X} \wedge \boldsymbol{X}) = \Pr(\boldsymbol{Y} \mid \boldsymbol{X})$.

In designing his axiomatic system for independence, Pearl builds on a set of properties that are satisfied by any joint probability distribution's independence relation; Theorem 3.1.2 reviews these properties.

Theorem 3.1.2 *Let $\boldsymbol{V}$ be a set of random variables. Let* Pr *be a joint probability distribution on $\boldsymbol{V}$ and let I_{Pr} be its independence relation. Then, I_{Pr} satisfies the properties*

- $I_{\mathrm{Pr}}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y}) \to I_{\mathrm{Pr}}(\boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{X})$; *(symmetry)*
- $I_{\mathrm{Pr}}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y} \cup \boldsymbol{W}) \to I_{\mathrm{Pr}}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y}) \wedge I_{\mathrm{Pr}}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{W})$; *(decomposition)*
- $I_{\mathrm{Pr}}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y} \cup \boldsymbol{W}) \to I_{\mathrm{Pr}}(\boldsymbol{X}, \boldsymbol{Z} \cup \boldsymbol{W}, \boldsymbol{Y})$; *(weak union)*
- $I_{\mathrm{Pr}}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y}) \wedge I_{\mathrm{Pr}}(\boldsymbol{X}, \boldsymbol{Z} \cup \boldsymbol{Y}, \boldsymbol{W}) \to I_{\mathrm{Pr}}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y} \cup \boldsymbol{W})$; *(contraction)*

for all mutually disjoint sets of variables $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{W} \subseteq \boldsymbol{V}$. If the distribution Pr *is strictly positive, then I_{Pr} satisfies the additional property*

- $I_{\mathrm{Pr}}(\boldsymbol{X}, \boldsymbol{Z} \cup \boldsymbol{W}, \boldsymbol{Y}) \wedge I_{\mathrm{Pr}}(\boldsymbol{X}, \boldsymbol{Z} \cup \boldsymbol{Y}, \boldsymbol{W}) \to I_{\mathrm{Pr}}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y} \cup \boldsymbol{W})$; *(intersection)*

for all mutually disjoint sets of variables $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{W} \subseteq \boldsymbol{V}$.

The properties stated in the previous theorem are easily verified from the basic axioms of probability theory. We would like to note that we have closely followed Pearl by stating the properties in the theorem to hold for *mutually disjoint* sets of variables only [Pearl, 1988]. These properties, however, also hold for overlapping sets of variables [Van der Gaag & Meyer, 1998], as well as empty sets. Moreover, it can be easily shown that the symmetry, contraction and intersection properties can actually be stated as a bi-implication ($\leftrightarrow$). The opposite of decomposition, called the *composition* property, $I_{\mathrm{Pr}}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y}) \wedge I_{\mathrm{Pr}}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{W}) \to I_{\mathrm{Pr}}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y} \cup \boldsymbol{W})$ is satisfied by many independence relations I_{Pr}, but not all [Gasse & Aussem, 2016].

Pearl now takes the properties stated in Theorem 3.1.2 as *axioms* for the qualitative concept of independence [Pearl, 1988]. Following the properties for I_{Pr}, Pearl assumed for each axiom that the sets of variables involved are mutually disjoint. Given the insight that the properties also hold for overlapping sets, we will lift the assumption of mutual disjointness in the next and all following definitions involving independence relations. The following now defines informational independence:

Definition 3.1.3 *Let $\boldsymbol{V}$ be a set of random variables. A* semi-graphoid independence relation *on $\boldsymbol{V}$ is a ternary relation $I \subseteq \mathcal{P}(\boldsymbol{V}) \times \mathcal{P}(\boldsymbol{V}) \times \mathcal{P}(\boldsymbol{V})$ such that I satisfies the properties*

- $I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y}) \rightarrow I(\boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{X})$;
- $I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y} \cup \boldsymbol{W}) \rightarrow I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y}) \wedge I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{W})$;
- $I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y} \cup \boldsymbol{W}) \rightarrow I(\boldsymbol{X}, \boldsymbol{Z} \cup \boldsymbol{W}, \boldsymbol{Y})$;
- $I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y}) \wedge I(\boldsymbol{X}, \boldsymbol{Z} \cup \boldsymbol{Y}, \boldsymbol{W}) \rightarrow I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y} \cup \boldsymbol{W})$;

for all sets of variables $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{W} \subseteq \boldsymbol{V}$. A graphoid independence relation *I on $\boldsymbol{V}$ is a semi-graphoid independence relation on $\boldsymbol{V}$ such that I satisfies the additional property*

- $I(\boldsymbol{X}, \boldsymbol{Z} \cup \boldsymbol{W}, \boldsymbol{Y}) \wedge I(\boldsymbol{X}, \boldsymbol{Z} \cup \boldsymbol{Y}, \boldsymbol{W}) \rightarrow I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y} \cup \boldsymbol{W})$;

for all sets of variables $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{W} \subseteq \boldsymbol{V}$.

The properties described in the previous definition with each other convey the idea that learning irrelevant information does not alter the independences among the variables discerned [Pearl, 1988]. We consider the qualitative meanings of the various properties separately.

The property

$$I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y}) \rightarrow I(\boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{X})$$

for all sets of variables $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z} \subseteq \boldsymbol{V}$, states that if information about $\boldsymbol{Y}$ is deemed irrelevant with respect to $\boldsymbol{X}$ in the context of some information about $\boldsymbol{Z}$, then information about $\boldsymbol{X}$ must be irrelevant with respect to $\boldsymbol{Y}$ in this context; this property is called the *symmetry axiom.*

The property

$$I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y} \cup \boldsymbol{W}) \rightarrow I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y}) \wedge I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{W})$$

for all sets of variables $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{W} \subseteq \boldsymbol{V}$, asserts that if information about both $\boldsymbol{Y}$ and $\boldsymbol{W}$ is judged irrelevant with respect to $\boldsymbol{X}$, then both information about $\boldsymbol{Y}$ and information about $\boldsymbol{W}$ must be irrelevant with respect to $\boldsymbol{X}$ separately; this property is known as the *decomposition axiom.* We would like to note that the decomposition axiom may be reformulated as $I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y} \cup \boldsymbol{W}) \rightarrow I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y})$; we have chosen, however, to use Pearl's original formulation because it conveys the idea of decomposition more clearly.

The property

$$I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y} \cup \boldsymbol{W}) \rightarrow I(\boldsymbol{X}, \boldsymbol{Z} \cup \boldsymbol{W}, \boldsymbol{Y})$$

for all sets of variables $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{W} \subseteq \boldsymbol{V}$, states that learning information about $\boldsymbol{W}$ that is known to be irrelevant with respect to $\boldsymbol{X}$ cannot help irrelevant information about $\boldsymbol{Y}$ to become relevant with respect to $\boldsymbol{X}$; this property is known as the *weak union axiom.*

The property

$$I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y}) \wedge I(\boldsymbol{X}, \boldsymbol{Z} \cup \boldsymbol{Y}, \boldsymbol{W}) \rightarrow I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y} \cup \boldsymbol{W})$$

for all sets of variables $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{W} \subseteq \boldsymbol{V}$, states that if we judge information about $\boldsymbol{W}$ to be irrelevant with respect to $\boldsymbol{X}$ after learning some irrelevant information about $\boldsymbol{Y}$, then the information about $\boldsymbol{W}$ must have been irrelevant with respect to $\boldsymbol{X}$ before we learned $\boldsymbol{Y}$; this property is known as the *contraction axiom.* Note that the contraction axiom can be reformulated as $I(\boldsymbol{X},\boldsymbol{Z},\boldsymbol{Y}) \rightarrow (I(\boldsymbol{X},\boldsymbol{Z}\cup\boldsymbol{Y},\boldsymbol{W}) \rightarrow I(\boldsymbol{X},\boldsymbol{Z},\boldsymbol{Y}\cup\boldsymbol{W}))$. From this reformulation, it is seen that the axiom can be looked upon as a *conditional* reverse of the weak union axiom.

We now consider the property

$$I(\boldsymbol{X},\boldsymbol{Z}\cup\boldsymbol{W},\boldsymbol{Y}) \wedge I(\boldsymbol{X},\boldsymbol{Z}\cup\boldsymbol{Y},\boldsymbol{W}) \rightarrow I(\boldsymbol{X},\boldsymbol{Z},\boldsymbol{Y}\cup\boldsymbol{W})$$

for all sets of variables $\boldsymbol{X},\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{W} \subseteq \boldsymbol{V}$, for graphoid independence relations. This property states that if, in the context of some information about $\boldsymbol{Z}$, learning information about $\boldsymbol{W}$ renders information about $\boldsymbol{Y}$ irrelevant with respect to $\boldsymbol{X}$ and learning $\boldsymbol{Y}$ renders $\boldsymbol{W}$ irrelevant with respect to $\boldsymbol{X}$, then the information about both $\boldsymbol{Y}$ and $\boldsymbol{W}$ must be irrelevant with respect to $\boldsymbol{X}$ given $\boldsymbol{Z}$; this property is known as the *intersection axiom.*

From Definition 3.1.3 and Theorem 3.1.2, we observe that every independence relation that is embedded in a joint probability distribution is a semi-graphoid independence relation; this property is stated more formally in the following corollary.

Corollary 3.1.4 *Let $\boldsymbol{V}$ be a set of random variables. Let* $\Pr$ *be a joint probability distribution on $\boldsymbol{V}$ and let $I_{\Pr}$ be its independence relation. Then, $I_{\Pr}$ is a semi-graphoid independence relation. Furthermore, if* $\Pr$ *is strictly positive, then $I_{\Pr}$ is a graphoid independence relation.*

Unfortunately, although any probability distribution's independence relation is a semi-graphoid independence relation, the reverse property does *not* hold. There exist semi-graphoid independence relations for which there do not exist joint probability distributions embedding them; for details, we refer to [Van der Gaag & Meyer, 1996, Studený, 1989]. We would like to note that it has been shown that a finite axiomatisation of the concept of probabilistic independence does not exist [Studený, 1992].

3.1.2 Properties of Independence Relations

Using the definition of informational independence, we derive some convenient properties of (semi-graphoid and graphoid) independence relations. The following lemma shows that the symmetry and contraction axioms are easily generalised to bi-implications.

Lemma 3.1.5 *Let $\boldsymbol{V}$ be a set of random variables. Furthermore, let I be a semi-graphoid independence relation on $\boldsymbol{V}$. Then,*

- $I(\boldsymbol{X},\boldsymbol{Z},\boldsymbol{Y}) \leftrightarrow I(\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{X})$*;*
- $I(\boldsymbol{X},\boldsymbol{Z},\boldsymbol{Y}) \wedge I(\boldsymbol{X},\boldsymbol{Z}\cup\boldsymbol{Y},\boldsymbol{W}) \leftrightarrow I(\boldsymbol{X},\boldsymbol{Z},\boldsymbol{Y}\cup\boldsymbol{W})$*;*

for all sets of variables $\boldsymbol{X},\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{W} \subseteq \boldsymbol{V}$.

Proof. We begin our proof by observing that since I is a semi-graphoid independence relation, it obeys the first four axioms stated in Definition 3.1.3. The first property stated in the lemma now follows directly from the symmetry axiom. For the second property, we observe that $I(\boldsymbol{X},\boldsymbol{Z},\boldsymbol{Y}) \wedge I(\boldsymbol{X},\boldsymbol{Z}\cup\boldsymbol{Y},\boldsymbol{W}) \rightarrow I(\boldsymbol{X},\boldsymbol{Z},\boldsymbol{Y}\cup\boldsymbol{W})$ coincides with the contraction axiom and therefore trivially holds for the relation I. We will now prove that $I(\boldsymbol{X},\boldsymbol{Z},\boldsymbol{Y}\cup\boldsymbol{W}) \rightarrow I(\boldsymbol{X},\boldsymbol{Z},\boldsymbol{Y}) \wedge I(\boldsymbol{X},\boldsymbol{Z}\cup\boldsymbol{Y},\boldsymbol{W})$. We have

$$I(\boldsymbol{X},\boldsymbol{Z},\boldsymbol{Y}\cup\boldsymbol{W}) \quad\Rightarrow\quad I(\boldsymbol{X},\boldsymbol{Z},\boldsymbol{Y}) \wedge I(\boldsymbol{X},\boldsymbol{Z},\boldsymbol{W}) \quad\Rightarrow\quad I(\boldsymbol{X},\boldsymbol{Z},\boldsymbol{Y})$$

by the decomposition axiom. In addition, we have

$$I(\boldsymbol{X},\boldsymbol{Z},\boldsymbol{Y}\cup\boldsymbol{W}) \quad\Rightarrow\quad I(\boldsymbol{X},\boldsymbol{Z}\cup\boldsymbol{Y},\boldsymbol{W})$$

by weak union. The property stated in the lemma now follows directly. □

For graphoid independence relations we have that the intersection axiom can also be generalised to a bi-implication.

Lemma 3.1.6 *Let* $\boldsymbol{V}$ *be a set of random variables. Furthermore, let* I *be a graphoid independence relation on* $\boldsymbol{V}$. *Then,*

$$I(\boldsymbol{X}, \boldsymbol{Z} \cup \boldsymbol{W}, \boldsymbol{Y}) \wedge I(\boldsymbol{X}, \boldsymbol{Z} \cup \boldsymbol{Y}, \boldsymbol{W}) \leftrightarrow I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y} \cup \boldsymbol{W})$$

for all sets of variables $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{W} \subseteq \boldsymbol{V}$.

Proof. We will only prove that $I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y} \cup \boldsymbol{W}) \rightarrow I(\boldsymbol{X}, \boldsymbol{Z} \cup \boldsymbol{W}, \boldsymbol{Y}) \wedge I(\boldsymbol{X}, \boldsymbol{Z} \cup \boldsymbol{Y}, \boldsymbol{W})$; the reverse property coincides with the intersection axiom and therefore trivially holds for the independence relation I. We have

$$I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y} \cup \boldsymbol{W}) \Rightarrow I(\boldsymbol{X}, \boldsymbol{Z} \cup \boldsymbol{W}, \boldsymbol{Y})$$

and

$$I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y} \cup \boldsymbol{W}) \Rightarrow I(\boldsymbol{X}, \boldsymbol{Z} \cup \boldsymbol{Y}, \boldsymbol{W})$$

by the weak union axiom. The property stated in the lemma now follows directly. □

In the sequel, we will use the phrase *independence relation* to denote a semi-graphoid independence relation, unless stated otherwise.

3.2 Graphical Representations of Independence

One of the problems in applying probability theory for automated reasoning with uncertainty in a knowledge-based system is the space complexity of representing a joint probability distribution. Since the concept of independence plays a key role in solving this problem, a formalism for representing joint probability distributions should allow for efficiently modelling independences. There are various ways of representing an independence relation. One way, for example, is to enumerate the separate statements of an independence relation explicitly. Such a representation clearly is impractical as the number of tuples in an independence relation can be astronomical. Another way is to make use of the axioms from Definition 3.1.3: only the statements from an appropriate subset of the independence relation are enumerated explicitly and all its other statements are defined implicitly by this set and the defining axioms. Although exploiting the axioms allows for a far more economical representation of an independence relation than explicit enumeration, it can still require exponential space.

In this section, we consider more concise representations of independence relations, building on the idea of *graphical encoding*, in which graphs are used as a language for communicating probabilistic independence statements. Graphs have no probabilistic meaning by themselves. For representing an independence relation in a graph, therefore, a probabilistic meaning has to be assigned to the topological properties of such a graph, that is, we have to assign a probabilistic meaning to both the vertices of the graph and to its arcs or edges. To this end we will model the *variables* of the independence relation as *vertices* in the graph and use the connection between vertices to describe the *independence statements*. In Section 3.2.1 we address modelling an independence relation in an undirected graph; in Section 3.2.2 we consider encoding an independence relation in the formalism of directed graphs.

3.2.1 Undirected Graphs

To formally capture the probabilistic meaning of an undirected graph, we begin by defining a graphical criterion for reading sets of vertices from a graph that allow for *blocking* all paths between two given sets of vertices; this graphical criterion is termed the *separation criterion* for undirected graphs.

Definition 3.2.1 *Let $G = (\boldsymbol{V}_G, \boldsymbol{E}_G)$ be an undirected graph. Let $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z} \subseteq \boldsymbol{V}_G$ be sets of vertices in G. The set of vertices $\boldsymbol{Z}$ is said to* separate *the sets of vertices $\boldsymbol{X}$ and $\boldsymbol{Y}$ in G, denoted as $\langle \boldsymbol{X} \mid \boldsymbol{Z} \mid \boldsymbol{Y} \rangle_G$, if for each vertex $V_i \in \boldsymbol{X}$ and each vertex $V_j \in \boldsymbol{Y}$ every simple path from V_i to V_j in G contains at least one vertex from $\boldsymbol{Z}$.*

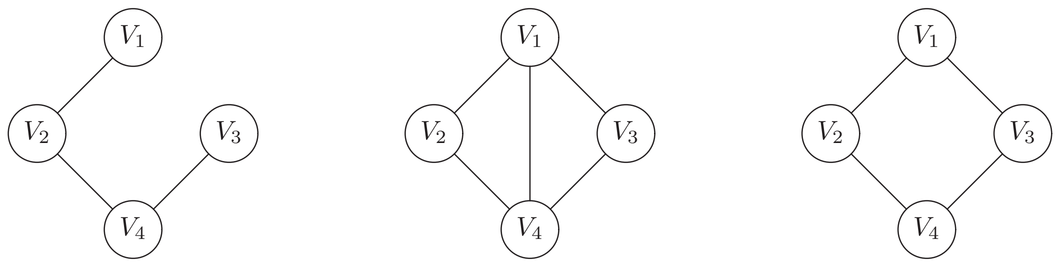
We look upon a separating set as effectively *blocking* influence: if a set of variables $\boldsymbol{Z}$ separates two sets of variables $\boldsymbol{X}$ and $\boldsymbol{Y}$, then $\boldsymbol{Z}$ is blocking any flow of information or influence between $\boldsymbol{X}$ and $\boldsymbol{Y}$.

The relation between the graphical notion of separation in undirected graphs and its probabilistic meaning is given by the following definition.

Definition 3.2.2 *Let $\boldsymbol{V}$ be a set of random variables and let I be an independence relation on $\boldsymbol{V}$. Furthermore, let $G = (\boldsymbol{V}_G, \boldsymbol{E}_G)$ be an undirected graph with $\boldsymbol{V}_G = \boldsymbol{V}$. Then,*

- *the graph G is called an* undirected dependence map*, or* D-map *for short, for I, if for all sets of variables $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z} \subseteq \boldsymbol{V}$, we have: if $I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y})$ then $\langle \boldsymbol{X} \mid \boldsymbol{Z} \mid \boldsymbol{Y} \rangle_G$;*
- *the graph G is called an* undirected independence map*, or* I-map *for short, for I, if for all sets of variables $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z} \subseteq \boldsymbol{V}$, we have: if $\langle \boldsymbol{X} \mid \boldsymbol{Z} \mid \boldsymbol{Y} \rangle_G$ then $I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y})$;*
- *the graph G is called an* undirected perfect map*, or* P-map *for short, for I, if G is both an undirected D-map and an undirected I-map for I.*

From the previous definition we have that the 'language' of undirected graphs is not necessarily apt to faithfully describe an independence relation: only an undirected P-map can perfectly encode such a relation. In an undirected D-map any pair of neighbouring vertices represents a pair of variables that are *de*pendent; however, not every pair of dependent variables needs be represented as a pair of neighbouring vertices. In an undirected D-map, therefore, a pair of non-neighbouring vertices may represent either a pair of dependent variables or a pair of (conditionally) independent variables.



(a) Example D-map (b) Example I-map (c) P-map

Figure 3.1: Undirected graph encodings for the independence relation from Example 3.2.3.

Example 3.2.3 Let $\boldsymbol{V} = \{V_1, V_2, V_3, V_4\}$ be a set of random variables. We consider the independence relation I on $\boldsymbol{V}$ defined by the independence statements $I(\{V_1\}, \{V_2, V_3\}, \{V_4\})$ and $I(\{V_2\}, \{V_1, V_4\}, \{V_3\})$. The undirected graph G shown in Figure 3.1a is an example of an undirected D-map for I, since for each independence statement $I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y})$,

$\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z} \subseteq \boldsymbol{V}$, from the relation I, there exists a matching separation statement $\langle \boldsymbol{X} \mid \boldsymbol{Z} \mid \boldsymbol{Y} \rangle_G$. For example, for the independence statement $I(\{V_1\}, \{V_2, V_3\}, \{V_4\})$ we find that the only path from V_1 to V_4 comprises the vertex V_2 which is included in the set $\{V_2, V_3\}$, that is, $\langle \{V_1\} \mid \{V_2, V_3\} \mid \{V_4\} \rangle_G$. □

In an undirected I-map, any pair of non-neighbouring vertices represents a pair of variables that are (conditionally) independent; however, not every pair of (conditionally) independent variables is represented as a pair of non-neighbouring vertices. In an undirected I-map, therefore, a pair of neighbouring vertices may represent either a pair of dependent variables or a pair of independent variables.

Example 3.2.4 Consider once more the independence relation I from Example 3.2.3. The graph G shown in Figure 3.1b is an example of an undirected I-map for the relation I, since for each separation statement $\langle \boldsymbol{X} \mid \boldsymbol{Z} \mid \boldsymbol{Y} \rangle_G$, $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z} \subseteq \boldsymbol{V}$, read from G, there exists a matching independence statement $I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y})$: for the single separation statement $\langle \{V_2\} \mid \{V_1, V_4\} \mid \{V_3\} \rangle_G$ read from the graph, we find the statement $I(\{V_2\}, \{V_1, V_4\}, \{V_3\})$ in the relation I. □

An undirected P-map for an independence relation faithfully represents all independences as well as all dependences from the relation.

Example 3.2.5 Consider once more the independence relation I from Example 3.2.3. The graph G shown in Figure 3.1c is the only undirected P-map for G. Note that G is an undirected D-map for I as well as an undirected I-map for I. □

From the above observations, we conclude that the various types of map provide for reading different types of probabilistic information from an undirected graph. If the graph is an undirected I-map for an independence relation, we can read independence statements from it as any separation statement is guaranteed to correspond with an independence statement; from an undirected D-map we can read dependence statements, and from an undirected P-map we can read both independence and dependence statements.

We now investigate the expressive power of the formalism of undirected graphs for representing independence relations.

Lemma 3.2.6 *For every independence relation, there exist an undirected D-map and an undirected I-map.*

Proof. We first show that for every independence relation there exists an undirected D-map. To this end, we show that the edgeless undirected graph $G = (\boldsymbol{V}, \varnothing)$ is an undirected D-map for any independence relation I on the set of variables $\boldsymbol{V}$. Since the graph G is edgeless, we have that the property: if $I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y})$ then $\langle \boldsymbol{X} \mid \boldsymbol{Z} \mid \boldsymbol{Y} \rangle_G$, trivially holds for all sets of variables $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z} \subseteq \boldsymbol{V}$. Informally speaking, as the graph G does not represent any dependences, it cannot represent any dependence incorrectly.

We now show that for every independence relation there exists an undirected I-map. To this end, we show that the complete undirected graph $G' = (\boldsymbol{V}, \boldsymbol{V} \times \boldsymbol{V})$ is an undirected I-map for any independence relation I on $\boldsymbol{V}$. Since in this graph every pair of vertices is connected by an edge, we have that the property: if $\langle \boldsymbol{X} \mid \boldsymbol{Z} \mid \boldsymbol{Y} \rangle_G$ then $I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y})$, trivially holds for all sets of variables $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z} \subseteq \boldsymbol{V}$. Informally speaking, as the graph G' does not represent any independences, it cannot represent any independence incorrectly. □

From the previous lemma, we have that every independence relation has an undirected

D-map and an undirected I-map; in fact, an independence relation may have *several* undirected D-maps and I-maps. Unfortunately, a similar property does not hold for undirected P-maps: not every independence relation has an undirected P-map.

Example 3.2.7 Consider an independence relation I on a set of variables $\boldsymbol{V}$ such that $I(\boldsymbol{X}, \boldsymbol{Z_1}, \boldsymbol{Y})$ for some sets of variables $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z_1} \subseteq \boldsymbol{V}$ and $\neg I(\boldsymbol{X}, \boldsymbol{Z_1} \cup \boldsymbol{Z_2}, \boldsymbol{Y})$ for some set $\boldsymbol{Z_2} \subseteq \boldsymbol{V}$ with $\boldsymbol{Z_2} \neq \varnothing$ and $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z_1}, \boldsymbol{Z_2}$ are disjoint. The independence relation I is an example of an independence relation comprising an *induced dependence*: the sets of variables $\boldsymbol{X}$ and $\boldsymbol{Y}$ are independent given $\boldsymbol{Z_1}$, but become dependent if information about $\boldsymbol{Z_2}$ becomes available. Now, for any undirected graph G to be an undirected P-map for this independence relation I, it should be an undirected D-map as well as an undirected I-map for I. For G to be an undirected D-map for I, it should display the set $\boldsymbol{Z_1}$ as a separating set for $\boldsymbol{X}$ and $\boldsymbol{Y}$; on the other hand, for G to be an undirected I-map, it should display $\boldsymbol{Z_1} \cup \boldsymbol{Z_2}$ as *not* separating $\boldsymbol{X}$ and $\boldsymbol{Y}$. We observe, however, that for any undirected graph G the property: if $\langle \boldsymbol{X} \mid \boldsymbol{Z_1} \mid \boldsymbol{Y} \rangle_G$ then $\langle \boldsymbol{X} \mid \boldsymbol{Z_1} \cup \boldsymbol{Z_2} \mid \boldsymbol{Y} \rangle_G$ holds by definition. More in general, if two sets of vertices $\boldsymbol{X}$ and $\boldsymbol{Y}$ are separated by a third set of vertices $\boldsymbol{Z}$ in an undirected graph G, then $\boldsymbol{X}$ and $\boldsymbol{Y}$ are separated by any superset of $\boldsymbol{Z}$ in G as well. From this observation, it will be evident that no undirected graph can satisfy the two requirements mentioned above simultaneously and, hence, that there does not exist an undirected P-map for I. $\square$

Although not every independence relation has an undirected P-map, there are independence relations that can indeed be represented faithfully. An independence relation for which an undirected P-map exists is termed *undirected graph-isomorphic.*

Definition 3.2.8 *An independence relation I is said to be* undirected graph-isomorphic *if there exists an undirected graph G such that G is an undirected P-map for I.*

We would like to note that, if an independence relation is undirected graph-isomorphic, then it allows one and only one undirected P-map. To conclude, we would like to mention that a necessary and sufficient condition has been identified for an independence relation to be undirected graph-isomorphic [Pearl, 1988]; this condition allows for testing whether a given independence relation lends itself to representation in an undirected graph. Here, we will not elaborate any further on this observation.

3.2.2 Directed Graphs

In the previous section, we have addressed the representation of an independence relation in the formalism of undirected graphs. We have seen that not every independence relation can be represented faithfully in this formalism. In this section, we investigate the formalism of *directed graphs (digraphs)* with its increased expressive power as a language for encoding independence relations; we will consider *acyclic* digraphs only.

Just like undirected graphs, do directed graphs not have a probabilistic meaning by themselves and therefore have to be assigned one. For this purpose, we will formulate a graphical criterion similar to the separation criterion for undirected graphs. For digraphs, this criterion is called the *d-separation criterion.* Before defining this criterion, however, we introduce the concept of *blocking* influence among variables. The definition provided here is an enhancement of the original definition by Pearl, based upon subsequent insights [Van der Gaag & Meyer, 1998].

Definition 3.2.9 *Let $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ be an acyclic digraph and let s be a chain in G between $V_i \in \boldsymbol{V}_G$ and $V_j \in \boldsymbol{V}_G$. The chain s is* blocked *by a (possibly empty) set of vertices $\boldsymbol{W} \subseteq \boldsymbol{V}_G$ if $V_i \in \boldsymbol{W}$ or $V_j \in \boldsymbol{W}$, or s contains three consecutive vertices X_1, X_2, X_3, for which (at least) one of the following conditions hold:*

a. arcs (X_2, X_1) and (X_2, X_3) are on the chain s, and $X_2 \in \boldsymbol{W}$;

b. arcs (X_1, X_2) and (X_2, X_3) are on the chain s, and $X_2 \in \boldsymbol{W}$;

c. arcs (X_1, X_2) and (X_3, X_2) are on the chain s, and $\boldsymbol{\sigma}^*_G(X_2) \cap \boldsymbol{W} = \varnothing$.

In defining the concept of a *blocked chain*, we have distinguished three graphical conditions. Figure 3.2 serves as a reference for these conditions; in the two chains representing the conditions *a.* and *b.*, vertex X_2 is drawn with shading to indicate that it is comprised in the blocking set $\boldsymbol{W}$ for the chain at hand.

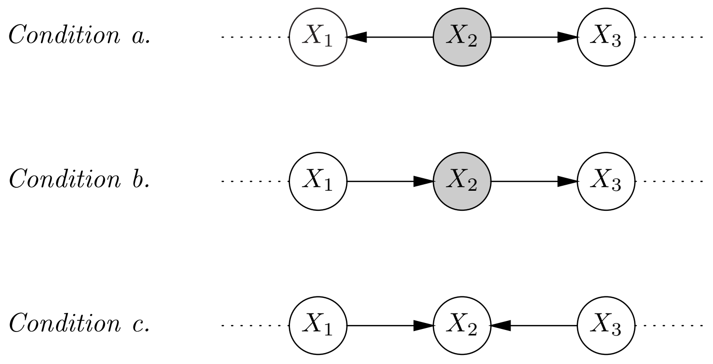


Figure 3.2: Chain Blocking; Shaded Vertices are Members of Blocking-set $\boldsymbol{W}$.

Building on the concept of blocking of single chains, we now define the d-separation criterion for reading from a digraph sets of vertices that allow for blocking *all* chains between two given sets of vertices.

Definition 3.2.10 *Let $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ be an acyclic digraph. Let $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z} \subseteq \boldsymbol{V}_G$ be sets of vertices in G. The set of vertices $\boldsymbol{Z}$ is said to* d-separate *the sets of vertices $\boldsymbol{X}$ and $\boldsymbol{Y}$ in G, denoted as $\langle \boldsymbol{X} \mid \boldsymbol{Z} \mid \boldsymbol{Y} \rangle^d_G$, if for each vertex $V_i \in \boldsymbol{X}$ and each vertex $V_j \in \boldsymbol{Y}$ every chain from V_i to V_j in G is blocked by $\boldsymbol{Z}$.*

We would like to note that in assigning a meaning to the topological properties of a directed graph, we want to distinguish between conditional independences and dependences, that is, between *two* alternatives only. The formalism of directed graphs, however, allows for distinguishing between *three* alternatives since there are three different ways in which two arcs between three vertices can be directed (up to renaming of vertices). This observation accounts for two conditions of the concept of blocking having been assigned the same meaning; these are the first two conditions depicted in Figure 3.2. Note that the third condition models an induced dependence.

The following definition now relates the d-separation criterion to the concept of independence.

Definition 3.2.11 *Let $\boldsymbol{V}$ be a set of random variables and let I be an independence relation on $\boldsymbol{V}$. Furthermore, let $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ be an acyclic digraph with $\boldsymbol{V}_G = \boldsymbol{V}$. Then,*

- *the graph G is called a* directed dependence map*, or* D-map *for short, for I, if for all sets of variables $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z} \subseteq \boldsymbol{V}$, we have: if $I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y})$ then $\langle \boldsymbol{X} \mid \boldsymbol{Z} \mid \boldsymbol{Y} \rangle^d_G$;*
- *the graph G is called a* directed independence map*, or* I-map *for short, for I, if for all sets of variables $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z} \subseteq \boldsymbol{V}$, we have: if $\langle \boldsymbol{X} \mid \boldsymbol{Z} \mid \boldsymbol{Y} \rangle^d_G$ then $I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y})$;*
- *the graph G is called a* directed perfect map*, or* P-map *for short, for I, if G is both a directed D-map and a directed I-map for I.*

An efficient algorithm exists that identifies d-separation properties represented in a directed graph in time $O(|\boldsymbol{A}_G|)$ [Geiger *et al.*, 1990]. A more intuitively appealing method for identifying d-separation properties is known by the name of *'Bayes-Ball'* (see [Shachter, 1998], Algorithm 2); the associated algorithm (Algorithm 3) identifies different sets of vertices—that allow for establishing more than just d-separation properties from a Bayesian network—in $O(|\boldsymbol{A}_G| + |\boldsymbol{V}_G|)$ time.

The previous definition suggests that the 'language' of directed graphs is also not necessarily apt to faithfully represent an independence relation: in addition to P-maps, D-maps and I-maps exist. The different types of directed map have the same meaning as the different types of undirected map we discerned in Section 3.2.1. In a directed D-map for an independence relation, any pair of neighbouring vertices again represents a pair of dependent variables; however, not every pair of dependent variables needs be represented as a pair of neighbouring vertices.

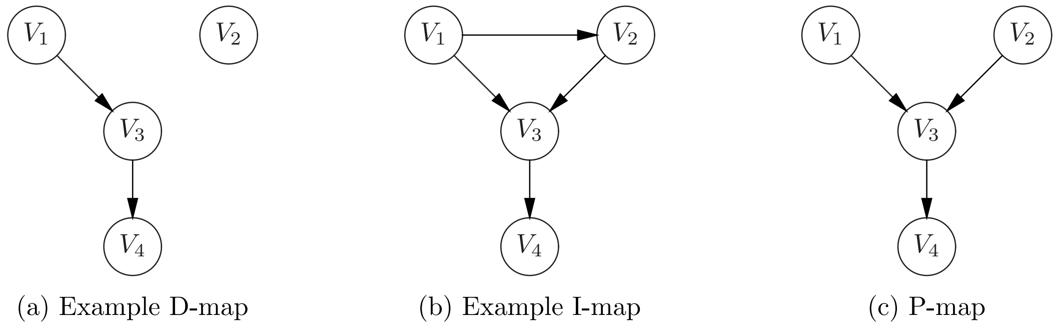


Figure 3.3: Directed graph encodings for the independence relation from Example 3.2.12.

Example 3.2.12 Let $\boldsymbol{V} = \{V_1, V_2, V_3, V_3\}$ be a set of random variables and consider the independence relation I on $\boldsymbol{V}$ that is defined by the independence statements $I(\{V_1\}, \varnothing, \{V_2\})$ and $I(\{V_1, V_2\}, \{V_3\}, \{V_4\})$. The digraph G shown in Figure 3.3a is an example of a directed D-map for I, since for each independence statement $I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y})$, $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z} \subseteq \boldsymbol{V}$, from the relation I, there exists a matching d-separation statement $\langle \boldsymbol{X} \mid \boldsymbol{Z} \mid \boldsymbol{Y} \rangle_G^d$. For example, for the independence statement $I(\{V_1\}, \varnothing, \{V_2\})$, we find that $\langle \{V_1\} \mid \varnothing \mid \{V_2\} \rangle_G^d$ as there does not exist any chain between the vertices V_1 and V_2 in G. $\square$

In a directed I-map, any pair of non-neighbouring vertices once more represents a pair of variables that are (conditionally) independent; however, not every pair of (conditionally) independent variables is represented as a pair of non-neighbouring vertices.

Example 3.2.13 Consider once more the independence relation I from Example 3.2.12. The digraph G shown in Figure 3.3b is an example of a directed I-map for the relation I, since for each d-separation statement $\langle \boldsymbol{X} \mid \boldsymbol{Z} \mid \boldsymbol{Y} \rangle_G^d$, $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z} \subseteq \boldsymbol{V}$, read from G, there exists a matching independence statement $I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y})$ in I. For example, for the d-separation statement $\langle \{V_1\} \mid \{V_3\} \mid \{V_4\} \rangle_G^d$ read from the digraph, we find the statement $I(\{V_1\}, \{V_3\}, \{V_4\})$ in the independence relation. $\square$

A directed P-map for an independence relation once more faithfully captures all independences and dependences from the relation.

Example 3.2.14 Consider once more the independence relation I from Example 3.2.12. The digraph G shown in Figure 3.3c is a directed P-map for G. Note that G is a directed D-map as well as a directed I-map for I. $\square$

From the above observations, we once more conclude that, if a digraph is a directed I-map for an independence relation, we can read independence statements from it; from a directed D-map we can read dependence statements, and from a directed P-map we can read both independence and dependence statements.

Just as we have done for undirected graphs, we investigate the expressive power of the formalism of directed graphs for representing independence relations. It can easily be shown that every independence relation has a directed D-map and a directed I-map.

Lemma 3.2.15 *For every independence relation, there exist a directed D-map and a directed I-map.*

Proof. The proof is analogous to the proof of Lemma 3.2.6. For the directed D-map we consider the arcless graph $G = (\boldsymbol{V}, \varnothing)$. For the directed I-map we consider a complete oriented graph $G = (\boldsymbol{V}, \boldsymbol{A}_G)$ where for each $V_i, V_j \in \boldsymbol{V}$, $i \neq j$, $\boldsymbol{A}_G$ includes either (V_i, V_j) or (V_j, V_i) and G is acyclic. That is, every pair of vertices is connected by a single arc; since arc directions can be arbitrarily chosen as long as the resulting graph is acyclic, multiple complete oriented graphs over the same set of vertices exist. □

Unfortunately, a similar property does not hold for directed P-maps: not every independence relation has a directed P-map.

Example 3.2.16 Consider an independence relation I on a set of variables $\boldsymbol{V}$ such that $I(\{V_1\}, \{V_2, V_3\}, \{V_4\})$ and $I(\{V_2\}, \{V_1, V_4\}, \{V_3\})$ for some variables V_1, V_2, V_3, $V_4 \in \boldsymbol{V}$. No acyclic digraph can model these independence statements simultaneously. □

Although not every independence relation has a directed P-map, there are independence relations that can indeed be represented faithfully. An independence relation for which a directed P-map exists is termed *directed graph-isomorphic.*

Definition 3.2.17 *An independence relation I is said to be* directed graph-isomorphic *if there exists an acyclic digraph G such that G is a directed P-map for I.*

We would like to note that, if an independence relation is directed graph-isomorphic, it may allow several different directed P-maps, that is, a directed P-map does not need to be unique. To conclude, we would like to mention that a necessary condition has been identified for an independence relation to be directed graph-isomorphic [Pearl, 1988]. Here, we will not elaborate any further on this observation.

3.2.3 Choosing a Graphical Representation

In the previous sections we have investigated two different graphical formalisms for representing independence relations: the formalism of undirected graphs and the formalism of directed graphs. Both formalisms allow for a *space-efficient* representation of an independence relation: all graphical representations considered are polynomial in terms of the number of variables of the relation at hand. In addition, the two formalisms provide for an *explicit* representation of independence that allows for efficiently verifying independence statements without requiring numerical computations. These properties render graphical formalisms highly suitable for representing independence relations in knowledge-based systems. Unfortunately, while the graphical formalisms allow for capturing some independence relations to accuracy, they do not allow for a faithful representation (P-map) for every such relation. The correspondence between independence relations on the one hand and graphical representations on the other hand is shown schematically in Figure 3.4. This

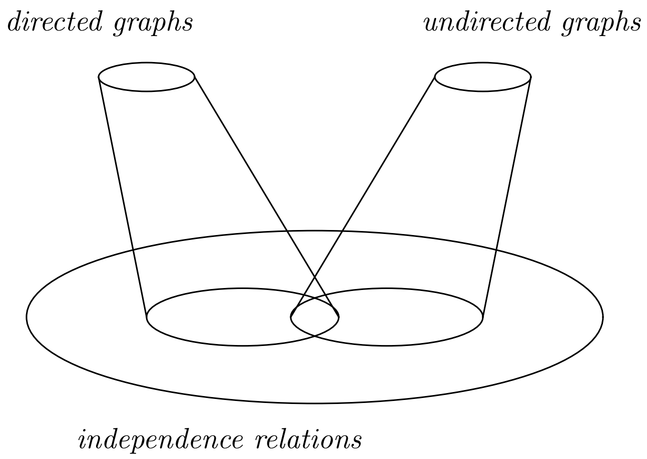


Figure 3.4: Independence Relations and Graphical Representations.

figure depicts that there are independence relations that are undirected graph-isomorphic yet not directed graph-isomorphic, and vice versa; also, there are independence relations that are both undirected and directed graph-isomorphic and independence relations that are not graph-isomorphic at all. This property negatively affects the suitability of graphical formalisms for representing independence relations. The efficiency of representation, however, generally is considered to outweigh the lack of expressive power of these formalisms. We would like to note that it is possible to mix directed and undirected graphs; such mixed graphs are called *chain graphs* [Studený, 1998] and are capable of representing a broader class of independence relations. Chain graphs are, however, hardly used for practical applications as a result of their more complex semantics. In the Bayesian network framework, therefore, a graphical formalism is used for representing independences. As experience learns that the independence relations that are typically encountered in practical problem domains often are best represented by a directed graph rather than by an undirected graph, the formalism of directed graphs is employed in the framework. There does exist an undirected counterpart of the Bayesian network in the family of Probabilistic Graphical Models: the *Markov network*.

In real-life problem domains, independence relations may be encountered that are not graph-isomorphic. For such an independence relation, it is not possible to faithfully represent all independences as well as all dependences. The relation therefore has to be represented in either a directed I-map or a directed D-map. Now observe that since we are interested in exploiting *in*dependences for simplifying computations, we have to make sure that independences that can be read from the graphical representation of an independence relation actually do hold in the relation at hand; otherwise, we would assume independences where there are none and thereby introduce errors in inference. An independence relation that is not directed graph-isomorphic therefore is best represented by a directed *I-map*. Furthermore, acknowledging that some independences will escape representation, we settle for a representation in which *as many* of the independences of the relation as possible are modelled, that is, we insist that the number of unrepresented independences is kept at a minimum. A directed I-map that does not contain any superfluous arcs is called *minimal*.

Definition 3.2.18 *Let $\boldsymbol{V}$ be a set of random variables and let I be an independence relation on $\boldsymbol{V}$. Furthermore, let $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ be an acyclic digraph with $\boldsymbol{V}_G = \boldsymbol{V}$. Then, the digraph G is called a* minimal *directed I-map for I if G is a directed I-map for I and no proper subgraph of G is a directed I-map for I.*

We would like to note that an independence relation may allow several minimal I-maps,

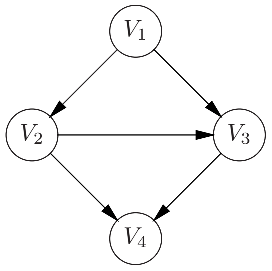


Figure 3.5: An Example Minimal Directed I-map.

that is, a minimal directed I-map need not be unique. Different I-maps that represent the same independence relation are called *Markov equivalent.*

Example 3.2.19 Let $\boldsymbol{V} = \{V_1, V_2, V_3, V_4\}$ be a set of random variables. We consider once more the independence relation I on $\boldsymbol{V}$ defined by the statements $I(\{V_1\}, \{V_2, V_3\}, \{V_4\})$ and $I(\{V_2\}, \{V_1, V_4\}, \{V_3\})$. In Example 3.2.16 we concluded that the relation I is not directed graph-isomorphic. The digraph G shown in Figure 3.5 now is an example of a minimal directed I-map for I. Note that, whereas the independence statement $I(\{V_1\}, \{V_2, V_3\}, \{V_4\})$ is portrayed by G, the statement $I(\{V_2\}, \{V_1, V_4\}, \{V_3\})$ has escaped representation. □

Exercises

*Exercise 3.1

Let $\boldsymbol{V}$ be a set of random variables. Let Pr *be a joint probability distribution on $\boldsymbol{V}$ and let $I_{\Pr}$ be its independence relation. Show that $I_{\Pr}$ satisfies the properties*

a. $I_{\Pr}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y}) \rightarrow I_{\Pr}(\boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{X})$;

b. $I_{\Pr}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y} \cup \boldsymbol{W}) \rightarrow I_{\Pr}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y}) \wedge I_{\Pr}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{W})$;

c. $I_{\Pr}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y} \cup \boldsymbol{W}) \rightarrow I_{\Pr}(\boldsymbol{X}, \boldsymbol{Z} \cup \boldsymbol{W}, \boldsymbol{Y})$;

d. $I_{\Pr}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y}) \wedge I_{\Pr}(\boldsymbol{X}, \boldsymbol{Z} \cup \boldsymbol{Y}, \boldsymbol{W}) \rightarrow I_{\Pr}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y} \cup \boldsymbol{W})$;

for all sets of variables $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{W} \subseteq \boldsymbol{V}$.

* Exercise 3.2

Let $\boldsymbol{V}$ be a set of random variables and let I be a semi-graphoid independence relation on $\boldsymbol{V}$. Show that

$$I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y} \cup \boldsymbol{W}) \wedge I(\boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{W}) \rightarrow I(\boldsymbol{X} \cup \boldsymbol{W}, \boldsymbol{Z}, \boldsymbol{Y})$$

for all sets of variables $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{W} \subseteq \boldsymbol{V}$.

* Exercise 3.3

Let $\boldsymbol{V}$ be a set of random variables and let I be a semi-graphoid independence relation on $\boldsymbol{V}$. Show that

$$I(\boldsymbol{X}, \boldsymbol{Y} \cup \boldsymbol{Z}, \boldsymbol{U} \cup \boldsymbol{W}) \wedge I(\boldsymbol{Y}, \boldsymbol{Z} \cup \boldsymbol{U}, \boldsymbol{X}) \rightarrow I(\boldsymbol{X}, \boldsymbol{Z} \cup \boldsymbol{U}, \boldsymbol{Y} \cup \boldsymbol{W})$$

for all sets of variables $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{W} \subseteq \boldsymbol{V}$.

Exercise 3.4

Let $\boldsymbol{V} = \{V_1, V_2, V_3, V_4\}$ be a set of random variables. Furthermore, let I be the following (semi-graphoid) independence relation on $\boldsymbol{V}$:

$$\begin{array}{lll}
I(\{V_1\}, \varnothing, \{V_4\}) & I(\{V_1\}, \varnothing, \{V_2\}) & I(\{V_1, V_3\}, \{V_2\}, \{V_4\}) \\
I(\{V_2\}, \varnothing, \{V_4\}) & I(\{V_2\}, \varnothing, \{V_1\}) & I(\{V_4\}, \{V_2\}, \{V_1\}) \\
I(\{V_3\}, \varnothing, \{V_4\}) & I(\{V_1, V_4\}, \varnothing, \{V_2\}) & I(\{V_4\}, \{V_2\}, \{V_3\}) \\
I(\{V_4\}, \varnothing, \{V_1\}) & I(\{V_2\}, \varnothing, \{V_1, V_4\}) & I(\{V_4\}, \{V_2\}, \{V_1, V_3\}) \\
I(\{V_4\}, \varnothing, \{V_2\}) & I(\{V_2, V_4\}, \varnothing, \{V_1\}) & I(\{V_1\}, \{V_3\}, \{V_4\}) \\
I(\{V_4\}, \varnothing, \{V_3\}) & I(\{V_1\}, \varnothing, \{V_2, V_4\}) & I(\{V_2\}, \{V_3\}, \{V_4\}) \\
I(\{V_1, V_2\}, \varnothing, \{V_4\}) & I(\{V_2\}, \{V_1\}, \{V_4\}) & I(\{V_1, V_2\}, \{V_3\}, \{V_4\}) \\
I(\{V_1, V_3\}, \varnothing, \{V_4\}) & I(\{V_3\}, \{V_1\}, \{V_4\}) & I(\{V_1\}, \{V_4\}, \{V_2\}) \\
I(\{V_2, V_3\}, \varnothing, \{V_4\}) & I(\{V_2, V_3\}, \{V_1\}, \{V_4\}) & I(\{V_2\}, \{V_4\}, \{V_1\}) \\
I(\{V_4\}, \varnothing, \{V_1, V_2\}) & I(\{V_4\}, \{V_1\}, \{V_2\}) & I(\{V_3\}, \{V_1, V_2\}, \{V_4\}) \\
I(\{V_4\}, \varnothing, \{V_1, V_3\}) & I(\{V_4\}, \{V_1\}, \{V_3\}) & I(\{V_4\}, \{V_1, V_2\}, \{V_3\}) \\
I(\{V_4\}, \varnothing, \{V_2, V_3\}) & I(\{V_4\}, \{V_1\}, \{V_2, V_3\}) & I(\{V_2\}, \{V_1, V_3\}, \{V_4\}) \\
I(\{V_1, V_2, V_3\}, \varnothing, \{V_4\}) & I(\{V_1\}, \{V_2\}, \{V_4\}) & I(\{V_4\}, \{V_1, V_3\}, \{V_2\}) \\
I(\{V_4\}, \varnothing, \{V_1, V_2, V_3\}) & I(\{V_3\}, \{V_2\}, \{V_4\}) & I(\{V_1\}, \{V_2, V_3\}, \{V_4\}) \\
 & & I(\{V_4\}, \{V_2, V_3\}, \{V_1\})
\end{array}$$

Show that each statement $I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y})$, $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z} \subseteq \boldsymbol{V}$, of the independence relation I can be derived from the statements $I(\{V_1, V_2, V_3\}, \varnothing, \{V_4\})$ and $I(\{V_1\}, \varnothing, \{V_2\})$ by the independence axioms from Definition 3.1.3.

* Exercise 3.5

Let $\boldsymbol{V} = \{V_1, V_2, V_3, V_4\}$ be a set of random variables. Let I be the independence relation on $\boldsymbol{V}$ that is defined by the statements $I(\{V_1\}, \{V_2, V_3\}, \{V_4\})$ and $I(\{V_2\}, \{V_1, V_4\}, \{V_3\})$.

a. Give all undirected D-maps for the independence relation I;

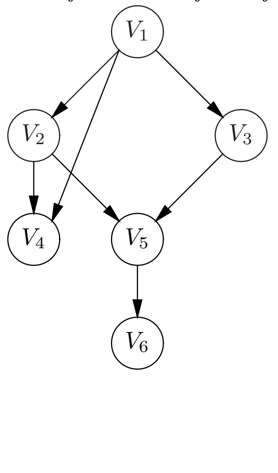
b. Give all undirected I-maps for the independence relation I.

* Exercise 3.6

Recall that ν_G is the neighbour set (see Definition 2.1.2). Show that for any undirected graph $G = (\boldsymbol{V}_G, \boldsymbol{E}_G)$ the following property holds: for any vertex $V_i \in \boldsymbol{V}_G$ and any vertex $V_j \in \boldsymbol{V}_G \setminus (\{V_i\} \cup \nu_G(V_i))$, we have that $\langle \{V_i\} \,|\, \nu_G(V_i) \,|\, \{V_j\} \rangle_G$.

* Exercise 3.7

Examine for each of the following statements whether or not it holds in graph G:



a. $\langle \{V_1\} \,|\, \{V_2, V_3\} \,|\, \{V_6\} \rangle^d_G$;

b. $\langle \{V_2\} \,|\, \varnothing \,|\, \{V_3\} \rangle^d_G$;

c. $\langle \{V_2\} \,|\, \{V_1\} \,|\, \{V_3\} \rangle^d_G$;

d. $\langle \{V_4\} \,|\, \{V_1\} \,|\, \{V_3\} \rangle^d_G$;

e. $\langle \{V_2\} \,|\, \{V_3, V_4\} \,|\, \{V_6\} \rangle^d_G$;

f. $\langle \{V_3\} \,|\, \varnothing \,|\, \{V_1\} \rangle^d_G$.

* Exercise 3.8

Let $\boldsymbol{V} = \{V_1, V_2, V_3, V_4\}$ be a set of random variables. Let I be the independence relation on $\boldsymbol{V}$ that is defined by the statements $I(\{V_1\}, \varnothing, \{V_2\})$ and $I(\{V_1, V_2\}, \{V_3\}, \{V_4\})$.

a. *Give some directed D-maps for the independence relation I;*

b. *Give some directed I-maps for the independence relation I.*

* Exercise 3.9

Show that for every independence relation there exists a directed D-map and a directed I-map (Lemma 3.2.6).

* Exercise 3.10

Given an example of an independence relation that has more than one directed P-map.

* Exercise 3.11

Let $\boldsymbol{V} = \{V_1, V_2, V_3, V_4\}$ be a set of random variables. Let I be the independence relation on $\boldsymbol{V}$ that is defined by the statements $I(\{V_1\}, \{V_2, V_3\}, \{V_4\})$ and $I(\{V_2\}, \{V_1, V_4\}, \{V_3\})$. Give some minimal directed I-maps for the relation I.

* Exercise 3.12

Show that for any acyclic directed graph $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ the following property holds: for any vertex $V_i \in \boldsymbol{V}_G$ and any vertex $V_j \in \boldsymbol{V}_G \setminus (\boldsymbol{\sigma}^*_G(V_i) \cup \boldsymbol{\rho}_G(V_i))$, we have that $\langle \{V_i\} \,|\, \boldsymbol{\rho}_G(V_i) \,|\, \{V_j\} \rangle^d_G$.

This property is known as the local Markov property *of a DAG.*

* Exercise 3.13

Let $\boldsymbol{V}$ be a set of random variables. Let I be an independence relation on $\boldsymbol{V}$ and let G be a directed I-map for I. Now, let H be the underlying graph of G. Is H an undirected I-map for I ?

* Exercise 3.14

a. *Give an example of an independence relation that has both an undirected P-map and a directed P-map.*

b. *Give an example of an independence relation that has an undirected P-map but no directed P-map.*

c. *Give an example of an independence relation that has a directed P-map but no undirected P-map.*

d. *Give an example of an independence relation that has no undirected P-map nor a directed P-map.*

Chapter 4

The Bayesian Network Framework

The Bayesian network framework is characterised by a powerful and intuitively appealing formalism for representing a joint probability distribution on a set of random variables, for use in a decision-support system. Associated with this formalism are algorithms for efficiently computing probabilities of interest and for processing evidence. These algorithms— referred to as algorithms for (probabilistic) *inference*— constitute the basic building blocks for reasoning within the modelled domain. In this chapter we introduce the Bayesian network formalism and detail a well-known message-passing algorithm for *exact* inference. The ideas underlying this algorithm are also found in more popular algorithms, both for exact and approximate inference [Korb & Nicholson, 2010]. Approximate inference is beyond the scope of this syllabus.

4.1 The Bayesian Network Formalism

The Bayesian network formalism provides a concise representation of a joint probability distribution on a set of random variables $\boldsymbol{V}$. The conciseness of representation is arrived at by explicit separation of knowledge of the independences holding in a distribution and the numerical quantities involved. To this end, a Bayesian network comprises two parts: a *qualitative* part and a *quantitative* part. The qualitative part of a Bayesian network is a graphical representation of the independences holding among the variables in the probability distribution that is being represented; more in specific, the qualitative part of a Bayesian network is a (minimal) directed I-map for the independence relation of the distribution. Associated with the qualitative part of a Bayesian network is a set of functions representing numerical quantities from the distribution; with each vertex in the digraph is associated a *assessment function* which basically is a set of (conditional) probabilities describing the influence of the values of the vertex' predecessors on the probabilities of the values of this vertex itself. These assessment functions with each other constitute the quantitative part of the Bayesian network.

We define the concept of a Bayesian network more formally.

Definition 4.1.1 *A* Bayesian network *is a tuple* $\mathcal{B} = (G, \Gamma)$ *where*

- $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ *is an acyclic digraph with vertices* $\boldsymbol{V}_G$ *representing a set of random variables* $\{V_1, \ldots, V_n\}$*,* $n \geq 1$*, and arcs* $\boldsymbol{A}_G$*;*
- $\Gamma = \{\gamma_{V_i} \mid V_i \in \boldsymbol{V}_G\}$ *is a set of real-valued non-negative functions*

  $$\gamma_{V_i} \colon \{c_{V_i}\} \times \{c_{\boldsymbol{\rho}_G(V_i)}\} \to [0, 1]$$

  called assessment functions*, such that for each configuration* $c_{\boldsymbol{\rho}_G(V_i)}$ *of the set* $\boldsymbol{\rho}_G(V_i)$ *of (immediate) predecessors of vertex* V_i *in* G*, we have that* $\gamma_{V_i}(\neg v_i \mid c_{\boldsymbol{\rho}_G(V_i)}) = 1 - \gamma_{V_i}(v_i \mid c_{\boldsymbol{\rho}_G(V_i)})$*,* $i = 1, \ldots, n$*.*

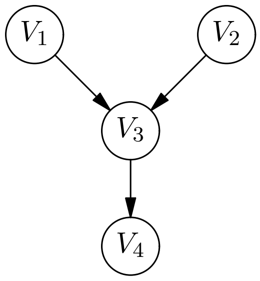


Figure 4.1: The Digraph of an Example Bayesian Network.

Note from the previous definition that there exists a one-to-one correspondence between vertices $V_i \in \boldsymbol{V}_G$ and random variables $V_i \in \boldsymbol{V}$.

Despite the fact that the numbers specified by the assessment functions are (conditional) probabilities, we prefer to use the γ notation over the use of Pr to explicitly distinguish between the probabilities that are part of the Bayesian network specification and the probabilities that can be computed from the network specification using inference algorithms.

The assessment functions for a single vertex are often represented in a table, typically referred to as *conditional probability table* or CPT. The specified numbers are therefore often called *CPT-parameters* or *CPT-entries*; we will use the more general phrase of *model-parameter* or *network-parameter* to refer to these probabilities.

A lot of literature uses the phrase *parameter*, and θ/Θ rather than γ/Γ to refer to the probabilities that are part of the network specification, especially in the context of learning these from data. However, the literature also uses "parameter" and θ to denote other unknown or varying quantities. Moreover, we will use "parameter" later on extensively to refer to messages in the message-passing algorithm.

Example 4.1.2 We consider the digraph G shown in Figure 4.1. With the digraph G, we associate a set $\Gamma = \{\gamma_{V_i} \mid i = 1, \ldots, 4\}$ of assessment functions γ_{V_i}. For example, for the vertices V_1 and V_2, the assessment functions γ_{V_1} and γ_{V_2} may be defined as

$$\begin{array}{llcll} \gamma_{V_1}(v_1) & = 0.25 & \text{and} & \gamma_{V_2}(v_2) & = 0.5 \\ \gamma_{V_1}(\neg v_1) & = 0.75 & & \gamma_{V_2}(\neg v_2) & = 0.5 \end{array}$$

For vertex V_4, the assessment function γ_{V_4} is defined as

$$\begin{array}{llll} \gamma_{V_4}(v_4 \mid v_3) & = 0.8 & \gamma_{V_4}(\neg v_4 \mid v_3) & = 0.2 \\ \gamma_{V_4}(v_4 \mid \neg v_3) & = 0 & \gamma_{V_4}(\neg v_4 \mid \neg v_3) & = 1.0 \end{array}$$

For vertex V_3, the assessment function γ_{V_3} is defined by the values

$$\begin{array}{llcll} \gamma_{V_3}(v_3 \mid v_1 \wedge v_2) & = 0.75 & & \gamma_{V_3}(\neg v_3 \mid v_1 \wedge v_2) & = 0.25 \\ \gamma_{V_3}(v_3 \mid \neg v_1 \wedge v_2) & = 0.4 & \text{and their } \textit{complements} & \gamma_{V_3}(\neg v_3 \mid \neg v_1 \wedge v_2) & = 0.6 \\ \gamma_{V_3}(v_3 \mid v_1 \wedge \neg v_2) & = 0.25 & & \gamma_{V_3}(\neg v_3 \mid v_1 \wedge \neg v_2) & = 0.75 \\ \gamma_{V_3}(v_3 \mid \neg v_1 \wedge \neg v_2) & = 0.2 & & \gamma_{V_3}(\neg v_3 \mid \neg v_1 \wedge \neg v_2) & = 0.8 \end{array}$$

The pair $\mathcal{B} = (G, \Gamma)$ is a Bayesian network. $\square$

The assessment functions of a Bayesian network with each other provide all information necessary for uniquely defining a joint probability distribution on the variables discerned that respects the independence relation portrayed by the qualitative part of the network; henceforth, we will call this distribution the joint probability distribution *defined* by the network (note again the use of the template notation).

Proposition 4.1.3 *Let $\mathcal{B} = (G, \Gamma)$ be a Bayesian network. Then,*

$$\Pr(\boldsymbol{V}_G) = \prod_{V_i \in \boldsymbol{V}_G} \gamma_{V_i}(V_i \mid \boldsymbol{\rho}_G(V_i))$$

defines a joint probability distribution Pr *on $\boldsymbol{V}_G$ such that G is a directed I-map for the independence relation $I_{\Pr}$ of* Pr.

Proof. Since the digraph G of the Bayesian network B is acyclic, it allows a total ordering of its vertices such that any successor of a vertex in the digraph follows it in the ordering; such an ordering is termed a *topological order* of the digraph's vertices. Note that any topological order of the vertices of G constitutes an ordering of the corresponding random variables. We take ι_G to be such a topological order; for ease of exposition, we assume that $\iota_G(V_i) = i$ for all $V_i \in \boldsymbol{V}_G$. We now consider an arbitrary joint probability distribution P on $\boldsymbol{V}_G$ such that G is a directed I-map for the independence relation of P. To the expression $P(\boldsymbol{V}_G)$ describing the distribution P, we apply the chain rule from probability theory, such that every variable V_i from $\boldsymbol{V}_G$ is conditioned on the variables $V_1, \ldots, V_{i-1}$, preceding it in the ordering ι_G, that is,

$$P(\boldsymbol{V}_G) = \prod_{V_i \in \boldsymbol{V}_G} P(V_i \mid V_1 \wedge \cdots \wedge V_{i-1})$$

From the digraph G, we read, by means of the d-separation criterion, that $\langle \{V_i\} \mid \boldsymbol{\rho}_G(V_i) \mid \{V_1, \ldots, V_{i-1}\} \setminus \boldsymbol{\rho}_G(V_i)\rangle_G^d$ for every vertex $V_i \in \boldsymbol{V}_G$. Since G is a directed I-map for the distribution P, we have for every V_i that $I_P(\{V_i\}, \boldsymbol{\rho}_G(V_i), \{V_1, \ldots, V_{i-1}\} \setminus \boldsymbol{\rho}_G(V_i))$. From this observation, we have that

$$P(V_i \mid V_1 \wedge \cdots \wedge V_{i-1}) = P(V_i \mid \boldsymbol{\rho}_G(V_i))$$

for every vertex $V_i \in \boldsymbol{V}_G$. By exploiting this property, we find that

$$P(\boldsymbol{V}_G) = \prod_{V_i \in \boldsymbol{V}_G} P(V_i \mid \boldsymbol{\rho}_G(V_i))$$

Since P has been chosen arbitrarily, we conclude that any joint probability distribution such that G is a directed I-map for its independence relation, satisfies this property. By reversing the argument, we find that there exists a (unique) joint probability distribution Pr on $\boldsymbol{V}_G$ such that G is a directed I-map for the independence relation of Pr and $\Pr(V_i \mid \boldsymbol{\rho}_G(V_i)) = \gamma_{V_i}(V_i \mid \boldsymbol{\rho}_G(V_i))$ for each variable $V_i \in \boldsymbol{V}_G$. □

We illustrate the basic idea of the previous proposition by means of an example.

Example 4.1.4 We consider once more the Bayesian network $\mathcal{B} = (G, \Gamma)$ from Example 4.1.2. From the property stated in Proposition 4.1.3, we have that from the expression

$$\Pr(V_1 \wedge V_2 \wedge V_3 \wedge V_4) \ = \gamma_{V_1}(V_1) \cdot \gamma_{V_2}(V_2) \cdot \gamma_{V_3}(V_3 \mid V_1 \wedge V_2) \cdot \gamma_{V_4}(V_4 \mid V_3)$$

any probability of interest can be computed. For example, we have that

$$\begin{aligned} \Pr(v_1 \wedge \neg v_2 \wedge v_3 \wedge v_4) \ &= \gamma_{V_1}(v_1) \cdot \gamma_{V_2}(\neg v_2) \cdot \gamma_{V_3}(v_3 \mid v_1 \wedge \neg v_2) \cdot \gamma_{V_4}(v_4 \mid v_3) = \\ &= 0.25 \cdot 0.5 \cdot 0.25 \cdot 0.8 = 0.025 \end{aligned}$$

and

$$\begin{aligned}\Pr(\neg v_1 \wedge \neg v_2 \wedge v_3 \wedge v_4) &= \gamma_{V_1}(\neg v_1) \cdot \gamma_{V_2}(\neg v_2) \cdot \gamma_{V_3}(v_3 \mid \neg v_1 \wedge \neg v_2) \cdot \gamma_{V_4}(v_4 \mid v_3) = \\ &= 0.06\end{aligned}$$

By marginalisation we can now compute the probability $\Pr(\neg v_2 \wedge v_3 \wedge v_4)$:

$$\begin{aligned}\Pr(\neg v_2 \wedge v_3 \wedge v_4) &= \Pr(v_1 \wedge \neg v_2 \wedge v_3 \wedge v_4) + \Pr(\neg v_1 \wedge \neg v_2 \wedge v_3 \wedge v_4) = \\ &= 0.085\end{aligned}$$

□

From Proposition 4.1.3 it is readily seen that the function values of a Bayesian network's assessment functions can be interpreted as conditional probabilities.

Corollary 4.1.5 *Let $\mathcal{B} = (G, \Gamma)$ be a Bayesian network and let* Pr *be the joint probability distribution defined by $\mathcal{B}$. Then, for each vertex $V_i \in \boldsymbol{V}_G$, we have that*

$$\Pr(V_i \mid \boldsymbol{\rho}_G(V_i)) = \gamma_{V_i}(V_i \mid \boldsymbol{\rho}_G(V_i))$$

4.2 Exact Probabilistic Inference

A Bayesian network generally is used for *probabilistic inference*, that is, for making probabilistic statements concerning the variables that are represented in the network. For this purpose, the property stated in Proposition 4.1.3 can be exploited: from the proposition we have that, since the digraph of a Bayesian network and its associated assessment functions with each other uniquely define a joint probability distribution, any (prior or posterior) probability of interest can be computed from the network. Explicitly generating the represented probability distribution as indicated by the proposition and then using the basic rules of marginalisation and conditioning, however, is computationally infeasible. In addition, such a straightforward approach would not exploit the independences that are represented by the qualitative part of the network at hand. More efficient algorithms for exact probabilistic inference have been designed that do exploit the represented independences. The best known of these are the *Belief Propagation* algorithm by J. Pearl [Pearl, 1988] and the algorithm of S.L. Lauritzen and D.J. Spiegelhalter [Lauritzen & Spiegelhalter, 1988]. Although the latter is typically used in practice for computational reasons, in the sequel we will focus on Pearl's algorithm. This choice is mostly made for didactic reasons. In addition, the underlying message-passing algorithm can be found in many other inference algorithms for probabilistic models, such as the Sum-Product and Viterbi algorithms.

Idea underlying belief propagation The basic idea of Pearl's algorithm for probabilistic inference is best explained from an object-centered point of view. The digraph of a Bayesian network is looked upon as a *computational architecture*: the vertices of the digraph are *autonomous objects* and its arcs are *bi-directional communication channels*. Each vertex has a local processor that is capable of performing simple probabilistic computations and a local memory in which its associated assessment function is stored. Through the communication channels the vertices send each other *parameters*[1] providing information about the joint probability distribution that is represented by the network — that is, information determined from the network's assessment functions and the independences portrayed in the digraph — and about the evidence that has been entered and processed

[1]Note that these parameters are *not* the same as the model-parameters γ specified by the network's assessment functions!

so far. Each vertex is now able to compute the probabilities of its values from its own assessment function and the information it receives from its neighbours.

Initially, a Bayesian network is in an *equilibrium* state: recomputation of the various compound parameters and message parameters that the vertices send one another, does not result in a change in any of them. Now, when a piece of evidence is entered into the network for some vertex, this equilibrium is perturbed, since the probability distribution over the network's variables under consideration has now changed: the distribution should be conditioned on the (additional) evidence obtained. Once the value of a variable is known with certainty, the probability distribution for the variable degenerates and can no longer change; entering evidence for a variable into a Bayesian network thus basically amounts to adding its corresponding vertex to the (initially empty) blocking-set W for the digraph, thereby changing the independences that have to be taken into account by the inference algorithm. Therefore, to process the evidence entered for some vertex, the message parameters this vertex sends to its neighbours are updated. After receiving updated message parameters, these neighbours in turn compute new message parameters to send to their neighbours, and so on. The impact of the evidence thus spreads throughout the network by *message-passing* between neighbouring vertices. After a full *network propagation*, when all vertices have sent and received message parameters and have updated their probabilities, a new equilibrium state is reached.

In this section, we detail the computations involved in Pearl's belief propagation algorithm. In doing so, we distinguish between various types of graph. In Section 4.2.1, we focus on directed trees. In Section 4.2.2 we address singly connected digraphs more in general. In Section 4.2.3 we discuss probabilistic inference with Bayesian networks comprising a multiply connected digraph.

4.2.1 Directed Trees

In discussing Pearl's algorithm, we begin by focusing on probabilistic inference with a Bayesian network comprising a *directed tree* for its qualitative part, that is, in the network's digraph a vertex may have several successors but at most one predecessor. Before detailing the computations involved in inference with such a network, we introduce some more terminology and notational convention that we will use in the sequel.

Let G be the digraph of a Bayesian network, and let $\boldsymbol{V}$ be (a subset of) the set of variables represented by its vertices. Now $V_i \in \boldsymbol{V}$ is called *instantiated* if its value is known with certainty; otherwise, it is called *uninstantiated*. Now, let $\boldsymbol{X} \subseteq \boldsymbol{V}$ be the set of instantiated variables or vertices from $\boldsymbol{V}$. The configuration $c_{\boldsymbol{X}}$ of $\boldsymbol{X}$ that is known with certainty is a *partial configuration* of $\boldsymbol{V}$ and will be denoted as $\widetilde{c}_{\boldsymbol{V}}$. Note that the notation $\widetilde{c}_{\boldsymbol{V}}$ provides for referring to the subset of instantiated variables or vertices in a Bayesian network without having to specify this subset explicitly.

At any time during probabilistic inference with a Bayesian network, the probabilities of the values of a variable of interest depend upon all evidence entered so far into the network.

Lemma 4.2.1 *Let $\mathcal{B} = (G, \Gamma)$ be a Bayesian network where $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ is a directed tree, and let* Pr *be the joint probability distribution defined by $\mathcal{B}$. Let $V_i \in \boldsymbol{V}_G$ be a vertex in G, and let $\boldsymbol{V_i^-} = \boldsymbol{\sigma}^*_G(V_i)$ and $\boldsymbol{V_i^+} = \boldsymbol{V}_G \setminus \boldsymbol{V_i^-}$. Then,*

$$\Pr(V_i \mid \widetilde{c}_{\boldsymbol{V}_G}) = \alpha \cdot \Pr(\widetilde{c}_{\boldsymbol{V_i^-}} \mid V_i) \cdot \Pr(V_i \mid \widetilde{c}_{\boldsymbol{V_i^+}})$$

where $\widetilde{c}_{\boldsymbol{V}_G} = \widetilde{c}_{\boldsymbol{V_i^-}} \wedge \widetilde{c}_{\boldsymbol{V_i^+}}$ and α is a normalisation constant.

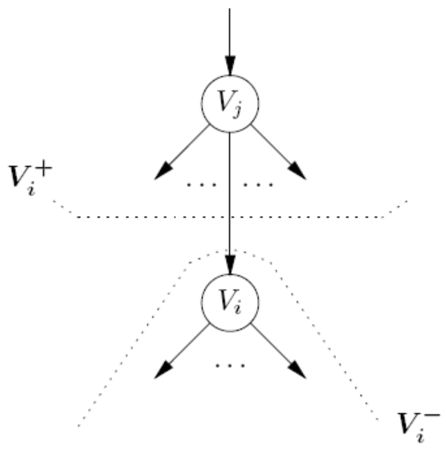


Figure 4.2: A Fragment of a Directed Tree.

Proof. For the probabilities $\Pr(V_i \mid \widetilde{c}_{\boldsymbol{V}_G})$ of the values of vertex V_i, we have

$$\Pr(V_i \mid \widetilde{c}_{\boldsymbol{V}_G}) = \frac{\Pr(\widetilde{c}_{\boldsymbol{V}_G} \mid V_i) \cdot \Pr(V_i)}{\Pr(\widetilde{c}_{\boldsymbol{V}_G})} = \frac{\Pr(\widetilde{c}_{\boldsymbol{V}_i^-} \wedge \widetilde{c}_{\boldsymbol{V}_i^+} \mid V_i) \cdot \Pr(V_i)}{\Pr(\widetilde{c}_{\boldsymbol{V}_i^-} \wedge \widetilde{c}_{\boldsymbol{V}_i^+})}$$

by Bayes' Theorem. Now consider Figure 4.2 showing a fragment of the directed tree G of the network. Using the d-separation criterion, we observe that $\langle \boldsymbol{X} \mid \{V_i\} \mid \boldsymbol{Y} \rangle_G^d$ for all sets of vertices $\boldsymbol{X} \subseteq \boldsymbol{V_i}^-$ and $Y \subseteq \boldsymbol{V_i}^+$. Since G is a directed I-map for the joint probability distribution Pr, we conclude that $I_{\Pr}(\boldsymbol{X}, \{V_i\}, \boldsymbol{Y})$ for all sets $\boldsymbol{X} \subseteq \boldsymbol{V_i}^-$, $\boldsymbol{Y} \subseteq \boldsymbol{V_i}^+$. Exploiting this observation, we find

$$\Pr(V_i \mid \widetilde{c}_{\boldsymbol{V}_G}) = \frac{\Pr(\widetilde{c}_{\boldsymbol{V}_i^-} \mid V_i) \cdot \Pr(\widetilde{c}_{\boldsymbol{V}_i^+} \mid V_i) \cdot \Pr(V_i)}{\Pr(\widetilde{c}_{\boldsymbol{V}_i^-} \wedge \widetilde{c}_{\boldsymbol{V}_i^+})} =$$

$$= \frac{\Pr(\widetilde{c}_{\boldsymbol{V}_i^-} \mid V_i) \cdot \Pr(V_i \mid \widetilde{c}_{\boldsymbol{V}_i^+})}{\Pr(\widetilde{c}_{\boldsymbol{V}_i^-} \mid \widetilde{c}_{\boldsymbol{V}_i^+})}$$

Now observe that the factor $(\Pr(\widetilde{c}_{\boldsymbol{V}_i^-} \mid \widetilde{c}_{\boldsymbol{V}_i^+}))^{-1}$ in the expression depends on the *location* of vertex V_i in the digraph and, if instantiated, on the value of its instantiation, but *not* on the value of V_i under consideration. It therefore is a constant with respect to V_i. In the sequel, this constant will universally be denoted as α. The constant α is generally referred to as a *normalisation constant* because it can be computed from $\Pr(v_i \mid \widetilde{c}_{\boldsymbol{V}_G}) + \Pr(\neg v_i \mid \widetilde{c}_{\boldsymbol{V}_G}) = 1$. The property stated in the lemma now follows by substitution. □

Defining Compound Parameters in Directed Trees

The previous lemma shows that the probabilities of the values of a vertex of interest can be expressed in terms of two factors describing the influence of evidence entered for this vertex' descendants and for all other vertices, separately. The following definition introduces some new terminology for these separate factors.

Definition 4.2.2 *Let $\mathcal{B} = (G, \Gamma)$ be a Bayesian network where $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ is a directed tree, and let* Pr *be the joint probability distribution defined by $\mathcal{B}$. Let $V_i \in \boldsymbol{V}_G$ be a vertex* *in G, and let $\boldsymbol{V}_i^-$ and $\boldsymbol{V}_i^+$ be as before. The* compound causal parameter π_{V_i} *for vertex V_i is the function $\pi_{V_i}\colon V_i \to [0,1]$ defined by*

$$\pi_{V_i}(V_i) = \Pr(V_i \mid \widetilde{c}_{\boldsymbol{V}_i^+})$$

The compound diagnostic parameter λ_{V_i} *for V_i is the function $\lambda_{V_i}\colon V_i \to [0,1]$ defined by*

$$\lambda_{V_i}(V_i) = \Pr(\widetilde{c}_{\boldsymbol{V}_i^-} \mid V_i)$$

Note that the compound diagnostic parameter for a vertex describes the combined influence on this vertex' probabilities of all evidence that has been entered for its descendants; the compound causal parameter for the vertex describes the combined influence of evidence entered for all other vertices in the digraph.

Special cases We take a closer look at the previous definition for some special cases. We begin by addressing *uninstantiated* vertices having either no incoming or no outgoing arcs. A directed tree has one vertex W without any incoming arcs; this vertex is the *root* of the tree. We observe that for W the set $\boldsymbol{W}^+$ is empty. So, $\widetilde{c}_{\boldsymbol{W}^+} = \mathsf{True}$. The compound causal parameter π_W for W therefore equals $\pi_W(W) = \Pr(W)$. The directed tree may further include several vertices having no outgoing arcs; these vertices are the *leaves* of the tree. For a leaf V, we observe that the set $\boldsymbol{V}^-$ consists of V only. From V being uninstantiated, we have that $\widetilde{c}_{\boldsymbol{V}^-} = \mathsf{True}$. The compound diagnostic parameter λ_V for V therefore equals $\lambda_V(V) = 1$, regardless of the value of V. To conclude, we consider *instantiated* vertices. For a vertex V_i for which the evidence $V_i = \mathit{true}$ has been entered, we find $\pi_{V_i}(v_i) = \Pr(v_i \mid \widetilde{c}_{\boldsymbol{V}_i^+})$ and $\pi_{V_i}(\neg v_i) = \Pr(\neg v_i \mid \widetilde{c}_{\boldsymbol{V}_i^+})$, and $\lambda_{V_i}(v_i) = \Pr(\widetilde{c}_{\boldsymbol{V}_i^-} \mid v_i)$ and $\lambda_{V_i}(\neg v_i) = 0$; an analogous observation holds for the case where the evidence $V_i = \mathit{false}$ has been entered.

Data fusion Using the definition of the compound causal and diagnostic parameters for a vertex, the property stated in Lemma 4.2.1 can now be reformulated: for each vertex V_i, we have that

$$\Pr(V_i \mid \widetilde{c}_{\boldsymbol{V}_G}) = \alpha \cdot \pi_{V_i}(V_i) \cdot \lambda_{V_i}(V_i)$$

where α is a normalisation constant. In this form, the lemma is known as the *data fusion lemma* [Pearl, 1988]. Note that the data fusion lemma implies that the compound and diagnostic parameters for a vertex provide it with enough information for computing the probabilities of its values, that is, no further knowledge of the joint probability distribution is needed.

Defining Message Parameters in Directed Trees

The compound diagnostic parameter for a vertex specifies probabilistic information from *all* its descendants combined; an analogous observation applies to the compound causal parameter for the vertex. To be able to exploit the digraph of a Bayesian network as a computational architecture as outlined before, the compound parameters for a vertex have to be computed from separate message parameters originating from its various neighbours. The following definition introduces such message parameters; the Lemmas 4.2.4 and 4.2.5 will show the computation of the compound parameters from these messages.

Definition 4.2.3 *Let $\mathcal{B} = (G, \Gamma)$ be a Bayesian network where $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ is a directed tree, and let* Pr *be the joint probability distribution defined by $\mathcal{B}$. For each vertex $V \in \boldsymbol{V}_G$,* *let* $\boldsymbol{V}^-$ *and* $\boldsymbol{V}^+$ *be as before. Now, let* V_i *be a vertex with a successor* V_k *in* G*. The* causal (message) parameter $\pi^{V_i}_{V_k}$ from V_i to V_k *is the function* $\pi^{V_i}_{V_k} : V_i \to [0,1]$ *defined by*

$$\pi^{V_i}_{V_k}(V_i) = \Pr(V_i \mid \widetilde{c}_{\boldsymbol{V_k^+}})$$

Now, let V_i *be a vertex having the predecessor* V_j *in* G*. The* diagnostic (message) parameter $\lambda^{V_j}_{V_i}$ from V_i to V_j *is the function* $\lambda^{V_j}_{V_i} : \boldsymbol{V_j} \to [0,1]$ *defined by*

$$\lambda^{V_j}_{V_i}(V_j) = \Pr(\widetilde{c}_{\boldsymbol{V_i^-}} \mid V_j)$$

Note that a causal parameter is a message parameter that a vertex sends to a successor to provide this successor with information concerning its non-descendants. A diagnostic parameter is a message parameter that a vertex sends to its predecessor to provide this predecessor with information concerning the vertices located in the subtree rooted at the vertex at hand. The separate causal and diagnostic parameters are the messages the vertices send each other through the communication channels of the computational architecture and, hence, may be looked upon as associated with the arcs of the directed tree of the Bayesian network at hand.

Special cases We take a closer look at the previous definition for some special cases. We observe that for the root of a directed tree no diagnostic parameter is defined because it does not have a predecessor. For the leaves of the tree no causal parameters are defined as these vertices do not have any successors. In addition, we note that for a vertex V_i for which the evidence $V_i = \mathit{true}$ is observed and entered into the network, we find that $\pi^{V_i}_{V_k}(v_i) = 1$ and $\pi^{V_i}_{V_k}(\neg v_i) = 0$ for any successor V_k of V_i; an analogous observation holds for the case where $V_i = \mathit{false}$ is observed.

Computing Compound Parameters in Directed Trees

The following lemma now shows that a vertex can compute its compound causal parameter from the causal parameter it receives from its predecessor and its own assessment function.

Lemma 4.2.4 *Let* $\mathcal{B} = (G, \Gamma)$ *be a Bayesian network where* $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ *is a directed tree. Let* $V_i \in \boldsymbol{V}_G$ *be a vertex with the predecessor* V_j *in* G*. Let* π_{V_i} *be the compound causal parameter for vertex* V_i*, and let* $\pi^{V_j}_{V_i}$ *be the causal parameter from* V_j *to* V_i*. Then,*

$$\pi_{V_i}(V_i) = \sum_{c_{V_j}} \gamma_{V_i}(V_i \mid c_{V_j}) \cdot \pi^{V_j}_{V_i}(c_{V_j})$$

Proof. Let Pr be the joint probability distribution defined by the Bayesian network $\mathcal{B}$. For vertex V_i, let $\boldsymbol{V_i^+}$ be as before. Then, by definition we have that

$$\pi_{V_i}(V_i) = \Pr(V_i \mid \widetilde{c}_{\boldsymbol{V_i^+}})$$

By conditioning on the values of V_i's predecessor V_j, we find

$$\pi_{V_i}(V_i) = \Pr(V_i \mid v_j \wedge \widetilde{c}_{\boldsymbol{V_i^+}}) \cdot \Pr(v_j \mid \widetilde{c}_{\boldsymbol{V_i^+}}) + \Pr(V_i \mid \neg v_j \wedge \widetilde{c}_{\boldsymbol{V_i^+}}) \cdot \Pr(\neg v_j \mid \widetilde{c}_{\boldsymbol{V_i^+}})$$

Now consider once more Figure 4.2 showing a fragment of the directed tree G of the network. Using the d-separation criterion, we observe that $\langle \{V_i\} \mid \{V_j\} \mid \boldsymbol{X} \rangle^d_G$ for all sets

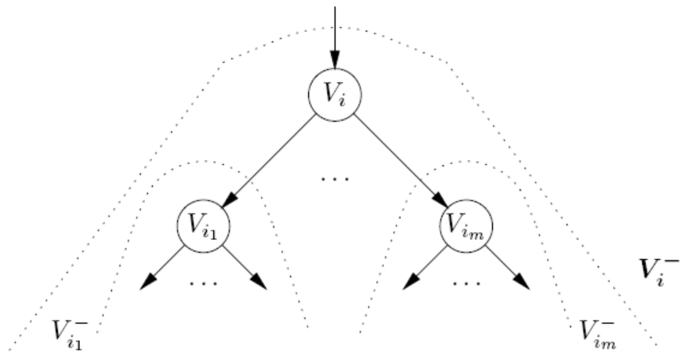


Figure 4.3: Exploiting d-Separation for Computing $\lambda_{V_i}(V_i)$.

of vertices $\boldsymbol{X} \subseteq \boldsymbol{V}_i^+$. Since G is a directed I-map for the distribution Pr, we have that $I_{\Pr}(\{V_i\},\{V_j\},\boldsymbol{X})$ for all sets $\boldsymbol{X} \subseteq \boldsymbol{V}_i^+$. Exploiting this observation, we find

$$\pi_{V_i}(V_i) = \Pr(V_i \mid v_j) \cdot \Pr(v_j \mid \tilde{c}_{\boldsymbol{V}_i^+}) + \Pr(V_i \mid \neg v_j) \cdot \Pr(\neg v_j \mid \tilde{c}_{\boldsymbol{V}_i^+})$$

The probabilities $\Pr(V_i \mid v_j)$ and $\Pr(V_i \mid \neg v_j)$ have been specified as function values of the assessment function γ_{V_i} and therefore are directly available to vertex V_i. In addition, vertex V_i receives the probabilities $\Pr(v_j \mid \tilde{c}_{\boldsymbol{V}_i^+})$ and $\Pr(\neg v_j \mid \tilde{c}_{\boldsymbol{V}_i^+})$ from its predecessor V_j as function values of the causal parameter $\pi_{V_i}^{V_j}$. The property stated in the lemma now follows by substitution. □

A vertex can further compute its compound diagnostic parameter from the separate diagnostic parameters it receives from its successors in the digraph. This property is stated more formally in the following lemma.

Lemma 4.2.5 *Let $\mathcal{B} = (G, \Gamma)$ be a Bayesian network where $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ is a directed tree. Let $V_i \in \boldsymbol{V}_G$ be an uninstantiated vertex with the successors $\boldsymbol{\sigma}_G(V_i) = \{V_{i_1}, \dots, V_{i_m}\}$, $m \geq 1$, in G. Furthermore, let λ_{V_i} be the compound diagnostic parameter for vertex V_i and, for each $V_{i_j} \in \boldsymbol{\sigma}_G(V_i)$, let $\lambda_{V_{i_j}}^{V_i}$ be the diagnostic parameter from V_{i_j} to V_i. Then,*

$$\lambda_{V_i}(V_i) = \prod_{j=1,\dots,m} \lambda_{V_{i_j}}^{V_i}(V_i)$$

Proof. Let Pr be the joint probability distribution defined by the Bayesian network B. For each vertex $V \in \boldsymbol{V}_G$, let $\boldsymbol{V}^-$ be as before. Since V_i is an uninstantiated vertex, we have that $\tilde{c}_{\boldsymbol{V}_i^-} = \tilde{c}_{\boldsymbol{V}_{i_1}^-} \wedge \dots \wedge \tilde{c}_{\boldsymbol{V}_{i_m}^-}$. So,

$$\lambda_{V_i}(V_i) \;=\; \Pr(\tilde{c}_{\boldsymbol{V}_i^-} \mid V_i) \;=\; \Pr(\tilde{c}_{\boldsymbol{V}_{i_1}^-} \wedge \dots \wedge \tilde{c}_{\boldsymbol{V}_{i_m}^-} \mid V_i)$$

Now consider Figure 4.3 showing a fragment of the directed tree G of the network. We observe that $\langle \boldsymbol{X} \mid \{V_i\} \mid \boldsymbol{Y} \rangle_G^d$ for all sets of vertices $\boldsymbol{X} \subseteq \boldsymbol{V}_{i_j}^-$ and $\boldsymbol{Y} \subseteq \bigcup_{k=1,\dots,m,k\neq j} \boldsymbol{V}_{i_k}^-$, $j = 1, \dots, m$. Since G is a directed I-map for the distribution Pr, we have that $I_{\Pr}(\boldsymbol{X}, \{V_i\}, \boldsymbol{Y})$ for all sets $\boldsymbol{X} \subseteq \boldsymbol{V}_{i_j}^-$, $\boldsymbol{Y} \subseteq \bigcup_{k=1,\dots,m,k\neq j} \boldsymbol{V}_{i_k}^-$, $j = 1, \dots, m$. It follows that

$$\lambda_{V_i}(V_i) = \Pr(\tilde{c}_{\boldsymbol{V}_{i_1}^-} \mid V_i) \cdot \ldots \cdot \Pr(\tilde{c}_{\boldsymbol{V}_{i_m}^-} \mid V_i)$$

The probabilities $\Pr(\tilde{c}_{\boldsymbol{V}_{i_j}^-} \mid V_i)$ are sent to vertex V_i by its successor V_{i_j} through the values of diagnostic parameter $\lambda_{V_{i_j}}^{V_i}$, $j = 1, \dots, m$. The property stated follows by substitution. □

Introducing dummy successors Note that the previous lemma applies to *uninstantiated* vertices only. However, the property mentioned in the lemma can be taken to hold for an *instantiated* vertex V_i as well, if entering evidence for V_i into the network is modelled by adding a 'dummy' successor D for vertex V_i that sends an appropriate diagnostic parameter to V_i. For the evidence $V_i = \mathit{true}$, this 'dummy' successor sends the diagnostic parameter $\lambda_D^{V_i}$ with $\lambda_D^{V_i}(v_i) = 1$ and $\lambda_D^{V_i}(\neg v_i) = 0$ to V_i; an analogous observation holds for the evidence $V_i = \mathit{false}$.

Computing Message Parameters in Directed Trees

So far, we have shown that a vertex can compute the probabilities of its values from its own assessment function and the causal and diagnostic message parameters it receives from its neighbours. Now observe that this vertex in turn has to compute message parameters to send to its neighbours. The following lemma shows that a vertex can compute the diagnostic message parameter to send to its predecessor from its own assessment function and the diagnostic parameters it receives from its successors. In other words, for this purpose it combines its own information about the joint probability distribution with the information it receives concerning the evidence entered so far for its descendants.

Lemma 4.2.6 *Let $\mathcal{B} = (G, \Gamma)$ be a Bayesian network where $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ is a directed tree. Let $V_i \in \boldsymbol{V}_G$ be a vertex with the predecessor V_j in G. Let λ_{V_i} be the compound diagnostic parameter for vertex V_i and let $\lambda_{V_i}^{V_j}$ be the diagnostic parameter from V_i to V_j. Then,*

$$\lambda_{V_i}^{V_j}(V_j) = \sum_{c_{V_i}} \lambda_{V_i}(c_{V_i}) \cdot \gamma_{V_i}(c_{V_i} \mid V_j)$$

Proof. Let Pr be the joint probability distribution defined by the Bayesian network $\mathcal{B}$. For vertex V_i, let $\boldsymbol{V_i^-}$ be as before. Then, by definition we have that

$$\lambda_{V_i}^{V_j}(V_j) = \Pr(\widetilde{c}_{\boldsymbol{V_i^-}} \mid V_j)$$

By conditioning on the values of V_i, we find

$$\lambda_{V_i}^{V_j}(V_j) = \Pr(\widetilde{c}_{\boldsymbol{V_i^-}} \mid v_i \wedge V_j) \cdot \Pr(v_i \mid V_j) + \Pr(\widetilde{c}_{\boldsymbol{V_i^-}} \mid \neg v_i \wedge V_j) \cdot \Pr(\neg v_i \mid V_j)$$

Now consider once more Figure 4.2 showing a fragment of the directed tree G of the network. Using the d-separation criterion, we observe that $\langle \boldsymbol{X} \mid \{V_i\} \mid \{V_j\} \rangle_G^d$ for all sets of vertices $\boldsymbol{X} \subseteq \boldsymbol{V_i^-}$. Since G is a directed I-map for the distribution Pr, it follows that $I_{\Pr}(\boldsymbol{X}, \{V_i\}, \{V_j\})$ for all sets $\boldsymbol{X} \subseteq \boldsymbol{V_i^-}$. So,

$$\lambda_{V_i}^{V_j}(V_j) = \Pr(\widetilde{c}_{\boldsymbol{V_i^-}} \mid v_i) \cdot \Pr(v_i \mid V_j) + \Pr(\widetilde{c}_{\boldsymbol{V_i^-}} \mid \neg v_i) \cdot \Pr(\neg v_i \mid V_j)$$

The probabilities $\Pr(v_i \mid V_j)$ and $\Pr(\neg v_i \mid V_j)$ have been specified as function values of the assessment function γ_{V_i} associated with vertex V_i and hence are directly available to V_i. In addition, V_i computes the probabilities $Pr(\widetilde{c}_{\boldsymbol{V_i^-}} \mid v_i)$ and $Pr(\widetilde{c}_{\boldsymbol{V_i^-}} \mid \neg v_i)$ as function values of its compound diagnostic parameter λ_{V_i}. The property stated in the lemma now follows by substitution. □

Similarly, a vertex can compute the causal parameter to send to a successor from its compound causal parameter and the diagnostic parameters it receives from its other successors. The following lemma states this property more formally.

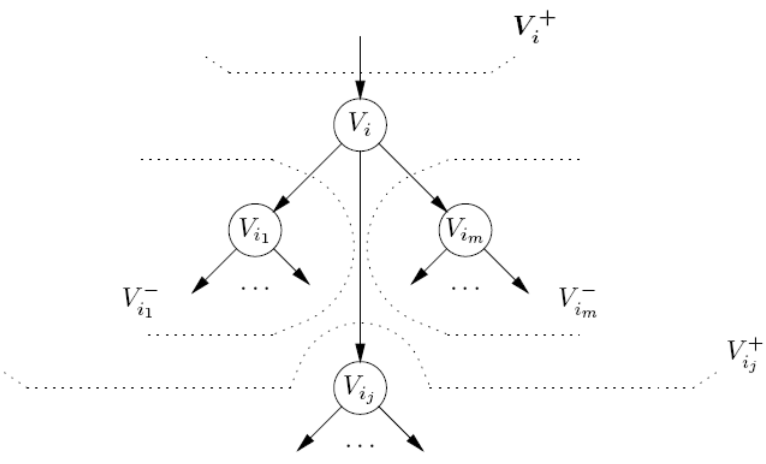


Figure 4.4: Exploiting d-Separation for Computing $\pi^{V_i}_{V_{i_j}}(V_i)$.

Lemma 4.2.7 *Let $\mathcal{B} = (G, \Gamma)$ be a Bayesian network where $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ is a directed tree. Let $V_i \in \boldsymbol{V}_G$ be an uninstantiated vertex with the successors $\boldsymbol{\sigma}_G(V_i) = \{V_{i_1}, \ldots, V_{i_m}\}$, $m \geq 1$, in G. Furthermore, let π_{V_i} be the compound causal parameter for vertex V_i; for each $V_{i_j} \in \boldsymbol{\sigma}_G(V_i)$, let $\pi^{V_i}_{V_{i_j}}$ be the causal parameter from V_i to V_{i_j} and let $\lambda^{V_i}_{V_{i_j}}$ be the diagnostic parameter from V_{i_j} to V_i. Then,*

$$\pi^{V_i}_{V_{i_j}}(V_i) = \alpha \cdot \pi_{V_i}(V_i) \cdot \prod_{k=1,\ldots,m,k\neq j} \lambda^{V_i}_{V_{i_k}}(V_i)$$

where α is a normalisation constant.

Proof. Let Pr be the joint probability distribution defined by the Bayesian network $\mathcal{B}$. For each vertex $V \in \boldsymbol{V}_G$, let $\boldsymbol{V}^+$ and $\boldsymbol{V}^-$ be as before. Then, by definition we have that

$$\pi^{V_i}_{V_{i_j}}(V_i) = \Pr(V_i \mid \widetilde{c}_{\boldsymbol{V}^+_{i_j}})$$

Using Bayes' Theorem, we find

$$\pi^{V_i}_{V_{i_j}}(V_i) = \frac{\Pr(\widetilde{c}_{\boldsymbol{V}^+_{i_j}} \mid V_i) \cdot \Pr(V_i)}{\Pr(\widetilde{c}_{\boldsymbol{V}^+_{i_j}})}$$

Now consider Figure 4.4 showing a fragment of the directed tree G of the network. Since V_i is an uninstantiated vertex, we have that $\widetilde{c}_{\boldsymbol{V}^+_{i_j}} = \widetilde{c}_{\boldsymbol{V}^+_i} \wedge (\bigwedge_{k=1,\ldots,m,k\neq j} \widetilde{c}_{\boldsymbol{V}^-_{i_k}})$. So,

$$\pi^{V_i}_{V_{i_j}}(V_i) = \frac{\Pr(\widetilde{c}_{\boldsymbol{V}^+_i} \wedge (\bigwedge_{k=1,\ldots,m,k\neq j} \widetilde{c}_{\boldsymbol{V}^-_{i_k}}) \mid V_i) \cdot \Pr(V_i)}{\Pr(\widetilde{c}_{\boldsymbol{V}^+_{i_j}})}$$

Furthermore, using the d-separation criterion, we observe that $\langle \boldsymbol{X} \mid \{V_i\} \mid \boldsymbol{Y} \rangle^d_G$ for all sets of vertices $\boldsymbol{X} \subseteq \boldsymbol{V}^+_i$ and $\boldsymbol{Y} \subseteq \boldsymbol{V}^-_{i_k}$, $k = 1, \ldots, m$, and for all sets of vertices $\boldsymbol{X} \subseteq \boldsymbol{V}^-_{i_k}$ and

$\boldsymbol{Y} \subseteq \boldsymbol{V}_{\boldsymbol{i_l}}^-$, $k = 1, \ldots, m$, $l = 1, \ldots, m$, $k \neq l$. Exploiting this observation, we find

$$\begin{aligned}\pi_{V_{i_j}}^{V_i}(V_i) &= \frac{\Pr(\widetilde{c}_{\boldsymbol{V}_{\boldsymbol{i}}^+} \mid V_i) \cdot \prod_{k=1,\ldots,m,k\neq j} \Pr(\widetilde{c}_{\boldsymbol{V}_{\boldsymbol{i_k}}^-} \mid V_i) \cdot \Pr(V_i)}{\Pr(\widetilde{c}_{\boldsymbol{V}_{\boldsymbol{i_j}}^+})} = \\ &= \frac{\Pr(V_i \mid \widetilde{c}_{\boldsymbol{V}_{\boldsymbol{i}}^+}) \cdot \prod_{k=1,\ldots,m,k\neq j} \Pr(\widetilde{c}_{\boldsymbol{V}_{\boldsymbol{i_k}}^-} \mid V_i) \cdot \Pr(\widetilde{c}_{\boldsymbol{V}_{\boldsymbol{i}}^+})}{\Pr(\widetilde{c}_{\boldsymbol{V}_{\boldsymbol{i_j}}^+})}\end{aligned}$$

The probabilities $\Pr(V_i \mid \widetilde{c}_{\boldsymbol{V}_{\boldsymbol{i}}^+})$ equal the function values of the compound causal parameter π_{V_i} for V_i. The probabilities $Pr(\widetilde{c}_{\boldsymbol{V}_{\boldsymbol{i_k}}^-} \mid V_i)$ equal the function values of the diagnostic parameter $\lambda_{V_{i_k}}^{V_i}$ vertex V_i receives from its successor V_{i_k}. In addition, we observe that the factor

$$\frac{\Pr(\widetilde{c}_{\boldsymbol{V}_{\boldsymbol{i}}^+})}{\Pr(\widetilde{c}_{\boldsymbol{V}_{\boldsymbol{i_j}}^+})} = \frac{1}{\Pr(\widetilde{c}_{\boldsymbol{V}_{\boldsymbol{i_j}}^+} \mid \widetilde{c}_{\boldsymbol{V}_{\boldsymbol{i}}^+})}$$

is dependent on the *location* of the variables V_i and V_{i_j} in the digraph but *not* on their values. This factor may therefore be looked upon as a normalisation constant for V_i and V_{i_j}, denoted as α; it can be computed from $\Pr(v_i \mid \widetilde{c}_{\boldsymbol{V}_{\boldsymbol{i_j}}^+}) + \Pr(\neg v_i \mid \widetilde{c}_{\boldsymbol{V}_{\boldsymbol{i_j}}^+}) = 1$. The property stated in the lemma now follows by substitution. □

The previous lemma applies to uninstantiated vertices only; the lemma, however, can be taken to hold for instantiated vertices as described before.

Pearl's Algorithm in Directed Trees

The data fusion lemma and the four computation rules provided by the Lemmas 4.2.4, 4.2.5, 4.2.6, and 4.2.7 with each other constitute Pearl's algorithm for probabilistic inference with a Bayesian network comprising a directed tree for its qualitative part. Note that Pearl's algorithm provides for computing probabilities as well as for processing evidence in such a Bayesian network. The computation rules for the separate causal and diagnostic message parameters enable a vertex to pass on the impact of a piece of evidence correctly, and allow for the evidence to spread throughout the network. Close examination of the computation rules from the Lemmas 4.2.6 and 4.2.7 further reveals that a vertex that sends an updated parameter will not receive a new parameter originating from the same evidence. A causal parameter or a diagnostic parameter *to* a vertex is not affected by the diagnostic parameter or the causal parameter, respectively, *from* that vertex. In addition, the topological property that in a directed tree there is at most one chain between any two vertices prohibits the process of parameter updating that originates from some vertex to reach this vertex along another chain. These properties with each other guarantee that feedback and circular reasoning are prevented and that evidence is propagated throughout the network in a single pass.

We state an additional property concerning the compound diagnostic parameter for a vertex that is useful for investigating the spreading of evidence. We call this the *Identity property* for compound diagnostic parameters.

Lemma 4.2.8 *Let $\mathcal{B} = (G, \Gamma)$ be a Bayesian network where $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ is a directed tree. For each vertex $V_i \in \boldsymbol{V}_G$, let $\lambda_{V_i}(V_i)$ be the compound diagnostic parameter for V_i. If $\widetilde{c}_{\boldsymbol{V}_G} = \mathsf{True}$, then $\lambda_{V_i}(V_i) = 1$ for all $V_i \in \boldsymbol{V}_G$.*

Proof. The property stated in the lemma can be easily proven directly from the definition of the compound diagnostic parameter (Definition 4.2.2). Here we present an alternative proof by (reverse) induction on the depth of the directed tree G. Let n be the maximal depth of the tree.

Induction Basis
The property holds for every leaf of the tree at depth n by definition.

Induction Hypothesis
For a specific $d \leq n$, we assume that $\lambda_{V_i}(V_i) = 1$ for all vertices V_i at depth $d, d+1, \ldots, n$.

Induction Step
Now consider a vertex V_i at depth $d-1$ in the tree. We distinguish between two cases. If V_i is a leaf of the tree, then $\lambda_{V_i}(V_i) = 1$ by definition. Now, suppose that V_i has m successors $V_{i_1}, \ldots, V_{i_m}$, $m \geq 1$. From $\widetilde{c}_{\boldsymbol{V}_G} = \mathsf{True}$, it follows that V_i is an uninstantiated vertex. Therefore, it follows from the property stated in Lemma 4.2.5 that

$$\lambda_{V_i}(V_i) = \prod_{j=1,\ldots,m} \lambda_{V_{i_j}}^{V_i}(V_i)$$

where $\lambda_{V_{i_j}}^{V_i}$ is the diagnostic parameter from V_{i_j} to V_i, $j = 1, \ldots, m$. For each parameter $\lambda_{V_{i_j}}^{V_i}$ we have from the property stated in Lemma 4.2.6 that

$$\lambda_{V_{i_j}}^{V_i}(V_i) = \sum_{c_{V_{i_j}}} \lambda_{V_{i_j}}(c_{V_{i_j}}) \cdot \gamma_{V_{i_j}}(c_{V_{i_j}} \mid V_i)$$

Since vertex V_{i_j} is a successor of V_i, it is located at depth d in the directed tree G. From the induction hypothesis we have that $\lambda_{V_{i_j}}(V_{i_j}) = 1$. So,

$$\lambda_{V_{i_j}}^{V_i}(V_i) = \sum_{c_{V_{i_j}}} \gamma_{V_{i_j}}(c_{V_{i_j}} \mid V_i) = 1$$

From $\lambda_{V_{i_j}}^{V_i}(V_i) = 1$ for all successors V_{i_j} of V_i, it follows that $\lambda_{V_i}(V_i) = 1$. Since vertex V_i has been chosen arbitrarily, it follows that for each vertex $V_i \in \boldsymbol{V}_G$, we have $\lambda_{V_i}(V_i) = 1$.
□

It will be evident that the previous lemma can be taken to apply to subtrees of a directed tree as well.

We conclude our discussion of Pearl's algorithm for probabilistic inference so far with an example.

Example 4.2.9 We consider the Bayesian network $\mathcal{B} = (G, \Gamma)$ shown in Figure 4.5. Let Pr be the joint probability distribution defined by $\mathcal{B}$. We address the computation of the probabilities and different parameters for the various vertices in the network.

Vertex V_1 sets out to compute the probabilities of its values by application of the data fusion lemma:

$$\Pr(v_1) \; = \alpha \cdot \pi_{V_1}(v_1) \cdot \lambda_{V_1}(v_1) \quad \text{and} \quad \Pr(\neg v_1) \; = \alpha \cdot \pi_{V_1}(\neg v_1) \cdot \lambda_{V_1}(\neg v_1)$$

To be able to compute the probabilities of interest from the lemma, vertex V_1 needs to compute its compound parameters π_{V_1} and λ_{V_1}. Since no evidence has been entered into

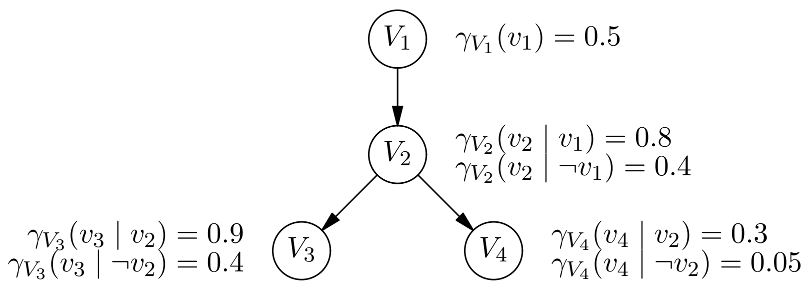

V_1
$\gamma_{V_1}(v_1) = 0.5$
V_2
$\gamma_{V_2}(v_2 \mid v_1) = 0.8$
$\gamma_{V_2}(v_2 \mid \neg v_1) = 0.4$
$\gamma_{V_3}(v_3 \mid v_2) = 0.9$
$\gamma_{V_3}(v_3 \mid \neg v_2) = 0.4$
V_3
V_4
$\gamma_{V_4}(v_4 \mid v_2) = 0.3$
$\gamma_{V_4}(v_4 \mid \neg v_2) = 0.05$

Figure 4.5: An Example Bayesian Network.

the network as yet, we have from the Identity property stated in Lemma 4.2.8 that the values of the compound diagnostic parameter λ_{V_1} equal

$$\lambda_{V_1}(v_1) \;= 1 \quad \text{and} \quad \lambda_{V_1}(\neg v_1) \;= 1$$

Vertex V_1 now turns to the computation of its compound causal parameter π_{V_1}. Using the computation rule from in Lemma 4.2.4, it finds the values of this parameter to be

$$\pi_{V_1}(v_1) \;= 0.5 \quad \text{and} \quad \pi_{V_1}(\neg v_1) \;= 0.5$$

Substitution of the values of the compound parameters into the data fusion lemma and subsequent elimination of the normalisation constant α yields

$$\Pr(v_1) \;= 0.5 \quad \text{and} \quad \Pr(\neg v_1) \;= 0.5$$

Now observe that vertex V_1 not just computes its probabilities, it also computes the message parameter to send to its neighbour in the tree. Using the computation rule stated in Lemma 4.2.7, it computes the values of the causal parameter $\pi_{V_2}^{V_1}$ for its successor V_2 to be

$$\pi_{V_2}^{V_1}(v_1) \;= 0.5 \quad \text{and} \quad \pi_{V_2}^{V_1}(\neg v_1) \;= 0.5$$

By sending the thus computed message parameter to vertex V_2, vertex V_1 provides V_2 with sufficient information about the represented probability distribution to enable vertex V_2 to compute its probabilities.

Just as vertex V_1, vertex V_2 sets out to compute the probabilities of its values by application of the data fusion lemma:

$$\Pr(v_2) \;= \alpha \cdot \pi_{V_2}(v_2) \cdot \lambda_{V_2}(v_2) \quad \text{and} \quad \Pr(\neg v_2) \;= \alpha \cdot \pi_{V_2}(\neg v_2) \cdot \lambda_{V_2}(\neg v_2)$$

Vertex V_2 now needs to compute its compound parameters π_{V_2} and λ_{V_2}. Since no evidence has been entered into the network as yet, we once more have from the Identity property stated in Lemma 4.2.8 that the values of the compound diagnostic parameter equal

$$\lambda_{V_2}(v_2) \;= 1 \quad \text{and} \quad \lambda_{V_2}(\neg v_2) \;= 1$$

Using the computation rule stated in Lemma 4.2.4, vertex V_2 now computes the compound causal parameter π_{V_2} from its own probability assessment function γ_{V_2} and the causal parameter $\pi_{V_2}^{V_1}$ it receives from its predecessor. For the values $\pi_{V_2}(v_2)$ and $\pi_{V_2}(\neg v_2)$, it thus finds

$$\begin{aligned}
\pi_{V_2}(v_2) \;&= \gamma_{V_2}(v_2 \mid v_1) \cdot \pi_{V_2}^{V_1}(v_1) + \gamma_{V_2}(v_2 \mid \neg v_1) \cdot \pi_{V_2}^{V_1}(\neg v_1) = \\
&= 0.8 \cdot 0.5 + 0.4 \cdot 0.5 = 0.6 \\
\pi_{V_2}(\neg v_2) \;&= \gamma_{V_2}(\neg v_2 \mid v_1) \cdot \pi_{V_2}^{V_1}(v_1) + \gamma_{V_2}(\neg v_2 \mid \neg v_1) \cdot \pi_{V_2}^{V_1}(\neg v_1) = \\
&= 0.2 \cdot 0.5 + 0.6 \cdot 0.5 = 0.4
\end{aligned}$$

Substitution of the values of the compound parameters into the data fusion lemma and subsequent elimination of the normalisation constant α yields

$$\Pr(v_2) \; = 0.6 \quad \text{and} \quad \Pr(\neg v_2) \; = 0.4$$

In addition to computing its own probabilities, vertex V_2 computes the message parameters to send to its neighbours in the tree. For its predecessor V_1, vertex V_2 computes the diagnostic parameter $\lambda^{V_1}_{V_2}$; the values of this message parameter equal

$$\lambda^{V_1}_{V_2}(v_1) \; = 1 \quad \text{and} \quad \lambda^{V_1}_{V_2}(\neg v_1) \; = 1$$

Vertex V_2 further computes a causal parameter for every one of its successors in the tree. Using the computation rule stated in Lemma 4.2.7, it computes the values of the causal parameter $\pi^{V_2}_{V_3}$ for vertex V_3 from

$$\begin{aligned} \pi^{V_2}_{V_3}(v_2) \quad &= \alpha \cdot \pi_{V_2}(v_2) \cdot \lambda^{V_2}_{V_4}(v_2) \\ \pi^{V_2}_{V_3}(\neg v_2) \quad &= \alpha \cdot \pi_{V_2}(\neg v_2) \cdot \lambda^{V_2}_{V_4}(\neg v_2) \end{aligned}$$

It is readily seen that the values of the diagnostic parameter $\lambda^{V_2}_{V_4}$ that vertex V_2 receives from vertex V_4 both equal one. Substitution of the various parameter values and subsequent elimination of the normalisation constant α now yields

$$\pi^{V_2}_{V_3}(v_2) \; = 0.6 \quad \text{and} \quad \pi^{V_2}_{V_3}(\neg v_2) \; = 0.4$$

For vertex V_4, vertex V_2 equally computes

$$\pi^{V_2}_{V_4}(v_2) \; = 0.6 \quad \text{and} \quad \pi^{V_2}_{V_4}(\neg v_2) \; = 0.4$$

By sending the thus computed message parameters to its respective neighbours, vertex V_2 enables these neighbours to compute their probabilities in turn. Vertex V_3 now computes the probabilities of its values to be

$$\Pr(v_3) \; = 0.7 \quad \text{and} \quad \Pr(\neg v_3) \; = 0.3$$

Vertex V_4 computes its probabilities to be

$$\Pr(v_4) \; = 0.2 \quad \text{and} \quad \Pr(\neg v_4) \; = 0.8$$

Processing evidence Now, suppose that the evidence $V_4 = \mathit{true}$ is observed and entered into the Bayesian network $\mathcal{B}$. We address the computation of the posterior probabilities and the different parameters for the various vertices in the network using Pearl's algorithm. Vertex V_4 once more sets out to compute the probabilities of its values by application of the data fusion lemma:

$$\Pr^{v_4}(v_4) \; = \alpha \cdot \pi_{V_4}(v_4) \cdot \lambda_{V_4}(v_4) \quad \text{and} \quad \Pr^{v_4}(\neg v_4) \; = \alpha \cdot \pi_{V_4}(\neg v_4) \cdot \lambda_{V_4}(\neg v_4)$$

Since the evidence $V_4 = \mathit{true}$ has been entered for a dummy successor of vertex V_4, and V_4 has no other successors, V_4 finds the values of its compound diagnostic parameter λ_{V_4} to be

$$\lambda_{V_4}(v_4) \; = 1 \quad \text{and} \quad \lambda_{V_4}(\neg v_4) \; = 0$$

resulting in

$$\Pr^{v_4}(v_4) \; = 1 \quad \text{and} \quad \Pr^{v_4}(\neg v_4) \; = 0$$

as expected. In addition to updating its own probabilities, vertex V_4 computes a new message parameter to send to its neighbour in the tree. Using the computation rule stated in Lemma 4.2.6, vertex V_4 computes the values of the diagnostic parameter $\lambda^{V_2}_{V_4}$ for its predecessor V_2 to be

$$\begin{aligned}
\lambda^{V_2}_{V_4}(v_2) &= \gamma_{V_4}(v_4 \mid v_2) \cdot \lambda_{V_4}(v_4) + \gamma_{V_4}(\neg v_4 \mid v_2) \cdot \lambda_{V_4}(\neg v_4) = \\
&= 0.3 \cdot 1 + 0.7 \cdot 0 = 0.3 \\
\lambda^{V_2}_{V_4}(\neg v_2) &= \gamma_{V_4}(v_4 \mid \neg v_2) \cdot \lambda_{V_4}(v_4) + \gamma_{V_4}(\neg v_4 \mid \neg v_2) \cdot \lambda_{V_4}(\neg v_4) = \\
&= 0.05 \cdot 1 + 0.95 \cdot 0 = 0.05
\end{aligned}$$

By sending the thus computed message parameter to vertex V_2, vertex V_4 informs V_2 of the newly entered evidence, enabling it to update its probabilities.

As before, vertex V_2 sets out to compute the probabilities of its values by application of the data fusion lemma:

$$\Pr^{v_4}(v_2) = \alpha \cdot \pi_{V_2}(v_2) \cdot \lambda_{V_2}(v_2) \quad \text{and} \quad \Pr^{v_4}(\neg v_2) = \alpha \cdot \pi_{V_2}(\neg v_2) \cdot \lambda_{V_2}(\neg v_2)$$

Since no evidence has been entered for vertex V_2's non-descendants, its compound causal parameter π_{V_2} is not affected and remains to be

$$\pi_{V_2}(v_2) = 0.6 \quad \text{and} \quad \pi_{V_2}(\neg v_2) = 0.4$$

Using the computation rule stated in Lemma 4.2.5, vertex V_2 computes the values of its compound diagnostic parameter to be

$$\lambda_{V_2}(v_2) = \lambda^{V_2}_{V_3}(v_2) \cdot \lambda^{V_2}_{V_4}(v_2) \quad \text{and} \quad \lambda_{V_2}(\neg v_2) = \lambda^{V_2}_{V_3}(\neg v_2) \cdot \lambda^{V_2}_{V_4}(\neg v_2)$$

It is readily seen from the Identity property that the values of the diagnostic parameter $\lambda^{V_2}_{V_3}$ that vertex V_2 receives from vertex V_3 both equal one. Substitution of the various parameter values now yields

$$\lambda_{V_2}(v_2) = 0.3 \quad \text{and} \quad \lambda_{V_2}(\neg v_2) = 0.05$$

Vertex V_2 substitutes the values of its compound parameters π_{V_2} and λ_{V_2} into the data fusion lemma to find

$$\Pr^{v_4}(v_2) = \alpha \cdot 0.6 \cdot 0.3 = \alpha \cdot 0.18 \quad \text{and} \quad \Pr^{v_4}(\neg v_2) = \alpha \cdot 0.4 \cdot 0.05 = \alpha \cdot 0.02$$

After eliminating the normalisation constant α it finds

$$\Pr^{v_4}(v_2) = 0.9 \quad \text{and} \quad \Pr^{v_4}(\neg v_2) = 0.1$$

In addition to updating its own probabilities, vertex V_2 computes new message parameters to send to its other neighbours than vertex V_4. Using the computation rule stated in Lemma 4.2.7, vertex V_2 computes the values of the causal parameter $\pi^{V_2}_{V_3}$ for its successor V_3 to be

$$\begin{aligned}
\pi^{V_2}_{V_3}(v_2) &= \alpha \cdot \pi_{V_2}(v_2) \cdot \lambda^{V_2}_{V_4}(v_2) = \alpha \cdot 0.6 \cdot 0.3 = 0.9 \\
\pi^{V_2}_{V_3}(\neg v_2) &= \alpha \cdot \pi_{V_2}(\neg v_2) \cdot \lambda^{V_2}_{V_4}(\neg v_2) = \alpha \cdot 0.4 \cdot 0.05 = 0.1
\end{aligned}$$

Using the computation rule stated in Lemma 4.2.6, vertex V_2 computes the values of the diagnostic parameter $\lambda^{V_1}_{V_2}$ for its predecessor V_1 to be

$$\begin{aligned}
\lambda^{V_1}_{V_2}(v_1) &= \gamma_{V_2}(v_2 \mid v_1) \cdot \lambda_{V_2}(v_2) + \gamma_{V_2}(\neg v_2 \mid v_1) \cdot \lambda_{V_2}(\neg v_2) = \\
&= 0.8 \cdot 0.3 + 0.2 \cdot 0.05 = 0.25 \\
\lambda^{V_1}_{V_2}(\neg v_1) &= \gamma_{V_2}(v_2 \mid \neg v_1) \cdot \lambda_{V_2}(v_2) + \gamma_{V_2}(\neg v_2 \mid \neg v_1) \cdot \lambda_{V_2}(\neg v_2) = \\
&= 0.4 \cdot 0.3 + 0.6 \cdot 0.05 = 0.15
\end{aligned}$$

By sending the thus computed parameters to vertex V_3 and vertex V_1, respectively, vertex V_2 enables these vertices to compute their updated probabilities. Vertex V_3 computes the updated probabilities of its values to be

$$\Pr^{v_4}(v_3) \; = 0.85 \quad \text{and} \quad \Pr^{v_4}(\neg v_3) \; = 0.15$$

Vertex V_1 computes its updated probabilities to be

$$\Pr^{v_4}(v_1) \; = 0.625 \quad \text{and} \quad \Pr^{v_4}(\neg v_1) \; = 0.375$$

□

4.2.2 Singly Connected Digraphs

So far, we have only discussed Pearl's algorithm for probabilistic inference for Bayesian networks comprising a directed tree for their qualitative part. In this section, we extend the algorithm to apply to Bayesian networks of which the qualitative part is a *singly connected digraph* more in general. Before detailing the computations involved in the extended algorithm, we introduce some new terminology and notational conventions that we will use in the sequel. We consider a singly connected digraph G. For this graph G, we observe that removal of any arc splits the graph into two separate components. From this property we have that in the digraph G we can identify for a vertex V_i with m neighbours, m subgraphs of G each containing a neighbour of V_i such that, after removal of V_i and all its incoming and outgoing arcs, there does not exist a path from one such subgraph to another. The following definition introduces these subgraphs more formally; Figure 4.6 illustrates the basic idea.

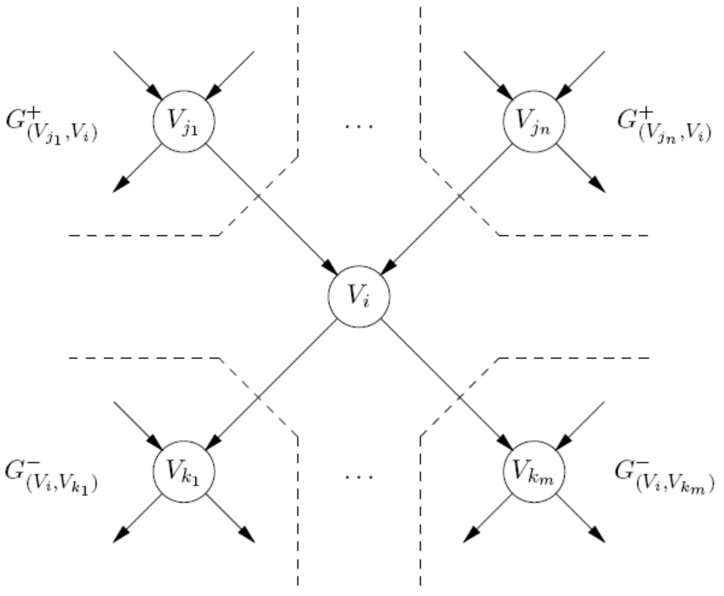


Figure 4.6: Upper and Lower Graphs.

Definition 4.2.10 *Let $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ be a singly connected directed graph. For each arc $(V_i, V_j) \in \boldsymbol{A}_G$, let $G_{(V_i,V_j)} = (\boldsymbol{V}_G, \boldsymbol{A}_G \backslash \{(V_i, V_j)\})$ be the digraph that results after removing arc (V_i, V_j) from G. Now, let $V_i \in \boldsymbol{V}_G$ be a vertex in G. For each predecessor $V_j \in \boldsymbol{\rho}_G(V_i)$ of V_i, let $G^+_{(V_j,V_i)}$ be the component of $G_{(V_j,V_i)}$ that includes V_j in its vertex set; $G^+_{(V_j,V_i)}$ is called an* upper graph *of vertex V_i. For each successor $V_k \in \boldsymbol{\sigma}_G(V_i)$ of V_i, let $G^-_{(V_i,V_k)}$ be the component of $G_{(V_i,V_k)}$ that includes V_k in its vertex set; $G^-_{(V_i,V_k)}$ is called a* lower graph *of V_i.*

In the previous section, we have detailed Pearl's computation rules for probabilistic inference with a Bayesian network comprising a directed tree: we recall that these rules address the computation of the different parameters for the various vertices in the tree. The proofs of the lemmas that we have presented show that these computation rules derive from *exploiting independences.* These independences are read from the qualitative part of a Bayesian network by *local inspection of a vertex' incoming and outgoing arcs only.* The computation rules therefore make use explicitly of the property that in the network's digraph there is at most one chain between any two vertices. Since singly connected digraphs share this property with directed trees, Pearl's tree algorithm is extended straightforwardly to apply to Bayesian networks comprising a singly connected digraph for their qualitative part. In fact, only the computation rules for the compound causal parameter and for the diagnostic message parameters are adapted to account for a vertex having multiple predecessors; all other computation rules remain unaltered. For the sake of completeness, we will review all computation rules involved.

Defining Compound Parameters

We begin by addressing the computation of probabilities from a Bayesian network comprising a singly connected digraph. It will be evident that the probabilities of the values of a vertex of interest are dependent upon all evidence entered into the network so far. As we have seen for the vertices in a directed tree, the probabilities of the values of a vertex in a singly connected digraph can be written in terms of two factors. These factors describe the combined influence on the vertex' probabilities of evidence entered for the vertices in its upper graphs and of evidence entered for the vertices in its lower graphs, respectively. We redefine the compound causal and diagnostic parameters for a vertex to capture these factors in view of singly connected digraphs.

Definition 4.2.11 *Let $\mathcal{B} = (G, \Gamma)$ be a Bayesian network where $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ is a singly connected digraph, and let* Pr *be the joint probability distribution defined by $\mathcal{B}$. Let $V_i \in \boldsymbol{V}_G$ be a vertex in G, and let $\boldsymbol{V_i^+} = \bigcup_{V_j \in \boldsymbol{\rho}_G(V_i)} \boldsymbol{V}_{G^+_{(V_j,V_i)}}$ and $\boldsymbol{V_i^-} = \boldsymbol{V}_G \setminus \boldsymbol{V_i^+}$. The* compound causal parameter π_{V_i} *for vertex V_i is the function $\pi_{V_i} \colon V_i \rightarrow [0,1]$ defined by*

$$\pi_{V_i}(V_i) = \Pr(V_i \mid \tilde{c}_{\boldsymbol{V_i^+}})$$

The compound diagnostic parameter λ_{V_i} *for V_i is the function $\lambda_{V_i} \colon V_i \rightarrow [0,1]$ defined by*

$$\lambda_{V_i}(V_i) = \Pr(\tilde{c}_{\boldsymbol{V_i^-}} \mid V_i)$$

Note that this redefinition of the compound parameters differs from Definition 4.2.2 with respect to the sets $\boldsymbol{V_i^+}$ and $\boldsymbol{V_i^-}$ for a vertex V_i. The basic idea of the parameters, however, remains unaltered. The compound causal parameter for a vertex describes the combined influence on its probabilities of all evidence that has been entered 'above' it in the digraph; the compound diagnostic parameter for the vertex describes the combined influence of all evidence that has been entered 'below' it in the digraph.

Special cases and data fusion reconsidered Note that again, for a vertex W without any incoming arcs, we once more find $\pi_W(W) = \Pr(W)$; for a vertex V without any outgoing arcs, we again have $\lambda_V(V) = 1$. We will refer to such vertices as *roots* and *leafs*, respectively, even in de context of a singly connected graph.

Using the redefinition of the compound causal and diagnostic parameters for a vertex, the *data fusion lemma* now applies to a Bayesian network comprising a singly connected digraph for its qualitative part: for each vertex V_i in the network, we once more have

$$\Pr(V_i \mid \tilde{c}_{\boldsymbol{V}_G}) = \alpha \cdot \pi_{V_i}(V_i) \cdot \lambda_{V_i}(V_i) \text{ with normalisation constant } \alpha.$$

Defining Message Parameters

The compound diagnostic and causal parameters for a vertex specify probabilistic information from its lower graphs combined and from all its upper graphs combined, respectively. We observe that to be able to exploit the qualitative part of a Bayesian network as a computational architecture, these compound parameters once again have to be computed from causal and diagnostic message parameters originating from the various neighbours of the vertex. We redefine these message parameters before addressing the computation of the compound ones.

Definition 4.2.12 *Let $\mathcal{B} = (G, \Gamma)$ be a Bayesian network where $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ is a singly connected digraph, and let* Pr *be the joint probability distribution defined by $\mathcal{B}$. Let V_i be a vertex with a successor V_k in G and let $G^+_{(V_i,V_k)}$ be as before. The* causal (message) parameter $\pi^{V_i}_{V_k}$ from V_i to V_k *is the function $\pi^{V_i}_{V_k} : V_i \rightarrow [0,1]$ defined by*

$$\pi^{V_i}_{V_k}(V_i) = \Pr(V_i \mid \widetilde{c}_{\boldsymbol{V}_{G^+_{(V_i,V_k)}}})$$

Now, let V_i be a vertex having a predecessor V_j in G and let $G^-_{(V_j,V_i)}$ be as before. The diagnostic (message) parameter $\lambda^{V_j}_{V_i}$ from V_i to V_j *is the function $\lambda^{V_j}_{V_i} : V_j \rightarrow [0,1]$ defined by*

$$\lambda^{V_j}_{V_i}(V_j) = \Pr(\widetilde{c}_{\boldsymbol{V}_{G^-_{(V_j,V_i)}}} \mid V_j)$$

The causal and diagnostic message parameters defined above once again are the messages that the vertices send one another through the communication channels of the computational architecture and, hence, may again be looked upon as associated with the arcs of the digraph of the Bayesian network at hand.

Computing Compound Parameters

The following lemma now shows that a vertex can compute its compound causal parameter from its own assessment function and the causal message parameters it receives from its various predecessors.

Lemma 4.2.13 *Let $\mathcal{B} = (G, \Gamma)$ be a Bayesian network where $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ is a singly connected digraph. Let $V_i \in \boldsymbol{V}_G$ be a vertex with the predecessors $\boldsymbol{\rho}_G(V_i) = \{V_{j_1}, \dots, V_{j_n}\}$, $n \geq 1$, in G. Furthermore, let π_{V_i} be the compound causal parameter for vertex V_i and, for each $V_{j_k} \in \boldsymbol{\rho}_G(V_i)$, let $\pi^{V_{j_k}}_{V_i}$ be the causal parameter from V_{j_k} to V_i. Then,*

$$\pi_{V_i}(V_i) = \sum_{c_{\boldsymbol{\rho}_G(V_i)}} \gamma_{V_i}(V_i \mid c_{\boldsymbol{\rho}_G(V_i)}) \cdot \prod_{k=1,\dots,n} \pi^{V_{j_k}}_{V_i}(c_{V_{j_k}})$$

where $c_{\boldsymbol{\rho}_G(V_i)} = \bigwedge_{k=1,\dots,n} c_{V_{j_k}}$.

Proof. Let Pr be the joint probability distribution defined by the Bayesian network $\mathcal{B}$. For vertex V_i, let $\boldsymbol{V}^+_{\boldsymbol{i}} = \bigcup_{k=1,\dots,n} \boldsymbol{V}_{G^+_{(V_{j_k},V_i)}}$ be as before. Then, by definition we have that

$$\pi_{V_i}(V_i) \;=\; \Pr(V_i \mid \widetilde{c}_{\boldsymbol{V}^+_{\boldsymbol{i}}}) \;=\; \Pr(V_i \mid \widetilde{c}_{\boldsymbol{V}_{G^+_{(V_{j_1},V_i)}}} \wedge \cdots \wedge \widetilde{c}_{\boldsymbol{V}_{G^+_{(V_{j_n},V_i)}}})$$

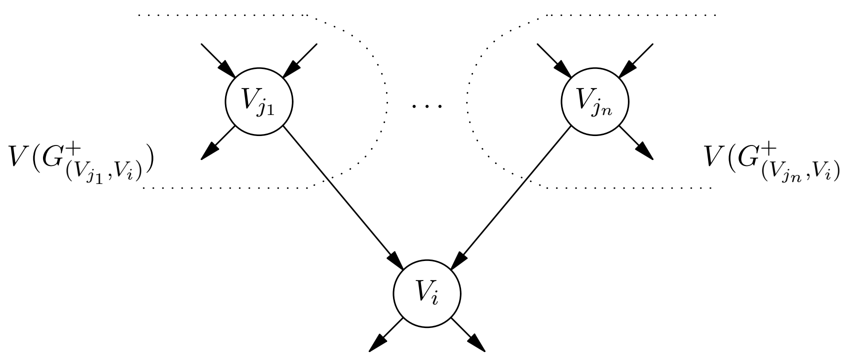


Figure 4.7: Exploiting d-Separation for Computing $\pi_{V_i}(V_i)$.

By conditioning on the various configurations of the set $\boldsymbol{\rho}_G(V_i)$ of V_i's predecessors, we find

$$\begin{aligned}\pi_{V_i}(V_i) &= \sum_{c_{\boldsymbol{\rho}_G(V_i)}} \Pr(V_i \mid c_{\boldsymbol{\rho}_G(V_i)} \wedge \widetilde{c}_{\boldsymbol{V}_{G^+_{(V_{j_1},V_i)}}} \wedge \cdots \wedge \widetilde{c}_{\boldsymbol{V}_{G^+_{(V_{j_n},V_i)}}})\cdot \\ &\qquad \cdot \Pr(c_{\boldsymbol{\rho}_G(V_i)} \mid \widetilde{c}_{\boldsymbol{V}_{G^+_{(V_{j_1},V_i)}}} \wedge \cdots \wedge \widetilde{c}_{\boldsymbol{V}_{G^+_{(V_{j_n},V_i)}}})\end{aligned}$$

Now consider Figure 4.7 showing a fragment of the singly connected digraph G of the network. Using the d-separation criterion, we observe that $\langle \{V_i\} \mid \boldsymbol{\rho}_G(V_i) \mid \boldsymbol{X}\rangle^d_G$ for all sets of vertices $\boldsymbol{X} \subseteq \boldsymbol{V_i^+}$. Since G is a directed I-map for the distribution Pr, it follows that $I_{\Pr}(\{V_i\}, \boldsymbol{\rho}_G(V_i), \boldsymbol{X})$ for all sets $\boldsymbol{X} \subseteq \boldsymbol{V_i^+}$. Exploiting this observation, we find

$$\pi_{V_i}(V_i) = \sum_{c_{\boldsymbol{\rho}_G(V_i)}} \Pr(V_i \mid c_{\boldsymbol{\rho}_G(V_i)}) \cdot \Pr(c_{\boldsymbol{\rho}_G(V_i)} \mid \widetilde{c}_{\boldsymbol{V}_{G^+_{(V_{j_1},V_i)}}} \wedge \cdots \wedge \widetilde{c}_{\boldsymbol{V}_{G^+_{(V_{j_n},V_i)}}})$$

By once more using the d-separation criterion, we find that $I_{\Pr}(\boldsymbol{X}, \{V_{j_k}\}, \boldsymbol{Y})$ for all sets of vertices $\boldsymbol{X} \subseteq \boldsymbol{V}_{G^+_{(V_{j_k},V_i)}}$, $\boldsymbol{Y} \subseteq \boldsymbol{V}_{G^+_{(V_{j_l},V_i)}}$, $l = 1,\ldots,n$, $k = 1,\ldots,n$, $k \neq l$. Exploiting this observation, we find

$$\pi_{V_i}(V_i) = \sum_{c_{\boldsymbol{\rho}_G(V_i)}} \Pr(V_i \mid c_{\boldsymbol{\rho}_G(V_i)}) \cdot \Pr(c_{V_{j_1}} \mid \widetilde{c}_{\boldsymbol{V}_{G^+_{(V_{j_1},V_i)}}}) \cdot \ldots \cdot \Pr(c_{V_{j_n}} \mid \widetilde{c}_{\boldsymbol{V}_{G^+_{(V_{j_n},V_i)}}})$$

where $c_{\boldsymbol{\rho}_G(V_i)} = \bigwedge_{k=1,\ldots,n} c_{V_{j_k}}$. The probabilities $\Pr(V_i \mid c_{\boldsymbol{\rho}_G(V_i)})$ have been specified as function values of the assessment function γ_{V_i} and therefore are directly available to vertex V_i. In addition, V_i receives the probabilities $\Pr(c_{V_{j_k}} \mid \widetilde{c}_{\boldsymbol{V}_{G^+_{(V_{j_k},V_i)}}})$ from its predecessor V_{j_k} as function values of the causal parameter $\pi^{V_{j_k}}_{V_i}$, $k = 1,\ldots,n$. The property stated in the lemma now follows by substitution. □

A vertex can further compute its compound diagnostic parameter from the diagnostic message parameters it receives from its successors in the digraph. This property is analogous to the property stated in Lemma 4.2.5, that is, for an uninstantiated vertex $V_i \in \boldsymbol{V}_G$ with successors $\boldsymbol{\sigma}_G(V_i) = \{V_{i_1}, \ldots, V_{i_m}\}$, $m \geq 1$, in the digraph, we have that

$$\lambda_{V_i}(V_i) = \prod_{j=1,\ldots,m} \lambda^{V_i}_{V_{i_j}}(V_i)$$

The property can be taken to apply to instantiated vertices using dummy vertices as outlined before. The Identity property stated in Lemma 4.2.8 also holds for singly connected digraphs.

Computing Message Parameters

A vertex in turn has to compute message parameters to send to its various neighbours. A vertex V_i can compute the diagnostic parameter to send to a predecessor V_{j_k} from its own probability assessment function, its compound diagnostic parameter, and the causal message parameters it receives from its other predecessors.

Lemma 4.2.14 *Let $\mathcal{B} = (G, \Gamma)$ be a Bayesian network where $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ is a singly connected graph. Let $V_i \in \boldsymbol{V}_G$ be a vertex with predecessors $\boldsymbol{\rho}_G(V_i) = \{V_{j_1}, \ldots, V_{j_n}\}$, $n \geq 1$, in G. Let λ_{V_i} be the compound diagnostic parameter for vertex V_i, and let $\pi_{V_i}^{V_{j_l}}$ be the causal parameter from $V_{j_l} \in \boldsymbol{\rho}_G(V_i)$ to V_i. Futhermore, for each $V_{j_k} \in \boldsymbol{\rho}_G(V_i)$, let $\lambda_{V_i}^{V_{j_k}}$ be the diagnostic parameter from V_i to V_{j_k}. Then,*

$$\begin{aligned}\lambda_{V_i}^{V_{j_k}}(V_{j_k}) &= \alpha \cdot \sum_{c_{V_i}} \lambda_{V_i}(c_{V_i}) \cdot \left[\sum_{c_{\boldsymbol{\rho}_G(V_i)\setminus\{V_{j_k}\}}} \Big(\gamma_{V_i}(c_{V_i} \mid c_{\boldsymbol{\rho}_G(V_i)\setminus\{V_{j_k}\}} \wedge V_{j_k}) \cdot \right. \\ &\qquad \left. \cdot \prod_{l=1,\ldots,n, l\neq k} \pi_{V_i}^{V_{j_l}}(c_{V_{j_l}}) \Big) \right]\end{aligned}$$

where $c_{\boldsymbol{\rho}_G(V_i)\setminus\{V_{j_k}\}} = \bigwedge_{l=1,\ldots,n,l\neq k} c_{V_{j_l}}$ and α is a normalisation constant.

Proof. Let Pr be the joint probability distribution defined by the Bayesian network $\mathcal{B}$. Let $\boldsymbol{V_i^-}$, G^- and G^+ be as before. Then, by definition we have that

$$\lambda_{V_i}^{V_{j_k}}(V_{j_k}) \;=\; \Pr(\widetilde{c}_{\boldsymbol{V}_{G^-_{(V_{j_k}, V_i)}}} \mid V_{j_k}) \;=\; \Pr(\widetilde{c}_{\boldsymbol{V_i^-}} \wedge \widetilde{c}_{\boldsymbol{V_i^{+\setminus k}}}) \mid V_{j_k}),$$

where $\boldsymbol{V_i^{+\setminus k}} = \bigcup_{h=1,\ldots,n,h\neq k} \boldsymbol{V}_{G^+_{(V_{j_h}, V_i)}}$ captures the vertices in the upper graphs of all predecessors of V_i except V_{j_k}.

By conditioning on the values of V_i, we now find

$$\lambda_{V_i}^{V_{j_k}}(V_{j_k}) = \sum_{c_{V_i}} \Pr(\widetilde{c}_{\boldsymbol{V_i^-}} \wedge \widetilde{c}_{\boldsymbol{V_i^{+\setminus k}}} \mid c_{V_i} \wedge V_{j_k}) \cdot \Pr(c_{V_i} \mid V_{j_k})$$

Using d-separation, it now follows that given V_i, $\boldsymbol{V_i^-}$ is independent of $\boldsymbol{V} \setminus \boldsymbol{V_i^-}$. So,

$$\begin{aligned}\lambda_{V_i}^{V_{j_k}}(V_{j_k}) &= \sum_{c_{V_i}} \Pr(\widetilde{c}_{\boldsymbol{V_i^-}} \mid c_{V_i} \wedge V_{j_k}) \cdot \Pr(\widetilde{c}_{\boldsymbol{V_i^{+\setminus k}}} \mid c_{V_i} \wedge V_{j_k}) \cdot \Pr(c_{V_i} \mid V_{j_k}) = \\ &= \sum_{c_{V_i}} \Pr(\widetilde{c}_{\boldsymbol{V_i^-}} \mid c_{V_i}) \cdot \Pr(c_{V_i} \mid V_{j_k}) \cdot \frac{\Pr(c_{V_i} \mid \widetilde{c}_{\boldsymbol{V_i^{+\setminus k}}} \wedge V_{j_k}) \cdot \Pr(\widetilde{c}_{\boldsymbol{V_i^{+\setminus k}}} \mid V_{j_k})}{\Pr(c_{V_i} \mid V_{j_k})} \\ &= \sum_{c_{V_i}} \Pr(\widetilde{c}_{\boldsymbol{V_i^-}} \mid c_{V_i}) \cdot \Pr(c_{V_i} \mid \widetilde{c}_{\boldsymbol{V_i^{+\setminus k}}} \wedge V_{j_k}) \cdot \Pr(\widetilde{c}_{\boldsymbol{V_i^{+\setminus k}}} \mid V_{j_k})\end{aligned}$$

Vertex V_i computes the probabilities $Pr(\widetilde{c}_{\boldsymbol{V_i^-}} \mid c_{V_i})$ as function values of its compound diagnostic parameter λ_{V_i}. d-Separation tells us that all predecessors of V_i are independent of each other, so

$$\Pr(\widetilde{c}_{\boldsymbol{V_i^{+\setminus k}}} \mid V_{j_k}) = \prod_{h=1,\ldots,n,h\neq k} \Pr(\widetilde{c}_{\boldsymbol{V}_{G^+_{(V_{j_h}, V_i)}}})$$

The latter factor is constant with respect to the values of V_{j_k} and may therefore be looked upon as a normalisation constant, which we denote by α.[2]

We now focus on the term $\Pr(c_{V_i} \mid \widetilde{c}_{\boldsymbol{V_i^{+\backslash k}}} \wedge V_{j_k})$ and condition on the values of all predecessors of V_i except V_{j_k}:

$$\sum_{c_{\boldsymbol{\rho}_G(V_i)\backslash\{V_{j_k}\}}} \Pr(c_{V_i} \mid \widetilde{c}_{\boldsymbol{V_i^{+\backslash k}}} \wedge V_{j_k} \wedge c_{\boldsymbol{\rho}_G(V_i)\backslash\{V_{j_k}\}}) \cdot \Pr(c_{\boldsymbol{\rho}_G(V_i)\backslash\{V_{j_k}\}} \mid \widetilde{c}_{\boldsymbol{V_i^{+\backslash k}}} \wedge V_{j_k})$$

Using d-separation, it now follows that given V_i's predecessors, V_i is independent of $\boldsymbol{V}\backslash\boldsymbol{V_i^-}$; in addition, recall that all predecessors of V_i are independent of each other. So the above term reduces to

$$\begin{aligned} & \sum_{c_{\boldsymbol{\rho}_G(V_i)\backslash\{V_{j_k}\}}} \Pr(c_{V_i} \mid V_{j_k} \wedge c_{\boldsymbol{\rho}_G(V_i)\backslash\{V_{j_k}\}}) \cdot \prod_{l=1,\ldots,n,l\neq k} \Pr(c_{V_{j_l}} \mid \widetilde{c}_{\boldsymbol{V_i^{+\backslash k}}} \wedge V_{j_k}) \\ = & \sum_{c_{\boldsymbol{\rho}_G(V_i)\backslash\{V_{j_k}\}}} \Pr(c_{V_i} \mid V_{j_k} \wedge c_{\boldsymbol{\rho}_G(V_i)\backslash\{V_{j_k}\}}) \cdot \prod_{l=1,\ldots,n,l\neq k} \Pr(c_{V_{j_l}} \mid \widetilde{c}_{\boldsymbol{V}^+_{G(V_{j_l},V_i)}}) \end{aligned}$$

The probabilities $\Pr(c_{V_i} \mid V_{j_k} \wedge c_{\boldsymbol{\rho}_G(V_i)\backslash\{V_{j_k}\}})$ have been specified as function values of the assessment function γ_{V_i} associated with vertex V_i and hence are directly available to V_i. In addition, V_i receives the probabilities $\Pr(c_{V_{j_l}} \mid \widetilde{c}_{\boldsymbol{V}^+_{G(V_{j_l},V_i)}})$ from each of its predecessors $\boldsymbol{\rho}_G(V_i) \setminus \{V_{j_k}\}$ as function values of their causal parameters $\pi^{V_{j_l}}_{V_i}$. The property stated in the lemma now follows by substitution. □

Finally, if we consider a vertex V_i with successors $\boldsymbol{\sigma}_G(V_i) = \{V_{i_1}, \ldots, V_{i_m}\}$ in the digraph, then the causal message parameter vertex V_i has to send to a successor, is computed from its compound causal parameter and the diagnostic parameters it receives from its other successors; this property is analogous to the property stated in Lemma 4.2.7:

$$\pi^{V_i}_{V_{i_j}}(V_i) = \alpha \cdot \pi_{V_i}(V_i) \cdot \prod_{k=1,\ldots,m,k\neq j} \lambda^{V_i}_{V_{i_k}}(V_i)$$

where α once more is a normalisation constant.

Pearl's Algorithm in Singly Connected Graphs

The data fusion lemma and the four computation rules mentioned above with each other constitute Pearl's algorithm for probabilistic inference with a Bayesian network comprising a singly connected digraph for its qualitative part. Once more, these computation rules provide for computing probabilities as well as for processing evidence in such a Bayesian network. The computation rules for the causal and diagnostic message parameters enable a vertex to correctly pass on the impact of a piece of evidence to its neighbours. As feedback and circular reasoning once more are excluded, evidence is thus propagated throughout the network in a single pass.

Example 4.2.15 We consider once more the Bayesian network $\mathcal{B} = (G, \Gamma)$ from Example 4.1.2; for ease of reference, the network is reproduced in Figure 4.8. Note that the digraph G of $\mathcal{B}$ is not a directed tree, yet is singly connected.

[2]Note that in this case α cannot be trivially computed, since $\lambda^{V_{j_k}}_{V_i}(v_{j_k}) + \lambda^{V_{j_k}}_{V_i}(\neg v_{j_k})$ typically does not equal 1. For this reason, in applying Pearl's algorithm, normalisation is usually postponed to the final step of data-fusion.

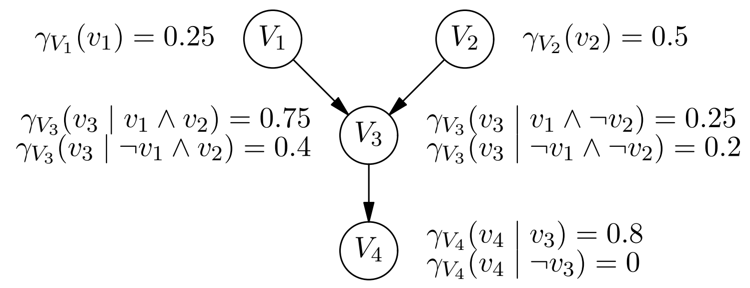

$\gamma_{V_1}(v_1) = 0.25$
V_1
V_2
$\gamma_{V_2}(v_2) = 0.5$
$\gamma_{V_3}(v_3 \mid v_1 \wedge v_2) = 0.75$
$\gamma_{V_3}(v_3 \mid \neg v_1 \wedge v_2) = 0.4$
V_3
$\gamma_{V_3}(v_3 \mid v_1 \wedge \neg v_2) = 0.25$
$\gamma_{V_3}(v_3 \mid \neg v_1 \wedge \neg v_2) = 0.2$
V_4
$\gamma_{V_4}(v_4 \mid v_3) = 0.8$
$\gamma_{V_4}(v_4 \mid \neg v_3) = 0$

Figure 4.8: The Example Bayesian Network Reproduced.

Let Pr be the joint probability distribution defined by the network $\mathcal{B}$. We address the computation of the probabilities $\Pr(v_3)$ and $\Pr(\neg v_3)$ using Pearl's algorithm. Vertex V_3 sets out to compute the probabilities of interest by application of the data fusion lemma:

$$\Pr(v_3) \; = \alpha \cdot \pi_{V_3}(v_3) \cdot \lambda_{V_3}(v_3) \quad \text{and} \quad \Pr(\neg v_3) \; = \alpha \cdot \pi_{V_3}(\neg v_3) \cdot \lambda_{V_3}(\neg v_3)$$

Vertex V_3 now needs to compute its compound parameters π_{V_3} and λ_{V_3}. Since no evidence has been entered into the network as yet, we have that the values of the compound diagnostic parameter λ_{V_3} equal

$$\lambda_{V_3}(v_3) \; = 1 \quad \text{and} \quad \lambda_{V_3}(\neg v_3) \; = 1 \qquad \text{(Identity property)}$$

Vertex V_3 computes the values of its compound causal parameter π_{V_3} from

$$\begin{aligned}
\pi_{V_3}(v_3) \; &= \gamma_{V_3}(v_3 \mid v_1 \wedge v_2) \cdot \pi_{V_3}^{V_1}(v_1) \cdot \pi_{V_3}^{V_2}(v_2) + \\
&\quad + \gamma_{V_3}(v_3 \mid \neg v_1 \wedge v_2) \cdot \pi_{V_3}^{V_1}(\neg v_1) \cdot \pi_{V_3}^{V_2}(v_2) + \\
&\quad + \gamma_{V_3}(v_3 \mid v_1 \wedge \neg v_2) \cdot \pi_{V_3}^{V_1}(v_1) \cdot \pi_{V_3}^{V_2}(\neg v_2) + \\
&\quad + \gamma_{V_3}(v_3 \mid \neg v_1 \wedge \neg v_2) \cdot \pi_{V_3}^{V_1}(\neg v_1) \cdot \pi_{V_3}^{V_2}(\neg v_2) \\
\pi_{V_3}(\neg v_3) \; &= \gamma_{V_3}(\neg v_3 \mid v_1 \wedge v_2) \cdot \pi_{V_3}^{V_1}(v_1) \cdot \pi_{V_3}^{V_2}(v_2) + \\
&\quad + \gamma_{V_3}(\neg v_3 \mid \neg v_1 \wedge v_2) \cdot \pi_{V_3}^{V_1}(\neg v_1) \cdot \pi_{V_3}^{V_2}(v_2) + \\
&\quad + \gamma_{V_3}(\neg v_3 \mid v_1 \wedge \neg v_2) \cdot \pi_{V_3}^{V_1}(v_1) \cdot \pi_{V_3}^{V_2}(\neg v_2) + \\
&\quad + \gamma_{V_3}(\neg v_3 \mid \neg v_1 \wedge \neg v_2) \cdot \pi_{V_3}^{V_1}(\neg v_1) \cdot \pi_{V_3}^{V_2}(\neg v_2)
\end{aligned}$$

It can be readily shown that vertex V_3 receives from its predecessor V_1 the causal parameter $\pi_{V_3}^{V_1}$ with the values

$$\pi_{V_3}^{V_1}(v_1) = \pi_{V_1}(v_1) = \gamma_{V_1}(v_1) \; = 0.25 \quad \text{and} \quad \pi_{V_3}^{V_1}(\neg v_1) \; = 0.75$$

The above steps follow from the property of *Causal parameter equivalence* (introduced in Exercise 4.3(c)) and the fact that V_1 is a root vertex. Likewise, vertex V_3 receives from its predecessor V_2 the causal parameter $\pi_{V_3}^{V_2}$ with the values

$$\pi_{V_3}^{V_2}(v_2) \; = 0.5 \quad \text{and} \quad \pi_{V_3}^{V_2}(\neg v_2) \; = 0.5$$

Substitution now yields the following values for V_3's compound causal parameter:

$$\begin{aligned}
\pi_{V_3}(v_3) \; &= 0.75 \cdot 0.25 \cdot 0.5 + 0.4 \cdot 0.75 \cdot 0.5 + 0.25 \cdot 0.25 \cdot 0.5 + 0.2 \cdot 0.75 \cdot 0.5 = \\
&= 0.35 \\
\pi_{V_3}(\neg v_3) \; &= 0.25 \cdot 0.25 \cdot 0.5 + 0.6 \cdot 0.75 \cdot 0.5 + 0.75 \cdot 0.25 \cdot 0.5 + 0.8 \cdot 0.75 \cdot 0.5 = \\
&= 0.65
\end{aligned}$$

Subsequent substitution of the values of the compound parameters into the data fusion lemma and elimination of the normalisation constant α yields

$$\Pr(v_3) \; = 0.35 \quad \text{and} \quad \Pr(\neg v_3) \; = 0.65$$

Processing evidence Now, suppose that the evidence $V_3 = \mathit{true}$ is observed and entered into the Bayesian network. We address the computation of the posterior probabilities $\Pr^{v_3}(v_1)$ and $\Pr^{v_3}(\neg v_1)$ using Pearl's algorithm. Vertex V_1 sets out to compute the probabilities of interest by application of the data fusion lemma:

$$\Pr^{v_3}(v_1) = \alpha \cdot \pi_{V_1}(v_1) \cdot \lambda_{V_1}(v_1) \quad \text{and} \quad \Pr^{v_3}(\neg v_1) = \alpha \cdot \pi_{V_1}(\neg v_1) \cdot \lambda_{V_1}(\neg v_1)$$

Root vertex V_1 now has to compute its compound parameters π_{V_1} and λ_{V_1}. For the compound causal parameter, it finds the values

$$\pi_{V_1}(v_1) = 0.25 \quad \text{and} \quad \pi_{V_1}(\neg v_1) = 0.75$$

The values of the compound diagnostic parameter λ_{V_1} are computed from

$$\lambda_{V_1}(v_1) = \lambda_{V_3}^{V_1}(v_1) \quad \text{and} \quad \lambda_{V_1}(\neg v_1) = \lambda_{V_3}^{V_1}(\neg v_1)$$

From its successor V_3, vertex V_1 receives the diagnostic parameter $\lambda_{V_3}^{V_1}$. The computation rule for this parameter includes compound parameter $\lambda_{V_3}(V_3)$ that is set through the 0-1 message V_3 receives from its dummy successor to reflect its instantiation. In addition, it includes causal message parameter $\pi_{V_3}^{V_2}(V_2)$, which is not affected by the instantiation and remains unchanged. We therefore find the following values

$$\begin{aligned}
\lambda_{V_3}^{V_1}(v_1) = {} & \lambda_{V_3}(v_3) \cdot \left[\gamma_{V_3}(v_3 \mid v_1 \wedge v_2) \cdot \pi_{V_3}^{V_2}(v_2) + \gamma_{V_3}(v_3 \mid v_1 \wedge \neg v_2) \cdot \pi_{V_3}^{V_2}(\neg v_2)\right] \\
& + \lambda_{V_3}(\neg v_3) \cdot \left[\gamma_{V_3}(\neg v_3 \mid v_1 \wedge v_2) \cdot \pi_{V_3}^{V_2}(v_2) + \gamma_{V_3}(\neg v_3 \mid v_1 \wedge \neg v_2) \cdot \pi_{V_3}^{V_2}(\neg v_2)\right] \\
= {} & \gamma_{V_3}(v_3 \mid v_1 \wedge v_2) \cdot \pi_{V_3}^{V_2}(v_2) + \gamma_{V_3}(v_3 \mid v_1 \wedge \neg v_2) \cdot \pi_{V_3}^{V_2}(\neg v_2) = \\
= {} & 0.75 \cdot 0.5 + 0.25 \cdot 0.5 = 0.5 \\
\lambda_{V_3}^{V_1}(\neg v_1) = {} & \lambda_{V_3}(v_3) \cdot \left[\gamma_{V_3}(v_3 \mid \neg v_1 \wedge v_2) \cdot \pi_{V_3}^{V_2}(v_2) + \gamma_{V_3}(v_3 \mid \neg v_1 \wedge \neg v_2) \cdot \pi_{V_3}^{V_2}(\neg v_2)\right] \\
& + \lambda_{V_3}(\neg v_3) \cdot \left[\gamma_{V_3}(\neg v_3 \mid \neg v_1 \wedge v_2) \cdot \pi_{V_3}^{V_2}(v_2) + \gamma_{V_3}(\neg v_3 \mid \neg v_1 \wedge \neg v_2) \cdot \pi_{V_3}^{V_2}(\neg v_2)\right] \\
= {} & \gamma_{V_3}(v_3 \mid \neg v_1 \wedge v_2) \cdot \pi_{V_3}^{V_2}(v_2) + \gamma_{V_3}(v_3 \mid \neg v_1 \wedge \neg v_2) \cdot \pi_{V_3}^{V_2}(\neg v_2) \\
= {} & 0.4 \cdot 0.5 + 0.2 \cdot 0.5 = 0.3
\end{aligned}$$

From the diagnostic parameter $\lambda_{V_3}^{V_1}$ it receives from its successor V_3, vertex V_1 now finds the values of its compound diagnostic parameter to be

$$\lambda_{V_1}(v_1) = 0.5 \quad \text{and} \quad \lambda_{V_1}(\neg v_1) = 0.3$$

Substitution of the values of the compound parameters into the data fusion lemma yields

$$\begin{aligned}
\Pr^{v_3}(v_1) &= \alpha \cdot 0.25 \cdot 0.5 = \alpha \cdot 0.125 \\
\Pr^{v_3}(\neg v_1) &= \alpha \cdot 0.75 \cdot 0.3 = \alpha \cdot 0.225
\end{aligned}$$

After elimination of the normalisation constant α (note that we postponed all normalisation until this last step!), vertex V_1 finds

$$\Pr^{v_3}(v_1) = 0.357 \quad \text{and} \quad \Pr^{v_3}(\neg v_1) = 0.643$$

□

Complexity of Pearl's Algorithm

To conclude our discussion of Pearl's algorithm so far, we take a (superficial) look at the algorithm's computational complexity. An analysis of the computation rules involved in the algorithm serves to show that it has a *worst-case* computational time complexity that is *exponential* in the number of vertices in a network's digraph. We consider, in a Bayesian network with n vertices, a vertex V_i with $O(n)$ predecessors and $O(n)$ successors. For this vertex, computing the compound causal parameter requires at most $O(2^n)$ computations; computing its compound diagnostic parameter requires at most $O(n)$ computations. Vertex V_i can therefore compute the probabilities of its values in at most $O(2^n)$ time. The computation of a single diagnostic message parameter from V_i requires at most $O(2^n)$ computations; the computation of a single causal message parameter requires only constant time. Vertex V_i can therefore compute all parameters to send to its various neighbours in at most $O(n \cdot 2^n)$ time. Propagating a single piece of evidence throughout the network may thus require $O(n^2 \cdot 2^n)$ computations. The exponential factor in the computational time complexity of Pearl's algorithm arises solely from the number of predecessors a vertex can have in a network's digraph: if the predecessor sets are bounded in size by a constant, then the algorithm has a *linear* runtime complexity.

4.2.3 Multiply Connected Digraphs

In the previous section, we have detailed Pearl's algorithm for probabilistic inference with a Bayesian network comprising a singly connected digraph for its qualitative part. We will now extend the algorithm to apply to Bayesian networks of which the qualitative part is a *multiply connected digraph.*

We recall from our discussion of Pearl's algorithm so far, that the computation rules for probabilistic inference with a Bayesian network comprising a singly connected digraph derive from exploiting independences that are read from the network's graph by local inspection of a vertex' incoming and outgoing arcs. These computation rules therefore make use explicitly of the property that in a singly connected digraph there is at most one chain between any two vertices. We observe that this property does not hold in a multiply connected digraph as in such a graph there may be multiple chains between vertices. Local inspection of a vertex' incident arcs now no longer suffices for reading independences from a Bayesian network's digraph. In fact, application of Pearl's algorithm discussed so far to a Bayesian network comprising a multiply connected digraph inevitably leads to various problems due to the presence of loops in the network's digraph [Suermondt & Cooper, 1990]: vertices may indefinitely send newly updated messages, originating from the same evidence, to their neighbours, causing the network never to reach a new equilibrium, and even if the network *does* reach an equilibrium, it is not guaranteed to reflect the correct updated joint probability distribution. Pearl's algorithm, therefore, cannot be extended based on similar ideas as before to apply to networks with a multiply connected digraph: the algorithm has to be supplemented with an additional method for coping with the loops in such a digraph.

The best-known method for coping with loops in probabilistic inference is the method of *loop cutset conditioning* [Pearl, 1988]. The idea underlying the method of loop cutset conditioning is that of *reasoning by assumption.* For a multiply connected digraph of a Bayesian network, vertices are selected that, upon instantiation, with each other effectively 'cut' all loops of the digraph and cause it to 'behave' as if it were singly connected; the selected vertices are said to constitute a *loop cutset* for the network's digraph. During probabilistic inference with the network, each configuration of the selected loop cutset is looked upon as a (compound) assumption on which reasoning is performed: for each vertex, the probabilities of its values are computed by conditioning successively on all possible configurations of the loop cutset and subsequently weighting the results obtained with the probabilities of these configurations. Since upon instantiation of the loop cutset the digraph behaves as if it were singly connected, the probabilities of a vertex' values can simply be computed from the network using Pearl's basic algorithm for Bayesian networks

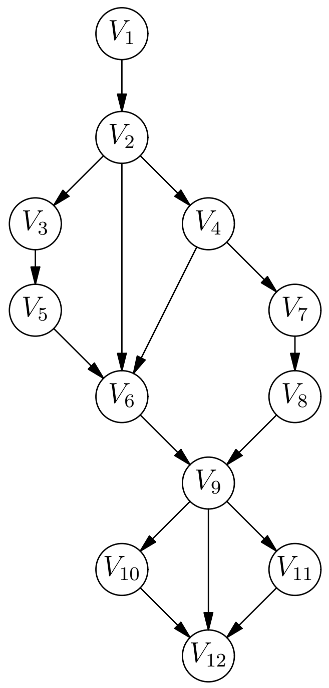


Figure 4.9: An Example Multiply Connected Digraph.

with a singly connected digraph.

Loop Cutsets

We begin our discussion of the method of loop cutset conditioning by detailing the concept of a loop cutset for a Bayesian network's multiply connected digraph. A loop cutset has to provide, upon instantiation, for the multiply connected digraph to behave as if it were singly connected, regardless of the evidence that has been entered into the network at hand. Note that a loop cutset then serves to prevent, at any time, probabilistic information originating at a single vertex in the digraph to reach another vertex along multiple chains. For this purpose, a loop cutset has to cut, or *block*, every cyclic chain, or loop, in the network's digraph. Note that it does not suffice to block a cyclic chain by the empty set as this chain may become unblocked as further evidence is entered into the network. From these observations, we find that a loop cutset has to block all cyclic chains in the digraph in such a way that each such chain contains at least one vertex from the loop cutset that has an outgoing arc on the chain.

Definition 4.2.16 *Let $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ be an acyclic multiply connected digraph. A set of vertices $\boldsymbol{L}_G \subseteq \boldsymbol{V}_G$ is called a* loop cutset *for G if each loop s in G contains three consecutive vertices X_1, X_2, X_3, for which one of the following conditions holds:*

- *arcs (X_2, X_1) and (X_2, X_3) are on the loop s, and $X_2 \in \boldsymbol{L}_G$;*
- *arcs (X_1, X_2) and (X_2, X_3) are on the loop s, and $X_2 \in \boldsymbol{L}_G$.*

Note that a Bayesian network's multiply connected digraph G may allow several different loop cutsets; in fact, any superset of a loop cutset for G is also a loop cutset for G. Also note that any loop cutset for G is a *non-empty* set of vertices.

Example 4.2.17 We consider the multiply connected digraph G shown in Figure 4.9. The sets of vertices $\{V_2, V_6, V_9\}$, $\{V_3, V_4, V_9\}$, and $\{V_2, V_8, V_{10}, V_{11}\}$ are example loop cutsets for G. □

Loop Cutset Conditioning

After having detailed the concept of a loop cutset, we now turn to the method of loop cutset conditioning itself. To this end, we consider a Bayesian network $\mathcal{B} = (G, \Gamma)$ where $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ is an acyclic multiply connected digraph; let Pr be the joint probability distribution defined by $\mathcal{B}$. Furthermore, suppose that we have the loop cutset $\boldsymbol{L}_G$ for the digraph G of the network, as defined above. We now address computing the probabilities of the values of some vertex V_i in G, that is, we consider the computation of the probabilities $\Pr(V_i \mid \widetilde{c}_{\boldsymbol{V}_G})$. As the digraph G is multiply connected, these probabilities cannot be computed directly from the network using Pearl's algorithm. Since the loop cutset has been chosen so as to block, upon instantiation, all cyclic chains in the digraph, we observe that any probability that is conditioned on a configuration of (at least)the loop cutset can be computed from the network directly. Now, by conditioning the probabilities $\Pr(V_i \mid \widetilde{c}_{\boldsymbol{V}_G})$ of interest on the various configurations of the loop cutset $\boldsymbol{L}_G$ for G, we find

$$\Pr(V_i \mid \widetilde{c}_{\boldsymbol{V}_G}) = \sum_{c_{\boldsymbol{L}_G}} \Pr(V_i \mid \widetilde{c}_{\boldsymbol{V}_G} \wedge c_{\boldsymbol{L}_G}) \cdot \Pr(c_{\boldsymbol{L}_G} \mid \widetilde{c}_{\boldsymbol{V}_G})$$

The probabilities $\Pr(V_i \mid \widetilde{c}_{\boldsymbol{V}_G} \wedge c_{\boldsymbol{L}_G})$ are now conditioned on a configuration of (at least) the loop cutset and therefore can be computed directly from the Bayesian network using Pearl's basic algorithm. The probabilities $\Pr(c_{\boldsymbol{L}_G} \mid \widetilde{c}_{\boldsymbol{V}_G})$, however, cannot be computed from the Bayesian network directly. For computing these probabilities, the following recursive computation rules are used:

- the prior probabilities $\Pr(c_{\boldsymbol{L}_G})$ are computed from the joint probability distribution Pr using the property stated in Proposition 4.1.3 [Suermondt & Cooper, 1991].

- for the first piece of evidence e_1 entered into the Bayesian network, the (updated) probabilities of the configurations of the loop cutset are computed from

  $$\Pr(c_{\boldsymbol{L}_G} \mid e_1) = \alpha \cdot \Pr(e_1 \mid c_{\boldsymbol{L}_G}) \cdot \Pr(c_{\boldsymbol{L}_G})$$

  where α is a normalisation constant. The probabilities $\Pr(e_1 \mid c_{\boldsymbol{L}_G})$ are computed directly from the network using Pearl's basic algorithm for singly connected digraphs; the probabilities $\Pr(c_{\boldsymbol{L}_G})$ have been computed before and, hence, are already available.

- for the j-th piece of evidence e_j entered into the network, the (updated) probabilities of the configurations of the loop cutset are computed from

  $$\begin{aligned} \Pr(c_{\boldsymbol{L}_G} \mid e_1 \wedge \cdots \wedge e_j) \quad = \quad & \alpha \cdot \Pr(e_j \mid c_{\boldsymbol{L}_G} \wedge e_1 \wedge \cdots \wedge e_{j-1}) \cdot \\ & \cdot \Pr(c_{\boldsymbol{L}_G} \mid e_1 \wedge \cdots \wedge e_{j-1}) \end{aligned}$$

  where α once more is a normalisation constant. The probabilities $\Pr(e_j \mid c_{\boldsymbol{L}_G} \wedge e_1 \wedge \cdots \wedge e_{j-1})$ are computed directly from the Bayesian network using Pearl's basic algorithm; the probability $\Pr(c_{\boldsymbol{L}_G} \mid e_1 \wedge \cdots \wedge e_{j-1})$ has been computed before in the previous step of the recursion and need not be re-computed.

Note that the probabilities of the configurations of a digraph's loop cutset involve information from the probability distribution concerning various vertices combined. These probabilities therefore cannot be computed from the network by local message passing between neighbouring vertices alone. For this purpose, the computational architecture is enhanced with a *global supervisor* that is capable of performing the recursive computation rules outlined above.

We briefly illustrate the method of loop cutset conditioning by means of an example.

Example 4.2.18 We consider the Bayesian network $\mathcal{B} = (G, \Gamma)$ shown in Figure 4.10. Let Pr be the joint probability distribution defined by $\mathcal{B}$. We address the computation of the probabilities $\Pr(v_4)$ and $\Pr(\neg v_4)$ using Pearl's algorithm.

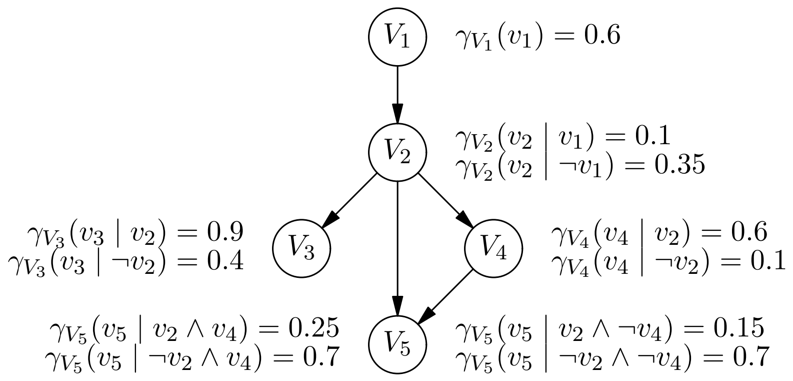

V_1
$\gamma_{V_1}(v_1) = 0.6$
V_2
$\gamma_{V_2}(v_2 \mid v_1) = 0.1$
$\gamma_{V_2}(v_2 \mid \neg v_1) = 0.35$
V_3
$\gamma_{V_3}(v_3 \mid v_2) = 0.9$
$\gamma_{V_3}(v_3 \mid \neg v_2) = 0.4$
V_4
$\gamma_{V_4}(v_4 \mid v_2) = 0.6$
$\gamma_{V_4}(v_4 \mid \neg v_2) = 0.1$
V_5
$\gamma_{V_5}(v_5 \mid v_2 \wedge v_4) = 0.25$
$\gamma_{V_5}(v_5 \mid \neg v_2 \wedge v_4) = 0.7$
$\gamma_{V_5}(v_5 \mid v_2 \wedge \neg v_4) = 0.15$
$\gamma_{V_5}(v_5 \mid \neg v_2 \wedge \neg v_4) = 0.7$

Figure 4.10: An Example Bayesian Network.

We begin by observing that the digraph G of the Bayesian network is multiply connected. To compute the probabilities of interest from the network, we extend Pearl's algorithm with the method of loop cutset conditioning. Suppose that for this purpose we choose for the digraph G the loop cutset $\boldsymbol{L}_G = \{V_2\}$. The global supervisor of the architecture now starts the computation by computing the probabilities of the configurations of the loop cutset. To this end, it employs the factorisation property stated in Proposition 4.1.3 to find from Γ that

$$\begin{aligned}
\Pr(v_2) \quad &= \Pr(v_1 \wedge v_2) + \Pr(\neg v_1 \wedge v_2) = \\
&= \gamma_{V_2}(v_2 \mid v_1) \cdot \gamma_{V_1}(v_1) + \gamma_{V_2}(v_2 \mid \neg v_1) \cdot \gamma(\neg v_1) = \\
&= 0.1 \cdot 0.6 + 0.35 \cdot 0.4 = 0.2 \\
\Pr(\neg v_2) \quad &= \Pr(v_1 \wedge \neg v_2) + \Pr(\neg v_1 \wedge \neg v_2) = \\
&= \gamma_{V_2}(\neg v_2 \mid v_1) \cdot \gamma_{V_1}(v_1) + \gamma_{V_2}(\neg v_2 \mid \neg v_1) \cdot \gamma(\neg v_1) = \\
&= 0.9 \cdot 0.6 + 0.65 \cdot 0.4 = 0.8
\end{aligned}$$

The probabilities of interest are now computed by conditioning on the different configurations of the chosen loop cutset:

$$\begin{aligned}
\Pr(v_4) \quad &= \Pr(v_4 \mid v_2) \cdot \Pr(v_2) + \Pr(v_4 \mid \neg v_2) \cdot \Pr(\neg v_2) \\
\Pr(\neg v_4) \quad &= \Pr(\neg v_4 \mid v_2) \cdot \Pr(v_2) + \Pr(\neg v_4 \mid \neg v_2) \cdot \Pr(\neg v_2)
\end{aligned}$$

The global supervisor *provisionally* enters the value *true* for vertex V_2 into the network to enable vertex V_4 to compute the probabilities $\Pr(v_4 \mid v_2)$ and $\Pr(\neg v_4 \mid v_2)$. Using the computation rules for a single connected graph, vertex V_4 computes these probabilities to be

$$\Pr(v_4 \mid v_2) \quad = 0.6 \quad \text{and} \quad \Pr(\neg v_4 \mid v_2) \quad = 0.4$$

The global supervisor now retracts the value *true* for vertex V_2 and subsequently enters the value *false* for the vertex. It now requests vertex V_4 to compute the probabilities $\Pr(v_4 \mid \neg v_2)$ and $\Pr(\neg v_4 \mid \neg v_2)$. Vertex V_4 returns

$$\Pr(v_4 \mid \neg v_2) \quad = 0.1 \quad \text{and} \quad \Pr(\neg v_4 \mid \neg v_2) \quad = 0.9$$

The global supervisor substitutes the probabilities returned by vertex V_4 and the probabilities it has itself computed for the loop cutset, into the conditioning formula to find

$$\Pr(v_4) \quad = 0.6 \cdot 0.2 + 0.1 \cdot 0.8 = 0.2 \quad \text{and} \quad \Pr(\neg v_4) \quad = 0.4 \cdot 0.2 + 0.9 \cdot 0.8 = 0.8$$

Processing evidence Now suppose that the evidence $V_3 = \textit{true}$ is observed and entered into the Bayesian network $\mathcal{B}$. We address the computation of the updated probabilities of the values of vertex V_4. The global supervisor begins with updating the probabilities of the configurations of the loop cutset according to the recursive computation rule (page 60)

$$\begin{array}{ll} \Pr(v_2 \mid v_3) & = \alpha \cdot \Pr(v_3 \mid v_2) \cdot \Pr(v_2) \qquad \text{and} \\ \Pr(\neg v_2 \mid v_3) & = \alpha \cdot \Pr(v_3 \mid \neg v_2) \cdot \Pr(\neg v_2) \end{array}$$

where α is a normalisation constant. The supervisor requests from vertex V_3 the probabilities $\Pr(v_3 \mid v_2)$ and $\Pr(v_3 \mid \neg v_2)$; to enable vertex V_3 to compute these probabilities, it provisionally enters and retracts the values *true* and *false* for vertex V_2, respectively. Vertex V_3 computes the requested probabilities to be

$$\Pr(v_3 \mid v_2) \ = 0.9 \quad \text{and} \quad \Pr(v_3 \mid \neg v_2) \ = 0.4$$

After substitution and subsequent elimination of the normalisation constant α, the global supervisor finds

$$\Pr(v_2 \mid v_3) \ = 0.36 \quad \text{and} \quad \Pr(\neg v_2 \mid v_3) \ = 0.64$$

The probabilities of interest are now computed by conditioning as before:

$$\begin{array}{ll} \Pr^{v_3}(v_4) & = \Pr(v_4 \mid v_2 \wedge v_3) \cdot \Pr(v_2 \mid v_3) + \Pr(v_4 \mid \neg v_2 \wedge v_3) \cdot \Pr(\neg v_2 \mid v_3) = \\ & = 0.6 \cdot 0.36 + 0.1 \cdot 0.64 = 0.28 \\ \Pr^{v_3}(\neg v_4) & = \Pr(\neg v_4 \mid v_2 \wedge v_3) \cdot \Pr(v_2 \mid v_3) + \Pr(\neg v_4 \mid \neg v_2 \wedge v_3) \cdot \Pr(\neg v_2 \mid v_3) = \\ & = 0.4 \cdot 0.36 + 0.9 \cdot 0.64 = 0.72 \end{array}$$

□

Complexity of Enhancing Pearl's Algorithm with Loop Cutset Conditioning

To conclude our discussion of the enhancement of Pearl's algorithm for probabilistic inference with the method of loop cutset conditioning, we take a (superficial) look at the algorithm's computational complexity. We recall that analysis of Pearl's algorithm for Bayesian networks comprising a singly connected digraph has revealed that a vertex can compute the probabilities of its values in at most $O(2^n)$ time, where n is the number of vertices in the network's digraph. Propagating a piece of evidence throughout the network may require as many as $O(n^2 \cdot 2^n)$ computations. An analysis of the method of loop cutset conditioning now serves to show that the method adds a factor to the basic algorithm's computational complexity that is *exponential* in the size of the loop cutset employed. Initialising the probabilities for the loop cutset's configurations requires $O(2^l)$ computations, where l is the number of vertices in the loop cutset. Computing the probabilities of a vertex' values now takes $O(2^{n+l})$ time, since these probabilities have to be computed conditioned on every possible configuration of the loop cutset. Propagating a piece of evidence requires at most $O(n^2 \cdot 2^{n+l})$ computations.

Finding a Loop Cutset

In the foregoing, we have argued that the computational complexity of Pearl's algorithm enhanced with loop cutset conditioning, for probabilistic inference with Bayesian networks comprising a multiply connected digraph, is exponential in the size of the loop cutset employed. From a computational point of view, therefore, the best loop cutset to use in practical applications is a loop cutset having the smallest number of vertices. A loop cutset with the smallest number of vertices is called an *optimal* loop cutset; we define the concept of an optimal loop cutset more formally.

Definition 4.2.19 *Let G be an acyclic multiply connected digraph. A loop cutset $\boldsymbol{L}_G$ for G is called* minimal *for G if no proper subset of $\boldsymbol{L}_G$ is a loop cutset for G; the loop cutset $\boldsymbol{L}_G$ is called* optimal *for G if for any other loop cutset $\boldsymbol{L}'_G$ for G we have that $|\boldsymbol{L}_G| \leq |\boldsymbol{L}'_G|$.*

Note that an optimal loop cutset for a multiply connected digraph also is a minimal loop cutset for this digraph; the reverse in general is not true. Also note that an optimal loop cutset for a digraph need not be unique.

Example 4.2.20 We consider again the multiply connected digraph G from Figure 4.9. The sets of vertices $\{V_2, V_6, V_9\}$ and $\{V_3, V_4, V_9\}$ are optimal loop cutsets for G. The set of vertices $\{V_2, V_8, V_{10}, V_{11}\}$ is a minimal loop cutset for G, but not optimal. The set of vertices $\{V_2, V_4, V_8, V_9\}$ is a loop cutset that is neither optimal nor minimal for G. □

Unfortunately, the problem of finding an optimal loop cutset for an acyclic multiply connected digraph is known to be NP-hard [Suermondt & Cooper, 1990]; hence, it is not likely that a generally applicable polynomial-time algorithm for this problem will be found. For the purpose of finding a good loop cutset for a given multiply connected digraph, therefore, generally a *heuristic algorithm* is used. The best known among these algorithms is a heuristic algorithm designed by H.J. Suermondt and G.F. Cooper [Suermondt & Cooper, 1990].

The Suermondt & Cooper heuristic The loop-cutset algorithm of Suermondt and Cooper takes an acyclic multiply connected digraph G for its input and yields a loop cutset $\boldsymbol{L}_G$ for G as its output. The algorithm is composed of two basic phases that are alternated recursively. In the first phase, the algorithm deletes from the digraph G as many vertices as possible that are not included in any loop; such vertices need not be considered as candidates for the loop cutset as upon instantiation they will never block any loops in G. For this purpose, all vertices that have *at most one* incident arc are removed recursively from G, along with their incident arcs. The second phase of the algorithm is entered with a reduced digraph G' in which each vertex has at least two incident arcs. From among this digraph's vertices, a single vertex is selected for inclusion in the loop cutset in the making. The selection of this vertex is based on a *heuristic criterion* that aims at selecting a vertex that upon instantiation blocks as many loops from the reduced digraph G' as possible. This heuristic criterion is based on the idea that a vertex with a larger number of incident arcs in G' is likely to be included in more loops than a vertex with a smaller number of incident arcs; however, a vertex with two incoming arcs on a loop does not serve to block this loop. The heuristic criterion therefore selects a vertex V_i with highest degree having at most one predecessor. The thus selected vertex is included in the loop cutset in the making. The vertex V_i is subsequently removed from the digraph, along with all its incident arcs. Note that in the thus reduced digraph all loops that are blocked by vertex V_i are effectively cut, leaving only loops that are yet unblocked. The algorithm then returns to the first phase. The two phases of the algorithm are thus alternated recursively until no more vertices remain to be considered in the reduced graph. Figure 4.11 summarises the heuristic loop-cutset algorithm of Suermondt and Cooper in pseudo-code.

We illustrate the heuristic loop-cutset algorithm outlined above through an example.

Example 4.2.21 We consider once again the multiply connected digraph G from Figure 4.9 and address the construction of a loop cutset for G using Suermondt and Cooper's heuristic algorithm. The algorithm sets out by initialising the loop cutset in the making with the empty set: $\boldsymbol{L}_G = \varnothing$. The algorithm now notes that the digraph G comprises a single vertex with at most one incident arc — this is the vertex V_1. The algorithm selects this vertex and removes it, along with its incident arc, from G. The thus reduced digraph G_1 is shown in Figure 4.12 (a). From the digraph G_1, the algorithm selects all

```
procedure loop-cutset($G$,$\boldsymbol{L}_G$)
    $\boldsymbol{L}_G := \varnothing$;
    while $\boldsymbol{V}_G \neq \varnothing$ do
        if there is a vertex $V \in \boldsymbol{V}_G$ with $degree(V) \leq 1$
        then select vertex $V$
        else $\boldsymbol{K} := \{V_i \mid V_i \in \boldsymbol{V}_G$ and $V_i$ has $indegree(V_i) \leq 1\}$;
            select from $\boldsymbol{K}$ a vertex $V$ with $degree(V) \geq degree(V_i)$ for all $V_i \in \boldsymbol{K}$;
            $\boldsymbol{L}_G := \boldsymbol{L}_G \cup \{V\}$
        fi;
        $\boldsymbol{V}_G := \boldsymbol{V}_G \setminus \{V\}$;
        $\boldsymbol{A}_G := \boldsymbol{A}_G \cap (\boldsymbol{V}_G \times \boldsymbol{V}_G)$
    od
end
```

Figure 4.11: The Loop-cutset Algorithm of Suermondt and Cooper.

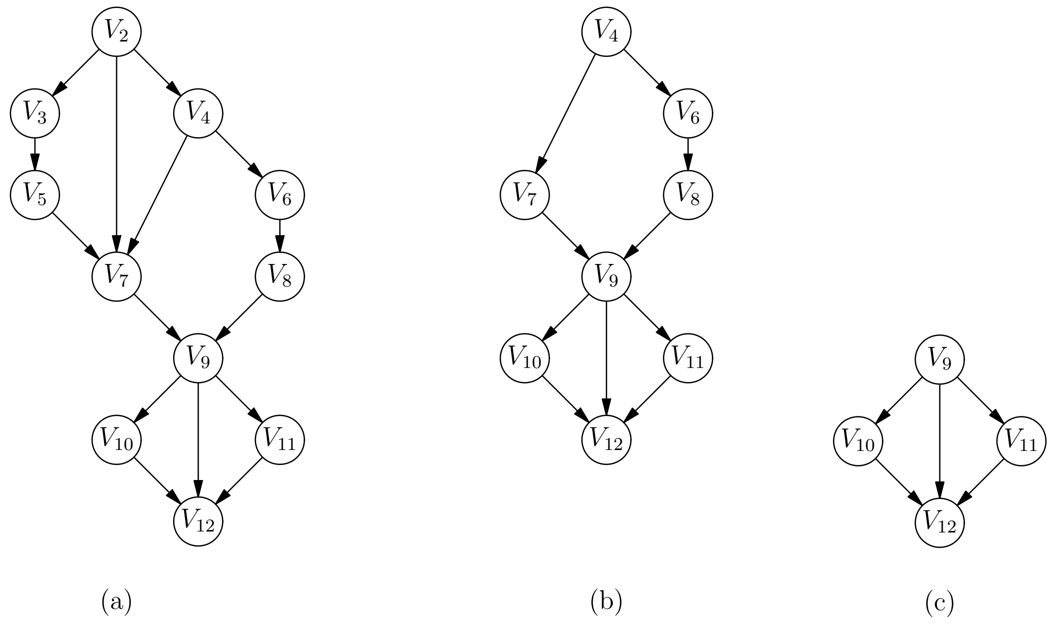


Figure 4.12: Application of the Loop-cutset Algorithm.

vertices with at most one incoming arc. These vertices constitute the set of candidates for inclusion in the loop cutset in the making:

$$\boldsymbol{K_1} = \{V_2, V_3, V_4, V_5, V_6, V_8, V_{10}, V_{11}\}$$

From among these candidates, the algorithm selects a vertex with highest degree. We suppose that the algorithm selects vertex V_2 to include it in the loop cutset in the making:

$$\boldsymbol{L}_G = \{V_2\}$$

The algorithm proceeds by removing from G_1 the selected vertex along with its incident arcs. After subsequently removing all vertices with at most one incident arc, the reduced digraph G_2 shown in Figure 4.12 (b) results. From the digraph G_2, the algorithm once more selects the set of candidates for inclusion in the loop cutset:

$$\boldsymbol{K_2} = \{V_4, V_6, V_7, V_8, V_{10}, V_{11}\}$$

From among these candidates, the algorithm selects a vertex with highest degree. We suppose that it selects vertex V_6. Vertex V_6 is now added to the loop cutset in the making:

$$\boldsymbol{L}_G = \{V_2, V_6\}$$

After removal of vertex V_6 and its incident arcs from G_2, and recursive removal of all vertices with at most one incident arc, the reduced digraph G_3 shown in Figure 4.12 (c) is yielded. From the digraph G_3, the algorithm selects the set of candidates to be

$$\boldsymbol{K_3} = \{V_9, V_{10}, V_{11}\}$$

From among these candidates, the algorithm selects vertex V_9 for inclusion in the loop cutset as it has the highest degree:

$$\boldsymbol{L}_G = \{V_2, V_6, V_9\}$$

After removing vertex V_9 with its incident arcs from G_3, subsequent removal of all vertices with at most one incident arc results in the empty graph. The algorithm halts and yields the loop cutset $\boldsymbol{L}_G = \{V_2, V_6, V_9\}$ for the digraph G. Note that the loop cutset yielded by the algorithm is an optimal loop cutset for G. $\square$

The heuristic loop-cutset algorithm of Suermondt and Cooper is correct in the sense that, for any acyclic multiply connected digraph, the algorithm will halt and yield a set of vertices that is a loop cutset of the digraph [Suermondt & Cooper, 1990]. The heuristic algorithm, however, is not guaranteed to always yield an optimal loop cutset. Experimental results have shown that for randomly generated acyclic multiply connected digraphs, the algorithm finds an optimal loop cutset in approximately 70% of the studied digraphs [Suermondt & Cooper, 1990].

4.2.4 Other Algorithms for Probabilistic Inference

In the previous sections, we have discussed Pearl's basic belief propagation algorithm and its enhancement for general probabilistic inference with a Bayesian network. Pearl's algorithm, however, is not the only algorithm designed for this purpose: various algorithms have been proposed in the course of the preceding decades. In general, an algorithm for exact probabilistic inference with a Bayesian network is built from two basic procedures:

- a procedure for (efficiently) computing probabilities of interest from a Bayesian network, and
- a procedure for processing evidence, that is, for entering evidence into the network and subsequently (efficiently) computing the updated joint probability distribution given the evidence.

Note that in Pearl's algorithm these basic procedures are combined into a single set of computation rules where they cannot easily be distinguished. In most other algorithms, however, these basic procedures are more readily discernible. The algorithms proposed further have two important properties in common: the qualitative part of a Bayesian network is exploited more or less directly as a computational architecture, and after a piece of evidence has been processed in the network, again a Bayesian network results. Note that the latter property allows for recursive processing of evidence.

Join-tree propagation Although all algorithms proposed for probabilistic inference build on the same notion of a Bayesian network, they differ considerably with respect to their underlying concepts. To support this observation, we briefly review another algorithm for probabilistic inference with a Bayesian network, designed by S.L. Lauritzen and D.J. Spiegelhalter [Lauritzen & Spiegelhalter, 1988]. Lauritzen and Spiegelhalter have observed that, after a piece of evidence has become available, updating the joint probability distribution generally entails going against the 'direction' of the initially assessed conditional probabilities. From this observation they have concluded that the *directed* qualitative part of a Bayesian network is not suitable as an architecture for probabilistic inference directly. Their algorithm therefore builds on an *undirected* representation of a joint probability distribution. Prior to probabilistic inference, the original Bayesian network is transformed into a so-called *decomposable* probabilistic network. A decomposable probabilistic network again consists of a qualitative part and a quantitative part. The qualitative part is a decomposable, or *chordal*, graph; the quantitative part is a set of marginal distributions on the vertex sets of the cliques of this graph. The computational architecture for the algorithm derives from the qualitative part of this decomposable probabilistic network in the following sense: the cliques of the decomposable graph are the autonomous objects in the architecture, and the clique intersections give rise to its communication channels. From a decomposable probabilistic network, a probability of interest is computed by further marginalisation of an appropriate marginal distribution. Processing evidence is performed per clique and by message-passing between neighbouring cliques in the architecture.

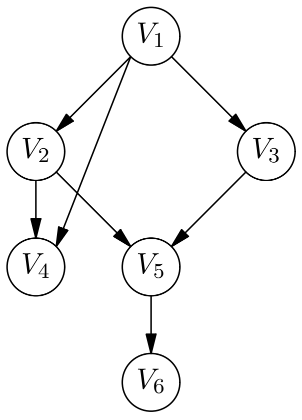
Complexity The various algorithms proposed for probabilistic inference with a Bayesian network also differ with respect to their computational complexity. We note that probabilistic inference with Bayesian networks without any topological restrictions is known to be NP-hard [Cooper, 1990]. All algorithms proposed therefore have an exponential worst-case computational complexity. However, the algorithms can be shown to behave polynomially under certain restrictions. These restrictions concern the topology of a Bayesian network's digraph and differ among the various algorithms. In general, the sparser the digraph of a Bayesian network, the better most algorithms perform. Experience with constructing Bayesian networks for real-life applications so far has indicated that in fact a Bayesian network's digraph tends to be sparse.

Approximate inference For very large networks, or applications that require fast real-time inference, exact inference can be infeasible. In that case, approximate inference can be used, although even approximate inference is NP-hard when we want guarantees on the error bound [Dagum & Luby, 1993]. Approximate inference can be achieved in multiply connected graphs by letting Pearl's algorithm *loop*. Many approximate inference algorithms, however, are typically based upon stochastic simulation, such as logistic or weighted sampling of the network distribution. The more sophisticated MCMC algorithms take an evolutionary approach to sampling by generating new samples through making random changes to previous samples, rather than continuously sampling the network distribution [Korb & Nicholson, 2010]. These algorithms are also suitable for PGMs in general, including those with continuous variables. The same applies to variational approaches that iteratively optimize an approximation to the posterior distribution of interest. Please see [Salmerón et al., 2018] for a review of exact and approximate inference algorithms.

Exercises

Exercise 4.1

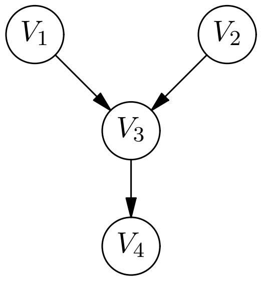
Consider the following acyclic digraph $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$:



Now suppose that the graph G is a (minimal) directed I-map for the independence relation of the joint probability distribution $\Pr$ *on $\boldsymbol{V}_G$. State which (conditional) probabilities from the distribution* $\Pr$ *have to be associated with G to arrive at a Bayesian network comprising G for its qualitative part.*

* Exercise 4.2

Consider the Bayesian network $\mathcal{B} = (G, \Gamma)$ where G is the digraph below and $\Gamma = \{\gamma_{V_i} \mid V_i \in \boldsymbol{V}_G\}$ is defined as:



$\gamma_{V_1}(v_1)$	$= 0.25$	$\gamma_{V_2}(v_2)$	$= 0.5$
$\gamma_{V_3}(v_3 \mid v_1 \wedge v_2)$	$= 0.75$	$\gamma_{V_4}(v_4 \mid v_3)$	$= 0.8$
$\gamma_{V_3}(v_3 \mid \neg v_1 \wedge v_2)$	$= 0.4$	$\gamma_{V_4}(v_4 \mid \neg v_3)$	$= 0$
$\gamma_{V_3}(v_3 \mid v_1 \wedge \neg v_2)$	$= 0.25$		
$\gamma_{V_3}(v_3 \mid \neg v_1 \wedge \neg v_2)$	$= 0.2$		

Let $\Pr$ *be the joint probability distribution defined by $\mathcal{B}$. Compute the following probabilities by exploiting the property stated in Proposition 4.1.3:*

a. $\Pr(v_1 \wedge v_2 \wedge v_3)$*;*

b. $\Pr(v_2 \wedge v_3)$*;*

c. $\Pr(v_1 \mid v_2 \wedge v_3)$*;*

d. $\Pr(v_1 \vee v_2 \vee \neg v_3 \vee v_4)$*.*

* Exercise 4.3

Let $\mathcal{B} = (G, \Gamma)$ be a Bayesian network where $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ is a singly connected graph. Suppose that Pearl's algorithm is employed for probabilistic inference with $\mathcal{B}$. For each vertex $V_i \in \boldsymbol{V}_G$, let $\boldsymbol{V}_i^-$ be as defined in Definition 4.2.11 and let $G^-_{(V_j,V_i)}$, $V_j \in \boldsymbol{\rho}_G(V_i)$, be as defined in Definition 4.2.10.

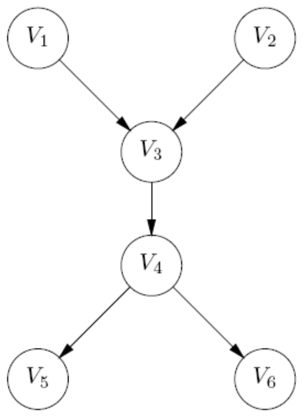
a. Let $\lambda_{V_i}(V_i)$ be the compound diagnostic parameter for a vertex V_i. Show that if $\tilde{c}_{\boldsymbol{V}_i^-} = \mathsf{True}$, then $\lambda_{V_i}(V_i) = 1$.

b. *Let* $\lambda_{V_i}^{V_j}(V_j)$ *be the diagnostic message* V_i *computes for its parent* V_j*. Show that if* $\widetilde{c}_{\boldsymbol{G}^-_{(\boldsymbol{V_j},\boldsymbol{V_i})}} = \mathsf{True}$*, then* $\lambda_{V_i}^{V_j}(V_j) = 1$*. We call this the* Identity property for diagnostic message parameters.

c. *Let* $\pi_{V_i}(V_i)$ *and* $\pi_{V_k}^{V_i}(V_i)$*, for* $V_k \in \boldsymbol{\sigma}_G(V_i)$*, be the compound causal parameter and a causal message computed by vertex* V_i*, respectively. Show that if* $\widetilde{c}_{\boldsymbol{V_i}^-} = \mathsf{True}$*, then* $\pi_{V_k}^{V_i}(V_i) = \pi_{V_i}(V_i)$*. We call this the* Causal parameter equivalence *property.*

* Exercise 4.4

Consider the Bayesian network $\mathcal{B} = (G, \Gamma)$ *where* $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ *is the digraph below and* $\Gamma = \{\gamma_{V_i} \mid V_i \in \boldsymbol{V}_G\}$ *is defined as*



$\gamma_{V_1}(v_1) = 0.5$ $\gamma_{V_2}(v_2) = 0.4$

$\gamma_{V_3}(v_3 \mid v_1 \wedge v_2) = 0.9$
$\gamma_{V_3}(v_3 \mid \neg v_1 \wedge v_2) = 0.6$
$\gamma_{V_3}(v_3 \mid v_1 \wedge \neg v_2) = 0.2$
$\gamma_{V_3}(v_3 \mid \neg v_1 \wedge \neg v_2) = 0.8$

$\gamma_{V_4}(v_4 \mid v_3) = 0.5$
$\gamma_{V_4}(v_4 \mid \neg v_3) = 0.65$

$\gamma_{V_5}(v_5 \mid v_4) = 0.25$
$\gamma_{V_5}(v_5 \mid \neg v_4) = 0.3$

$\gamma_{V_6}(v_6 \mid v_4) = 0.6$
$\gamma_{V_6}(v_6 \mid \neg v_4) = 0.1$

Let Pr *be the joint probability distribution defined by the Bayesian network* $\mathcal{B}$*. Suppose that Pearl's algorithm is employed for probabilistic inference with* $\mathcal{B}$*.*

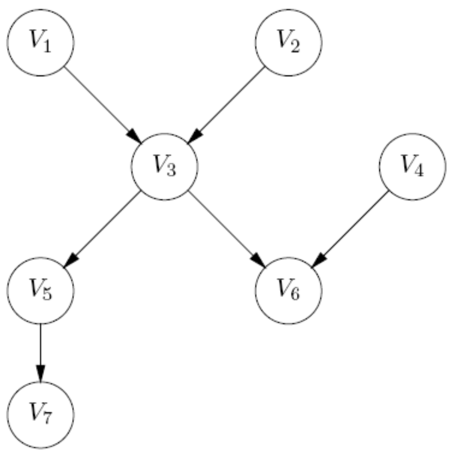
a. *Compute the probabilities* $\Pr(v_4)$ *and* $\Pr(\neg v_4)$ *for vertex* V_4 *from the Bayesian network* $\mathcal{B}$*.*

b. *Suppose that the evidence* $V_6 = true$ *is observed and entered into the network. Compute the updated probabilities* $\Pr^{v_6}(v_4)$ *and* $\Pr^{v_6}(\neg v_4)$ *for vertex* V_4*.*

c. *Now, suppose that after processing the evidence* $V_6 = true$*, the evidence* $V_2 = false$ *is obtained. For each vertex in the network, examine whether or not this new evidence can influence the probabilities of its values.*

* Exercise 4.5

Consider the Bayesian network $\mathcal{B} = (G, \Gamma)$ *where* $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ *is the digraph below and* $\Gamma = \{\gamma_{V_i} \mid V_i \in \boldsymbol{V}_G\}$ *is defined as*



$\gamma_{V_1}(v_1) = 0.8$ $\gamma_{V_2}(v_2) = 0.5$

$\gamma_{V_4}(v_4) = 0.3$

$\gamma_{V_3}(v_3 \mid v_1 \wedge v_2) = 0.2$
$\gamma_{V_3}(v_3 \mid \neg v_1 \wedge v_2) = 0.6$
$\gamma_{V_3}(v_3 \mid v_1 \wedge \neg v_2) = 0.5$
$\gamma_{V_3}(v_3 \mid \neg v_1 \wedge \neg v_2) = 0.1$

$\gamma_{V_6}(v_6 \mid v_3 \wedge v_4) = 0.3$
$\gamma_{V_6}(v_6 \mid \neg v_3 \wedge v_4) = 0.8$
$\gamma_{V_6}(v_6 \mid v_3 \wedge \neg v_4) = 0.2$
$\gamma_{V_6}(v_6 \mid \neg v_3 \wedge \neg v_4) = 0.1$

$\gamma_{V_5}(v_5 \mid v_3) = 0.6$
$\gamma_{V_5}(v_5 \mid \neg v_3) = 0.4$

$\gamma_{V_7}(v_7 \mid v_5) = 0.9$
$\gamma_{V_7}(v_7 \mid \neg v_5) = 0.8$

Let Pr *be the joint probability distribution defined by the Bayesian network $\mathcal{B}$. Suppose that Pearl's algorithm is employed for probabilistic inference with $\mathcal{B}$.*

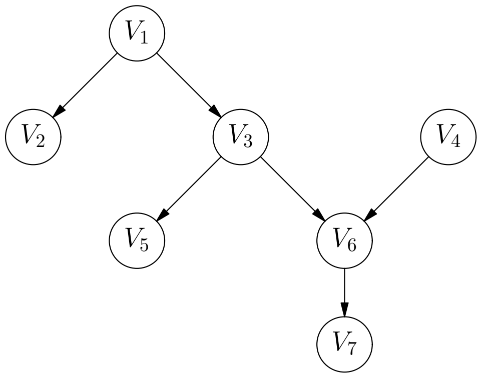
a. *Compute the probabilities* $\Pr(v_5)$ *and* $\Pr(\neg v_5)$ *for vertex* V_5 *from the Bayesian network* $\mathcal{B}$.

b. *Suppose that the evidence* $V_3 = true$ *is observed and entered into the Bayesian network* $\mathcal{B}$. *Compute the updated probabilities* $\Pr^{v_3}(v_5)$ *and* $\Pr^{v_3}(\neg v_5)$ *for vertex* V_5 *from the network.*

c. *For each vertex in the network, examine whether or not the evidence* $V_3 = true$ *may influence the probabilities of this vertex' values.*

* Exercise 4.6

Consider the Bayesian network $\mathcal{B} = (G, \Gamma)$ *where* $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ *is the digraph below and* $\Gamma = \{\gamma_{V_i} \mid V_i \in \boldsymbol{V}_G\}$ *is defined as*



$$\gamma_{V_1}(v_1) = 0.5$$

$$\gamma_{V_2}(v_2 \mid v_1) = 0.35 \qquad \gamma_{V_2}(v_2 \mid \neg v_1) = 0.1$$

$$\gamma_{V_3}(v_3 \mid v_1) = 0.5 \qquad \gamma_{V_3}(v_3 \mid \neg v_1) = 0$$

$$\gamma_{V_4}(v_4) = 0.5$$

$$\gamma_{V_5}(v_5 \mid v_3) = 0.2 \qquad \gamma_{V_5}(v_5 \mid \neg v_3) = 1$$

$$\gamma_{V_6}(v_6 \mid v_3 \wedge v_4) = 0.75 \qquad \gamma_{V_6}(v_6 \mid \neg v_3 \wedge v_4) = 0.3$$
$$\gamma_{V_6}(v_6 \mid v_3 \wedge \neg v_4) = 0.25 \qquad \gamma_{V_6}(v_6 \mid \neg v_3 \wedge \neg v_4) = 0.2$$

$$\gamma_{V_7}(v_7 \mid v_6) = 0.8 \qquad \gamma_{V_7}(v_7 \mid \neg v_6) = 0$$

Let Pr *be the joint probability distribution defined by the Bayesian network $\mathcal{B}$. Suppose that Pearl's algorithm is employed for probabilistic inference with $\mathcal{B}$.*

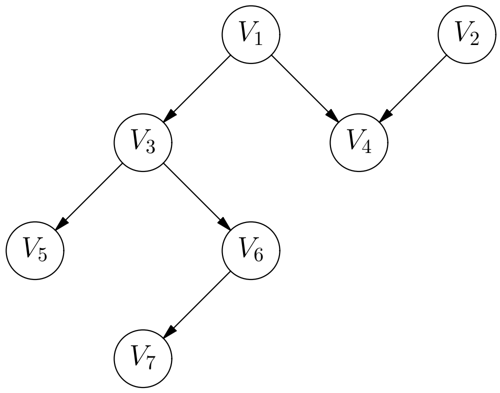
a. *Suppose that the evidence* $V_7 = true$ *has been entered into the network* $\mathcal{B}$. *Compute the probabilities* $\Pr^{v_7}(v_1)$ *and* $\Pr^{v_7}(\neg v_1)$ *for vertex* V_1 *from the Bayesian network* $\mathcal{B}$.

b. *Now suppose that the evidence* $V_4 = true$ *is entered into the network as well. Does the probability* $\Pr^{v_7,v_4}(v_1)$ *differ from the probability* $\Pr^{v_7}(v_1)$ *?*

c. *Suppose that, for the application for which the Bayesian network* $\mathcal{B}$ *has been developed, the probability* $\Pr^{v_7,v_4}(v_1 \wedge \neg v_2)$ *needs to be computed. Explain how this probability can be computed from the network efficiently.*

* Exercise 4.7

Consider the Bayesian network $\mathcal{B} = (G, \Gamma)$ where $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ is the digraph below and $\Gamma = \{\gamma_{V_i} \mid V_i \in \boldsymbol{V}_G\}$ is defined as:



$\gamma_{V_1}(v_1) = 0.05$

$\gamma_{V_2}(v_2) = 0.3$

$\gamma_{V_3}(v_3 \mid v_1) = 0.75$
$\gamma_{V_3}(v_3 \mid \neg v_1) = 0.4$

$\gamma_{V_4}(v_4 \mid v_1 \wedge v_2) = 0.1$
$\gamma_{V_4}(v_4 \mid \neg v_1 \wedge v_2) = 0.25$
$\gamma_{V_4}(v_4 \mid v_1 \wedge \neg v_2) = 0.8$
$\gamma_{V_4}(v_4 \mid \neg v_1 \wedge \neg v_2) = 0.95$

$\gamma_{V_5}(v_5 \mid v_3) = 0.35$
$\gamma_{V_5}(v_5 \mid \neg v_3) = 0$

$\gamma_{V_6}(v_6 \mid v_3) = 0.8$
$\gamma_{V_6}(v_6 \mid \neg v_3) = 0.6$

$\gamma_{V_7}(v_7 \mid v_6) = 0.5$
$\gamma_{V_7}(v_7 \mid \neg v_6) = 0.45$

Let Pr *be the joint probability distribution defined by the Bayesian network $\mathcal{B}$. Suppose that Pearl's algorithm is employed for probabilistic inference with $\mathcal{B}$.*

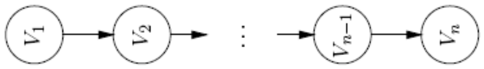
a. *Suppose that the evidence V_1 = true and V_6 = false has been entered into the Bayesian network $\mathcal{B}$. Compute the probabilities $\Pr^{v_1, \neg v_6}(v_3)$ and $\Pr^{v_1, \neg v_6}(\neg v_3)$ for vertex V_3 from $\mathcal{B}$.*

b. *Suppose that, for the application for which the Bayesian network $\mathcal{B}$ has been developed, the probability $\Pr^{v_1, \neg v_6}(v_2 \wedge v_5)$ needs to be computed. Explain how this probability can be computed from the network efficiently.*

c. *Now suppose that, for the application for which the Bayesian network has been developed, the probability $\Pr^{v_1, \neg v_6}(v_2 \vee v_4)$ needs to be computed. Explain how this probability can be computed from the network efficiently.*

* Exercise 4.8

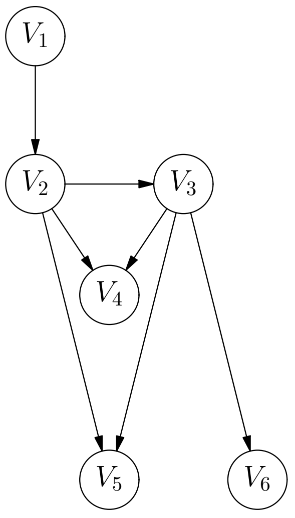
Let $\mathcal{B} = (G, \Gamma)$ be a Bayesian network where $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ with $\boldsymbol{V}_G = \{V_1, \ldots, V_n\}$, $n \geq 1$, and $\boldsymbol{A}_G = \{(V_i, V_{i+1}) \mid i = 1, \ldots, n-1\}$ is the following digraph:



Let Pr *be the joint probability distribution defined by the Bayesian network $\mathcal{B}$. Furthermore, let $\mathcal{B}' = (G, \Gamma')$ be the Bayesian network that is obtained from $\mathcal{B}$ by substituting for a vertex V_k, $k > 1$, the function value $\gamma_{V_k}(v_k \mid v_{k-1})$ by $\gamma_{V_k}(v_k \mid v_{k-1}) + \epsilon$, for some small $\epsilon > 0$. Let $\Pr'$ be the joint probability distribution defined by $\mathcal{B}'$. Show that $|\Pr'(V_i) - \Pr(V_i)| \leq \epsilon$, for all $V_i \in \boldsymbol{V}_G$, $i = 1, \ldots, n$.*

Exercise 4.9

Consider the Bayesian network $\mathcal{B} = (G, \Gamma)$ where $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ is the digraph below and $\Gamma = \{\gamma_{V_i} \mid V_i \in \boldsymbol{V}_G\}$ is defined as



$$\gamma_{V_1}(v_1) = 0.5$$

$$\begin{array}{ll}
\gamma_{V_2}(v_2 \mid v_1) = 0.9 & \gamma_{V_3}(v_3 \mid v_2) = 0.5 \\
\gamma_{V_2}(v_2 \mid \neg v_1) = 0.7 & \gamma_{V_3}(v_3 \mid \neg v_2) = 0.8
\end{array}$$

$$\begin{array}{ll}
\gamma_{V_4}(v_4 \mid v_2 \wedge v_3) = 0.1 & \gamma_{V_5}(v_5 \mid v_2 \wedge v_3) = 0.3 \\
\gamma_{V_4}(v_4 \mid \neg v_2 \wedge v_3) = 0.25 & \gamma_{V_5}(v_5 \mid \neg v_2 \wedge v_3) = 0.7 \\
\gamma_{V_4}(v_4 \mid v_2 \wedge \neg v_3) = 0.6 & \gamma_{V_5}(v_5 \mid v_2 \wedge \neg v_3) = 0.2 \\
\gamma_{V_4}(v_4 \mid \neg v_2 \wedge \neg v_3) = 0.75 & \gamma_{V_5}(v_5 \mid \neg v_2 \wedge \neg v_3) = 0.7
\end{array}$$

$$\begin{array}{l}
\gamma_{V_6}(v_6 \mid v_3) = 0.45 \\
\gamma_{V_6}(v_6 \mid \neg v_3) = 0.65
\end{array}$$

Let Pr *be the joint probability distribution defined by the Bayesian network $\mathcal{B}$.*

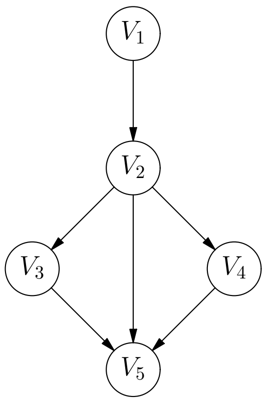
* a. *Compute a loop cutset for the digraph G by employing the heuristic algorithm of Suermondt and Cooper.*

* b. *Compute the probabilities* $\Pr(v_5)$ *and* $\Pr(\neg v_5)$ *for vertex V_5 from the Bayesian network $\mathcal{B}$ by Pearl's algorithm with loop cutset conditioning.*

c. *Suppose that the evidence $V_3 = false$ is observed and entered into the network. Furthermore, suppose that after entering $V_3 = false$ into the network, the additional evidence $V_1 = true$ is observed. For each vertex in the network, examine whether or not this new evidence can influence the probabilities of its values.*

* Exercise 4.10

Consider the Bayesian network $\mathcal{B} = (G, \Gamma)$ where $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ is the digraph below and $\Gamma = \{\gamma_{V_i} \mid V_i \in \boldsymbol{V}_G\}$ is defined as:



$$\begin{array}{ll}
\gamma_{V_1}(v_1) = 0.9 & \gamma_{V_5}(v_5 \mid v_2 \wedge v_3 \wedge v_4) = 0.2 \\
 & \gamma_{V_5}(v_5 \mid \neg v_2 \wedge v_3 \wedge v_4) = 0.25 \\
\gamma_{V_2}(v_2 \mid v_1) = 0.125 & \gamma_{V_5}(v_5 \mid v_2 \wedge \neg v_3 \wedge v_4) = 0.7 \\
\gamma_{V_2}(v_2 \mid \neg v_1) = 0.875 & \gamma_{V_5}(v_5 \mid v_2 \wedge v_3 \wedge \neg v_4) = 0.6 \\
 & \gamma_{V_5}(v_5 \mid \neg v_2 \wedge \neg v_3 \wedge v_4) = 0.55 \\
\gamma_{V_3}(v_3 \mid v_2) = 0.7 & \gamma_{V_5}(v_5 \mid \neg v_2 \wedge v_3 \wedge \neg v_4) = 0.8 \\
\gamma_{V_3}(v_3 \mid \neg v_2) = 0.2 & \gamma_{V_5}(v_5 \mid v_2 \wedge \neg v_3 \wedge \neg v_4) = 0.3 \\
 & \gamma_{V_5}(v_5 \mid \neg v_2 \wedge \neg v_3 \wedge \neg v_4) = 0.15 \\
\gamma_{V_4}(v_4 \mid v_2) = 0.35 & \\
\gamma_{V_4}(v_4 \mid \neg v_2) = 0.1 &
\end{array}$$

Let Pr *be the joint probability distribution defined by the Bayesian network $\mathcal{B}$.*

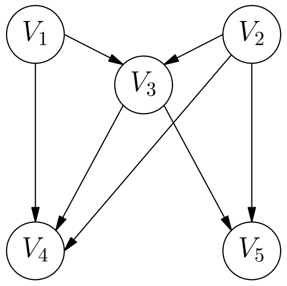
a. *Compute a loop cutset for the digraph G by employing the heuristic algorithm of Suermondt and Cooper.*

b. *Compute the probabilities* $\Pr(v_3)$ *and* $\Pr(\neg v_3)$ *for vertex V_3 from the Bayesian network $\mathcal{B}$ by Pearl's algorithm with loop cutset conditioning.*

c. *Suppose that the evidence $V_5 = true$ is entered into the network. Explain how the probabilities $\Pr^{v_5}(v_3)$ and $\Pr^{v_5}(\neg v_3)$ are computed from the network.*

* Exercise 4.11

a. *Consider a Bayesian network $\mathcal{B} = (G, \Gamma)$ where G is the following digraph:*



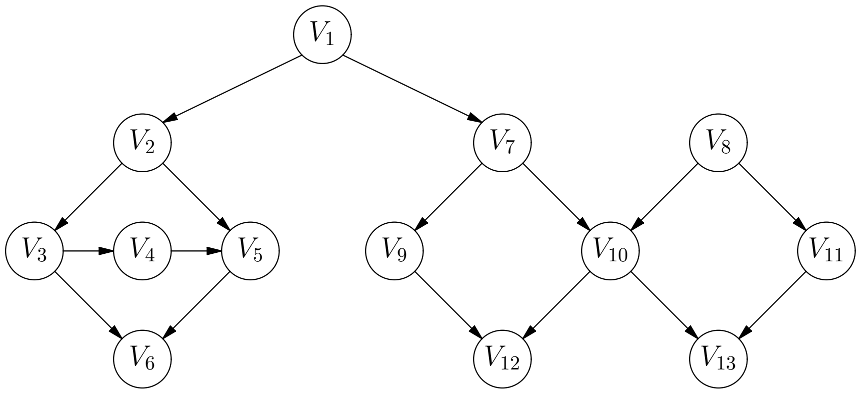
Compute a loop cutset for G by employing the heuristic algorithm of Suermondt and Cooper.

b. *The heuristic algorithm of Suermondt and Cooper aims at computing for a multiply connected digraph an* optimal loop cutset*. Now consider a Bayesian network in which the variables are not necessarily binary but may take a value from an arbitrarily large finite set of values. Illustrate by means of an example that for such a Bayesian network a non-optimal loop cutset may be a better choice for the purpose of loop cutset conditioning than an optimal loop cutset.*

c. *In the heuristic algorithm of Suermondt and Cooper the number of values for the various variables of a Bayesian network are not taken into consideration. Modify the algorithm in such a way that the number of values is taken into consideration to arrive at a 'good' loop cutset.*

* Exercise 4.12

a. *Consider the following digraph G:*



Compute a loop cutset for G by employing the heuristic algorithm of Suermondt and Cooper.

b. *Let $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ be an acyclic multiply connected digraph and let $\boldsymbol{C} \subseteq \boldsymbol{V}_G$ be a set of vertices in G. Furthermore, let G' be the digraph that is obtained from G by deleting for each vertex from $\boldsymbol{C}$ its outgoing arcs, that is, $G' = (\boldsymbol{V}_G, \boldsymbol{A}'_G)$ with $\boldsymbol{A}'_G = \boldsymbol{A}_G \setminus \{(V_i, V_j) \mid V_i \in \boldsymbol{C}\}$. Show that the set $\boldsymbol{C}$ is a loop cutset of G if and only if G' is a singly connected digraph.*

c. *Upon employing the heuristic algorithm of Suermondt and Cooper for computing a loop cutset for a multiply connected digraph G, a non-minimal loop cutset $\boldsymbol{L}_G$ may be yielded. Give a sketch of an algorithm that takes for its input the non-minimal loop cutset $\boldsymbol{L}_G$ and yields a minimal loop cutset $\boldsymbol{L}'_G$ of G.*

* Exercise 4.13

For probabilistic inference with a Bayesian network, sometimes Pearl's algorithm is supplemented with a method called evidence absorption. *The basic idea of this method is to dynamically modify a Bayesian network after evidence has been entered so as to reflect as many of the new independences occasioned by the evidence as possible. Now consider a Bayesian network $\mathcal{B} = (G, \Gamma)$ and suppose that the evidence $V_i = true$ is entered for some vertex $V_i \in \boldsymbol{V}_G$. Evidence absorption now amounts to modifying the digraph G of $\mathcal{B}$ by deleting all outgoing arcs from vertex V_i; with the thus modified digraph G' a new set Γ' of assessment functions is constructed such that the new Bayesian network $\mathcal{B}' = (G', \Gamma')$ reflects the updated joint probability distribution.*

a. *Show that the digraphs G and G' portray the same set of independences given V_i.*

b. *Show by means of an example that the digraphs G and G' do not represent the same set of independences.*

c. *Explain why repeated application of the method of evidence absorption reduces the average-case computational complexity of reasoning with a Bayesian network.*

*Exercise 4.14

In addition to the standard inference query (computing $\Pr(c_H \mid c_{\boldsymbol{E}})$), in practice we are often interested in two additional queries: finding the so-called MAP (maximum a-posteriori probability) assignment or the MPE (most probable explanation) for evidence $c_{\boldsymbol{E}}$:

$$\mathrm{MPE}(c_{\boldsymbol{E}}) = \arg\max_{c_{\boldsymbol{Y}}} \Pr(c_{\boldsymbol{Y}} \mid c_{\boldsymbol{E}}) = \arg\max_{c_{\boldsymbol{Y}}} \Pr(c_{\boldsymbol{Y}} \wedge c_{\boldsymbol{E}}), \ \textit{where } \boldsymbol{Y} = \boldsymbol{V}_G \setminus \boldsymbol{E}$$

$$\text{and for } \boldsymbol{X} \subset \boldsymbol{Y}: \ \mathrm{MAP}(\boldsymbol{X}, c_{\boldsymbol{E}}) = \arg\max_{c_{\boldsymbol{X}}} \Pr(c_{\boldsymbol{X}} \mid c_{\boldsymbol{E}}) = \arg\max_{c_{\boldsymbol{X}}} \Pr(c_{\boldsymbol{X}} \wedge c_{\boldsymbol{E}})$$

a. *Consider the example network in Figure 4.1.*
Let $c_{\boldsymbol{E}} = v_1$ be the observed evidence. Compute MPE($c_{\boldsymbol{E}}$) and MAP($\{V_3, V_4\}, c_{\boldsymbol{E}}$).

b. *Which of the following statements comparing MPE and MAP assignments is correct:*

   A. *computing an MPE is generally more efficient, since it involves all network variables*

   B. *computing a MAP assignment is generally more efficient, since it involves only a subset of all network variables*

   C. *computing MAP and MPE is equally (in)efficient:*
   let $\boldsymbol{Y} = \boldsymbol{X} \cup \boldsymbol{W}$ and $\mathrm{MPE}(c_{\boldsymbol{E}}) = c^*_{\boldsymbol{Y}} = c'_{\boldsymbol{X}} \wedge c'_{\boldsymbol{W}}$ for some configurations $c'_{\boldsymbol{X}}$ and $c'_{\boldsymbol{W}}$; then $c^*_{\boldsymbol{X}} = \mathrm{MAP}(\boldsymbol{X}, c_{\boldsymbol{E}})$ follows immediately from $c^*_{\boldsymbol{Y}}$ since $c^*_{\boldsymbol{X}} = c'_{\boldsymbol{X}}$

   D. *none of the above*

   Clearly explain your answer.

Part II

Towards BN applications

Chapter 5

Building a Bayesian Network

The Bayesian network framework generally is used for applying probability theory for reasoning with uncertainty in knowledge-based systems. For employing the framework for a real-life domain of application, relevant knowledge of the domain at hand is represented in the Bayesian network formalism; the basic algorithms of the framework are taken as building blocks for shaping the system's intelligent problem-solving behaviour. In this chapter we address the task of *building* a Bayesian network for a domain of application; shaping intelligent problem-solving behaviour is one of the topics of the next chapter.

Building a Bayesian network for a domain of application involves various different tasks. The first of these tasks is to identify the *variables* that are of importance in the domain at hand, along with their possible values. Once the important domain variables have been identified, the qualitative part of the Bayesian network in the making is constructed: the second task in building a Bayesian network therefore is to identify the *independences* among the variables discerned and to express these in an acyclic digraph. After the qualitative part of the network has been established, the *probability assessment functions* are defined: the last task in building a Bayesian network is to estimate the (conditional) probabilities that are required to constitute the network's quantitative part. The three tasks are summarised in Figure 5.1. The various tasks in building a Bayesian network are, in

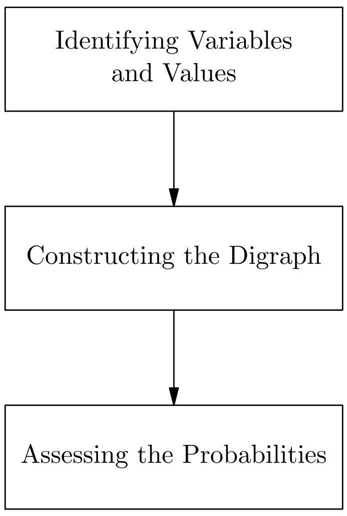


Figure 5.1: The tasks in building a Bayesian network.

principle, performed one after the other. Building a network, however, often requires a careful *trade-off* between the desire for a large and rich model to obtain accurate results on the one hand, and the costs of construction and maintenance, and the complexity of probabilistic inference on the other hand. In practice, therefore, building a Bayesian network is a cyclic process that iterates over these tasks until a network results that is deemed satisfactory for the domain at hand. We would like to note that, although the Bayesian network framework has been around for some time, methodologies for building real-life Bayesian networks do not yet abound. Building a Bayesian network is an engineering task and it therefore makes sense to adopt common practices from(software) engineering, especially for building large models [Laskey & Mahoney, 2000]. Recently, various Bayesian networks have been built for environmental applications. This led to the publication of a paper with guidelines to developing and evaluating Bayesian network models of environmental systems [Chen & Pollino, 2012]; the paper summarises most of the steps detailed in this section and includes lots of useful references.

In the following sections, we address the tasks involved in building a Bayesian network separately. In Section 5.1 we address the identification of the important domain variables and values. In Section 5.2 we discuss the construction of the qualitative part of a Bayesian network in the making. Section 5.3 focuses on the assessment of the probabilities required for the network's quantitative part.

5.1 Identifying Variables and Values

The first task in building a Bayesian network for a domain of application is to identify the variables that are of importance in the domain, along with their possible values. Identifying the important domain variables is typically performed with the help of one or more experts. Since a Bayesian network, as any model, necessarily is a simplification of reality, well-founded decisions have to be taken as to which variables should be included in the network and which may be omitted. We would like to note that this task is not reserved for building Bayesian networks, but instead is quite common in engineering knowledge-based systems. A knowledge engineer can therefore make use of the various elicitation techniques designed for engineering knowledge-based systems in general. For example, the use of ontologies was suggested to help structure the domain, its variables and the different types of relations that exist between the variables [Helsper & Van der Gaag, 2002]. Such ontologies have proved useful tools upon constructing a Bayesian network's digraph, for distilling the variables and their interrelationships that are important for the problem at hand. We will not further elaborate on this task here and confine ourselves to emphasising that it needs to be performed with care since the domain variables that are modelled in the Bayesian network demarcate the scope of the resulting system.

Once the important domain variables have been identified, each of them has to be expressed as a *random variable* to allow for inclusion in the Bayesian network in the making. We recall that a random variable is characterised by its values being *mutually exclusive* and *collectively exhaustive.* To allow for inclusion in a Bayesian network, a random variable has to take its value from a *finite* set of discrete values. Only if the set of values of a domain variable satisfies these properties, can the variable be included in the network as it is. Otherwise, the domain variable is modelled in terms of one or more random variables.

For modelling domain variables for inclusion in a Bayesian network, we distinguish between the following types of variable:

- a *single-valued* domain variable that takes a value from a *finite* set of (discrete) values;
- a *single-valued* domain variable that takes a value from an *infinite* set of values;
- a *multi-valued* domain variable.

We begin by considering a *single-valued* domain variable that takes its value from a *finite* set of discrete values. The values of this variable are mutually exclusive by definition. Moreover, the variable's values are easily rendered collectively exhaustive. A single-valued domain variable can therefore be expressed straightforwardly as a random variable in the Bayesian network in the making.

For a *single-valued* domain variable that takes its value from an *infinite* set of values, we equally have that this variable's values are mutually exclusive and are easily rendered collectively exhaustive. However, its set of possible values being infinite prohibits including the domain variable as it is in the Bayesian network in the making. To allow for inclusion in the network, the set of values is *discretised.* To this end, the variable's possible values are clustered in a finite number of mutually exclusive and collectively exhaustive clusters of values; these clusters then are taken as the values of a *new* single-valued variable that is included in the Bayesian network in the making.

Example 5.1.1 Suppose that, in a medical domain of application, a patient's *temperature* has been identified as an important domain variable. This domain variable is single-valued, taking its value from the (in essence) infinite range of values between, say, 35 °C and 41 °C. This range of values can be discretised in, for example, the five intervals [35;36), [36;37), [37;38), [38;40), and [40;41]. Note that these intervals can be looked upon as values that are mutually exclusive and collectively exhaustive. For inclusion in the Bayesian network in the making, the domain variable is now expressed as a random variable that takes its value from among the five values mentioned above. □

In general, a domain variable's set of values admits numerous discretisations. The discretisation best chosen is highly dependent on the purpose for which the Bayesian network is being developed and has to be decided upon in consultation with one or more domain experts.

A domain variable taking *multiple* values from a finite set of discrete values cannot be expressed directly as a random variable since its values are not mutually exclusive. To allow for inclusion in the Bayesian network in the making, such a domain variable needs to be re-modelled so as to satisfy the properties of a random variables. Re-modelling may be done by redefining the variable's set of values in such a way that the new values are mutually exclusive and collectively exhaustive.

Example 5.1.2 Suppose that, in a medical domain of application, a patient's *blood count* has been identified as an important domain variable. This domain variable is multi-valued, taking its values from among the set of values {*normal*, *lymphocytosis*, *lymphocytopenia*, *leucocytosis*, *leucocytopenia*}. The values *lymphocytosis* and *lymphocytopenia* refer to the level of lymphocytes in a patient's blood sample and the values *leucocytosis* and *leucocytopenia* refer to the leucocyte level; these values are not mutually exclusive. Domain knowledge now indicates that a patient's *blood count* cannot take any arbitrary combination of the values mentioned above. In fact, only nine combinations of values are meaningful – these are {*normal*}, {*lymphocytosis*}, {*lymphocytopenia*}, {*leucocytosis*}, {*leucocytopenia*}, {*lymphocytosis, leucocytosis*},
{*lymphocytosis, leucocytopenia*}, {*lymphocytopenia, leucocytosis*}, and {*lymphocytopenia, leucocytopenia*}. These combinations are now taken as the mutually exclusive values of a new random variable, modelling a patient's blood count, to be included in the Bayesian network in the making. □

Multi-valuedness of a domain variable often arises from the variable being *compound*, that is, implicitly built from several other domain variables. Instead of redefining its set of values, a compound variable may be modelled by decomposing it into the domain variables it is built from and subsequently modelling these separate variables in the Bayesian network in the making.

Example 5.1.3 We consider once more the domain variable expressing a patient's *blood count* as introduced in the previous example. We observe that this domain variable is a compound variable built from two variables. One of these variables expresses the level of lymphocytes in a patient's blood sample, taking its value from the set of values {*normal*, *lymphocytosis*, *lymphocytopenia*}; the other variable captures the level of leucocytes, taking its value from the set of values {*normal*, *leucocytosis*, *leucocytopenia*}. Note that these two variables are random variables, that can be included directly in the Bayesian network in the making. The originally multi-valued domain variable is now decomposed into these variables. □

Decomposing a compound variable requires considerable knowledge of the domain at hand and therefore is best performed with the help of a domain expert.

After all important domain variables have been expressed as random variables for inclusion in the Bayesian network in the making, the meanings of these random variables have to be properly documented. While each random variable has a unique meaning in the context of the Bayesian network at hand, its meaning may very well differ from the meanings associated with variables with the same name in other models or in the literature in the domain. The task of documenting the meanings of the random variables modelled in the Bayesian network therefore is crucial to avoid any ambiguity in future reference. This task is not typical for building Bayesian networks and has in fact been recognised as vital in knowledge-engineering practice in general.

5.2 Constructing the Digraph

Once the domain variables of importance have been identified and expressed as random variables, the construction of the qualitative part of the Bayesian network in the making can commence. In principle, for constructing the network's qualitative part, the independence relation of the joint probability distribution on the variables discerned has to be identified and represented in an acyclic digraph. In practice, however, the digraph typically is conceived without explicitly identifying independences: it is constructed directly, either by hand or by learning from data.

5.2.1 Constructing the Digraph by Hand

For most domains of application, the qualitative part of a Bayesian network in the making is largely *handcrafted*. For this purpose, one or more domain experts are interviewed. In the interviews, the qualitative relationships between the variables discerned are elicited, using the concept of *causality* as a heuristic guiding principle. Typical questions asked during the interviews are "What could cause this effect ?", "What manifestations could this cause have ?" [Henrion, 1989]. The elicited causal relationships among the variables are expressed in graphical terms by taking the direction of causality for directing the arcs between related variables. The resulting graphical representation can then be taken as a basis for further refinement.

Using the concept of causality as a guiding principle during the interviews with domain experts may meet with some difficulties [Jensen & Nielsen, 2007, Ch. 3]. For example, not every influential relationship among variables can be interpreted as causal. If a non-causal influential relationship comes to the fore during an interview, a more elaborate analysis of the independences involved is required before it can be expressed in graphical terms. A tool that can support this analysis by visually and verbally explaining the (in)dependences captured in the graphical structure is *Matilda* [Boneh *et al.*, 2006]. Another problem that may arise from building on causality concerns the creation of cycles in the graphical representation. As the digraph of a Bayesian network has to be acyclic, any cycle that has resulted from the interviews needs to be removed. There are several ways to deal with cycles [Peek & Ottenkamp, 1997], the simplest being the deletion of a single arc from a cycle to be removed. To conclude, the concept of causality may leave room for multiple interpretations [Pearl, 2009].

Despite the fact that building on the concept of causality does not suffice in itself for constructing the digraph of a Bayesian network in the making, it brings the advantage that domain experts are allowed to express their knowledge in either the *causal* or *diagnostic* direction. Since they are allowed to express their knowledge in a form they feel comfortable with, the experts' statements tend to be quite robust. Some experiences with the issues discussed above in building a Bayesian network for a real-life application are described in [Van der Gaag & Helsper, 2002].

We would like to note that the task of eliciting relationships among variables from domain experts is not reserved for building Bayesian networks: the elicitation task has parallels with engineering knowledge-based systems in general for which several methodologies have been developed [Boose & Gaines, 1988]. Recent research has focussed on the use of knowledge expressed in ontologies, stories, Wigmore charts and structured arguments to inform the construction of Bayesian networks[Helsper & Van der Gaag, 2002, Vlek *et al.*, 2014, Kadane & Schum, 1996, Wieten *et al.*, 2019].

5.2.2 Learning the Digraph from Data

In more and more domains of application, data is collected and maintained over numerous years of every-day problem solving. Such a data collection usually contains highly valuable information about the relationships among the variables discerned, be it implicitly. If a comprehensive data set is available in the domain for which a Bayesian network is developed, the construction of the network's qualitative part may be performed *automatically*: the basic idea of *learning* the qualitative part from data is to distil information from the data set and exploit it for constructing a digraph.

The data set

For learning the qualitative part of a Bayesian network, a data set with information from the domain at hand is required.

We formally define the concept of a database.

Definition 5.2.1 *Let $\boldsymbol{V}$ be a set of variables. A* data set *$\boldsymbol{D}$ over $\boldsymbol{V}$ is a multi-set of configurations of $\boldsymbol{V}$. An element $c_{\boldsymbol{V}}$ of $\boldsymbol{D}$ is called a* case.

To be suitable for learning purposes, a data set has to satisfy various properties. First of all, the data comprised in the data set must have been collected very *carefully.* Biases that are introduced in the data set as a result of the data collection strategies used will have impact on the topology of the resulting digraph, yet may not be desirable for the purpose for which the Bayesian network is being developed; unfortunately, biases are not easily detected in a once constructed network. Also, to be suitable for the purpose of learning a Bayesian network's digraph, the variables and associated values that occur in the data set should *match* the variables and values that are to be modelled in the network, or should at least admit easy translation into these variables and values. Moreover, the data set should comprise enough data to allow for *reliable* identification of probabilistic relationships among the variables discerned.

In addition to the general prerequisites outlined before, a data set should satisfy several properties that are implicitly assumed by most algorithms for learning a Bayesian network's digraph from data. A property that is commonly assumed is that each case in the data set specifies a value for every variable discerned, that is, there are no *missing values*. Unfortunately, for most real-life data sets this property does not hold. To use a data set with missing values for learning purposes, therefore, the missing values have to be *filled in*, for example based upon (roughly) estimated prior probabilities for these values or with the help of domain experts; the EM (Expectation-Maximisation) algorithm is often used for this purpose [Dempster *et al.*, 1977]. Other properties assumed by most learning algorithms concern the real-life process that generated the cases comprised in the data set at hand. It is assumed that this process generates cases *independently*, that is, the values specified for the variables in a case are assumed not to be influenced in any way by the values in previously generated cases. Also, it is assumed that the process is not time-dependent, that is, the data set is assumed not to reflect information that varies over time.

As we will discuss in the sequel, a data set of cases over a set of random variables is used for estimating various conditional probabilities. These conditional probabilities are used for analysing the strengths of various different relationships among the variables. The conditional probabilities required for this purpose are obtained from the data set by *counting*. We consider a data set $\boldsymbol{D}$ over the set of variables $\boldsymbol{V}$, comprising N cases. For each set of variables $\boldsymbol{X} \subseteq \boldsymbol{V}$, we will use $N(c_{\boldsymbol{X}})$ to denote the number of cases in $\boldsymbol{D}$ in which the variables from $\boldsymbol{X}$ have adopted the conjunction of values $c_{\boldsymbol{X}}$; for the empty set, we take $N(c_{\varnothing}) = N$. For any two sets of variables $\boldsymbol{X}, \boldsymbol{Y} \subseteq \boldsymbol{V}$, the probability distribution $\Pr(\boldsymbol{X} \mid \boldsymbol{Y})$ can now be estimated from the data set by

$$\Pr(c_{\boldsymbol{X}} \mid c_{\boldsymbol{Y}}) = \frac{N(c_{\boldsymbol{X}} \wedge c_{\boldsymbol{Y}})}{N(c_{\boldsymbol{Y}})}$$

for all configurations $c_{\boldsymbol{X}}$ and $c_{\boldsymbol{Y}}$ of $\boldsymbol{X}$ and $\boldsymbol{Y}$, respectively.

Various algorithms exist for learning Bayesian networks from data. Roughly, two approaches can be distinguished: one where statistical tests are applied to the data to determine the independences; the other where the space of possible networks is searched for a good scoring network. Here, we will focus on a score & search-based method that lies at the basis of most algorithms that are popular today.

The quality measure

An algorithm for learning the qualitative part of a Bayesian network from data typically generates various different acyclic digraphs and compares these as to their ability to describe or explain the data at hand. We will focus on the task of comparing digraphs before turning to their generation.

For comparing digraphs as to their ability to describe the data from the database at hand, a *quality measure* is employed. A quality measure is a function that assigns to a digraph a numerical value expressing how well this digraph fits the data; this numerical value is called the *quality* of the digraph given the database. In the literature, various different quality measures have been proposed, originating from different theories. Here, we will discuss only the *MDL quality measure*. This measure originates from coding theory and is built on the minimum description length principle. For an overview of the most popular quality measures in use for learning the qualitative part of a Bayesian network from data, we refer the reader to [Bouckaert, 1995].

We define the MDL quality measure to pertain to an acyclic digraph and a database.

Definition 5.2.2 *Let* $\boldsymbol{V} = \{V_1, \ldots, V_n\}$*,* $n \geq 1$*, be a set of random variables. Let* $\boldsymbol{D}$ *be a data set over* $\boldsymbol{V}$ *and let* N *be the number of cases in* $\boldsymbol{D}$*. Furthermore, let* P *be a probability distribution over the set of acyclic digraphs with vertex set* $\boldsymbol{V}$*. Let* $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ *be an acyclic digraph with* $\boldsymbol{V}_G = \boldsymbol{V}$*. Then, the* quality of G given $\boldsymbol{D}$*, denoted* $Q(G, \boldsymbol{D})$*, is defined by*

$$Q(G, \boldsymbol{D}) = \log P(G) - N \cdot H(G, \boldsymbol{D}) - \tfrac{1}{2} K \cdot \log N$$

where

$$H(G, \boldsymbol{D}) = - \sum_{V_i \in \boldsymbol{V}} \sum_{c_{V_i}} \sum_{c_{\boldsymbol{\rho}_G(V_i)}} \left(\frac{N(c_{V_i} \wedge c_{\boldsymbol{\rho}_G(V_i)})}{N} \right) \cdot \log \left(\frac{N(c_{V_i} \wedge c_{\boldsymbol{\rho}_G(V_i)})}{N(c_{\boldsymbol{\rho}_G(V_i)})} \right)$$

and

$$K = \sum_{V_i \in \boldsymbol{V}} 2^{|\boldsymbol{\rho}_G(V_i)|}$$

From the previous definition, we have that the quality $Q(G, \boldsymbol{D})$ of an acyclic digraph G given a data set $\boldsymbol{D}$ involves three terms: $\log P(G)$, $-N \cdot H(G, \boldsymbol{D})$ and $-\frac{1}{2} K \cdot \log N$. We will take a closer look at these terms separately.

In the first term, $\log P(G)$, of the quality of a digraph G given a data set $\boldsymbol{D}$, the probability $P(G)$ denotes the prior probability of the digraph G being the qualitative part of a Bayesian network that describes the joint probability distribution that generated the data set $\boldsymbol{D}$. This term therefore expresses information about the 'real' qualitative part *prior* to observation of the database. In a real-life application of a learning algorithm, this term is used for modelling domain knowledge. For example, if a domain expert suggests existence of a specific arc in the qualitative part of the 'real' network, then digraphs that adhere to this suggestion are given a higher prior probability than digraphs that do not. In the sequel, we will assume that no prior domain knowledge is available and, hence, that the probability distribution P is a uniform distribution.

The second term, $-N \cdot H(G, \boldsymbol{D})$, of the quality of a digraph G given a data set $\boldsymbol{D}$ is proportional to the probability of the data set $\boldsymbol{D}$ being generated by the joint probability distribution represented by a Bayesian network $\mathcal{B}$ that involves the digraph G. Now, recall that a Bayesian network not only involves a digraph but also includes a set of probability assessment functions. To enhance the network's 'match' with the data set, all conditional probabilities required are assessed from the data: for each variable $V_i \in \boldsymbol{V}$, we take

$$\gamma_{V_i}(c_{V_i} \mid c_{\boldsymbol{\rho}_G(V_i)}) = \frac{N(c_{V_i} \wedge c_{\boldsymbol{\rho}_G(V_i)})}{N(c_{\boldsymbol{\rho}_G(V_i)})}$$

for all configurations c_{V_i} of V_i and all configurations $c_{\boldsymbol{\rho}_G(V_i)}$ of the set $\boldsymbol{\rho}_G(V_i)$ of V_i's predecessors in G. Now consider the probability of the data set $\boldsymbol{D}$ being generated by the joint probability distribution Pr defined by the Bayesian network $\mathcal{B} = (G, \Gamma)$ where Γ is the set of probability assessment functions estimated from $\boldsymbol{D}$. Since we have assumed that all cases in $\boldsymbol{D}$ have been generated independently, we have that the probability $P'(\boldsymbol{D} \mid \mathcal{B})$ of the data set $\boldsymbol{D}$ given the Bayesian network $\mathcal{B}$ equals

$$P'(\boldsymbol{D} \mid \mathcal{B}) = \prod_{c_{\boldsymbol{V}} \in \boldsymbol{D}} \Pr(c_{\boldsymbol{V}})$$

By exploiting the property stated in Proposition 4.1.3, we find that

$$P'(\boldsymbol{D} \mid \mathcal{B}) = \prod_{c_{\boldsymbol{V}} \in \boldsymbol{D}} \prod_{V_i \in \boldsymbol{V}} \gamma_{V_i}(c_{V_i} \mid c_{\boldsymbol{\rho}_G(V_i)})$$

where $\bigwedge_{V_i \in \boldsymbol{V}} (c_{V_i} \wedge c_{\boldsymbol{\rho}_G(V_i)}) \equiv c_{\boldsymbol{V}}$. By reordering terms, we ultimately find that

$$P'(\boldsymbol{D} \mid \mathcal{B}) = \prod_{V_i \in \boldsymbol{V}} \prod_{c_{V_i}} \prod_{c_{\boldsymbol{\rho}_G(V_i)}} \gamma_{V_i}(c_{V_i} \mid c_{\boldsymbol{\rho}_G(V_i)})^{N(c_{V_i} \wedge c_{\boldsymbol{\rho}_G(V_i)})}$$

Substitution of the probability estimates for the function values of the various different assessment functions now yields

$$P'(\boldsymbol{D} \mid \mathcal{B}) = \prod_{V_i \in \boldsymbol{V}} \prod_{c_{V_i}} \prod_{c_{\boldsymbol{\rho}_G(V_i)}} \left(\frac{N(c_{V_i} \wedge c_{\boldsymbol{\rho}_G(V_i)})}{N(c_{\boldsymbol{\rho}_G(V_i)})} \right)^{N(c_{V_i} \wedge c_{\boldsymbol{\rho}_G(V_i)})}$$

Hence,

$$\log P'(\boldsymbol{D} \mid \mathcal{B}) = N \cdot \sum_{V_i \in \boldsymbol{V}} \sum_{c_{V_i}} \sum_{c_{\boldsymbol{\rho}_G(V_i)}} \left(\frac{N(c_{V_i} \wedge c_{\boldsymbol{\rho}_G(V_i)})}{N} \right) \cdot \log \left(\frac{N(c_{V_i} \wedge c_{\boldsymbol{\rho}_G(V_i)})}{N(c_{\boldsymbol{\rho}_G(V_i)})} \right)$$

The result equals

$$\log P'(\boldsymbol{D} \mid \mathcal{B}) = -N \cdot H(G, \boldsymbol{D})$$

where $H(G, \boldsymbol{D})$ denotes the mutual *entropy* of G and $\boldsymbol{D}$.

Entropy is an information-theoretic measure of uncertainty. In general, the entropy of a variable is maximal when the uncertainty concerning its value is maximal; the entropy is zero when there is complete knowledge as to the variable's value. Informally speaking, we have, in the context of the MDL quality measure, that the better a digraph fits the database, the lower their mutual entropy. In general, an acyclic digraph with more arcs will have a lower entropy than an acyclic digraph with fewer arcs: a digraph with more arcs expresses fewer independences and therefore is capable of capturing more of the nuances reflected in the cases of the data set. So, for a tightly connected acyclic digraph G the term $-N \cdot H(G, \boldsymbol{D})$ in this digraph's quality tends to approach zero.

For an acyclic digraph with more arcs more conditional probabilities have to be estimated from the data set to define the probability assessment functions to be associated with the digraph, than for a digraph with fewer arcs. The more probabilities have to be estimated from the data set, the smaller the number of cases in the data set these estimates can be based upon. So, the more probabilities have to be estimated the less reliable the estimates will be. The third term, $-\frac{1}{2}K \cdot \log N$, of the quality $Q(G, \boldsymbol{D})$ of a digraph G given a database $\boldsymbol{D}$ captures this aspect. In this term, K expresses the number of probabilities that have to be assessed from the data set to define the probability assessment functions for the digraph at hand. So, the more arcs a digraph has, the larger K and the smaller the term $-\frac{1}{2}K \cdot \log N$. The term $-\frac{1}{2}K \cdot \log N$ is often referred to as the *penalty term* as it imposes a penalty on adding arcs to the digraph in the making.

In the sequel, we shall see that an algorithm for learning the digraph of a Bayesian network from data constructs a digraph by successively adding single arcs to an initially arc-less graph. For the initial arc-less digraph, the entropy term $-N \cdot H(G, \boldsymbol{D})$ of its quality will be extremely small and the term $-\frac{1}{2}K \cdot \log N$ will be quite close to zero. On successively adding arcs to the digraph, the term $-N \cdot H(G, \boldsymbol{D})$ will grow rapidly. The term $-\frac{1}{2}K \cdot \log N$ on the other hand will decrease as the digraph becomes more tightly connected. As long as the entropy term increases more rapidly than the penalty term decreases, the quality of the digraph in the making will increase. The increase of the digraph's quality upon successive arc addition continues until the increase of the entropy term is dominated by the decrease of the penalty term.

The search heuristic

We have mentioned before that an algorithm for learning the qualitative part of a Bayesian network from data generates various different acyclic digraphs and compares these digraphs with respect to their quality given the database at hand. The basic purpose of the algorithm is to select from among all possible acyclic digraphs a digraph with *highest quality.* Unfortunately, it is not feasible from a computational point of view to generate all digraphs and compute their qualities. A learning algorithm therefore incorporates a *search heuristic* that searches the set of all possible acyclic digraphs for digraphs that are *likely* to have a high quality; only for these digraphs is the quality given the data set actually computed. Several such search heuristics have been proposed in the literature. Here, we will only discuss the B search heuristic. For an overview of various search heuristics and their properties, we refer the reader once more to [Bouckaert, 1995].

Let $\boldsymbol{V}$ be the set of random variables of the Bayesian network in the making. The B search heuristic for selecting a digraph of high quality from among all possible acyclic digraphs with vertex set $\boldsymbol{V}$ commences with investigating the *arc-less* digraph $G_0 = (\boldsymbol{V}, \varnothing)$. To this initially arc-less graph, the search heuristic successively adds single arcs to improve the digraph's quality. Now, suppose that after adding some arcs, a digraph $G_k = (\boldsymbol{V}, \boldsymbol{A}_{G_k})$ has resulted. For selecting a new arc to add to the digraph, the search heuristic identifies all possible arcs that can be added to the digraph G_k without introducing a cycle. For each identified arc (V_i, V_j), it computes the *gain in quality* that would be yielded by adding this arc to the digraph in the making. So, it computes the difference in quality between the new digraph $G_{k+1} = (\boldsymbol{V}, \boldsymbol{A}_{G_k} \cup \{(V_i, V_j)\})$ and the digraph G_k. The search heuristic then selects an arc that yields the highest gain in quality and adds this arc to the digraph in the making. This process of arc addition is repeated until no gain in quality can be achieved any more. Note that the B search heuristic is a *greedy* search heuristic in the sense that it considers single arcs only.

From the informal outline of the B search heuristic, it will be evident that for selecting a digraph with high quality, the search heuristic has to compare qualities of various different digraphs. We observe, however, that for selecting the next arc to add to the digraph in the making the search heuristic does not actually need to *know* the qualities of the various digraphs considered since an arc is selected on the basis of *differences* in quality between digraphs only. From B being a greedy search heuristic, we further have that these differences are only computed for pairs of digraphs that differ in one arc. The B search heuristic exploits these observations by only *partially* computing the qualities of the digraphs concerned.

For the purpose of partially computing qualities, the B search heuristic makes use of the concept of the quality of a *vertex* in a digraph.

Definition 5.2.3 *Let $\boldsymbol{V} = \{V_1, \ldots, V_n\}$, $n \geq 1$, be a set of random variables. Let $\boldsymbol{D}$ be a data set over $\boldsymbol{V}$ and let N be the number of cases in $\boldsymbol{D}$. Let $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ be an acyclic digraph with $\boldsymbol{V}_G = \boldsymbol{V}$. For each vertex $V_i \in \boldsymbol{V}_G$, the* quality *of V_i in G given $\boldsymbol{D}$, denoted by $q(V_i, \boldsymbol{\rho}_G(V_i), \boldsymbol{D})$, is defined as*

$$\begin{aligned} q(V_i, \boldsymbol{\rho}_G(V_i), \boldsymbol{D}) &= \textstyle\sum_{c_{V_i}} \sum_{c_{\boldsymbol{\rho}_G(V_i)}} N(c_{V_i} \wedge c_{\boldsymbol{\rho}_G(V_i)}) \cdot \log \left(\frac{N(c_{V_i} \wedge c_{\boldsymbol{\rho}_G(V_i)})}{N(c_{\boldsymbol{\rho}_G(V_i)})} \right) + \\ &\qquad - \tfrac{1}{2} \cdot 2^{|\boldsymbol{\rho}_G(V_i)|} \cdot \log N \end{aligned}$$

The quality of an acyclic digraph given a data set can now be expressed in terms of the qualities of its various vertices; this property is known as the *sum property* of a digraph's quality given a database and is stated more formally in the following lemma.

Lemma 5.2.4 *Let $\boldsymbol{V} = \{V_1, \ldots, V_n\}$, $n \geq 1$, be a set of random variables. Let $\boldsymbol{D}$ be a data set over $\boldsymbol{V}$ and let N be the number of cases in $\boldsymbol{D}$. Let P be a probability distribution over the set of all acyclic digraphs with vertex set $\boldsymbol{V}$. Let $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ be such an acyclic digraph with $\boldsymbol{V}_G = \boldsymbol{V}$. Furthermore, let $Q(G, \boldsymbol{D})$ be the quality of G given $\boldsymbol{D}$ and, for each vertex $V_i \in \boldsymbol{V}_G$, let $q(V_i, \boldsymbol{\rho}_G(V_i), \boldsymbol{D})$ be the quality of V_i in G given $\boldsymbol{D}$. Then,*

$$Q(G, \boldsymbol{D}) = \log P(G) + \sum_{V_i \in \boldsymbol{V}} q(V_i, \boldsymbol{\rho}_G(V_i), \boldsymbol{D})$$

Proof. The property stated in the lemma follows directly from Definition 5.2.2 and Definition 5.2.3. □

Note that since we have assumed that the probability distribution P over the set of all acyclic digraphs with a given vertex set is a uniform distribution, the term $\log P(G)$ in a digraph's quality is the same for every digraph G and therefore is a constant with respect to learning the digraph of a Bayesian network from a data set.

Let $\boldsymbol{V}$ once more be the set of random variables discerned and let $G_k = (\boldsymbol{V}, \boldsymbol{A}_{G_k})$ be a digraph that has resulted at some stage during application of the B search algorithm. Now, consider a new digraph $G_{k+1} = (\boldsymbol{V}, \boldsymbol{A}_{G_k} \cup \{(V_i, V_j)\})$ that differs from G_k in the arc (V_i, V_j) only. From Lemma 5.2.4 it is easily seen that the difference in quality between these two digraphs equals the difference in quality of vertex V_j in the two digraphs only; this property is stated in the following corollary.

Corollary 5.2.5 *Let $\boldsymbol{V} = \{V_1, \ldots, V_n\}$, $n \geq 1$, be a set of random variables. Let $\boldsymbol{D}$ be a database over $\boldsymbol{V}$. Let $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ and $G' = (\boldsymbol{V}_{G'}, \boldsymbol{A}_G \cup \{(V_i, V_j)\})$ be acyclic digraphs with $\boldsymbol{V}_G = \boldsymbol{V}_{G'} = \boldsymbol{V}$ Furthermore, let $Q(G, \boldsymbol{D})$ and $Q(G', \boldsymbol{D})$ be the quality of G given $\boldsymbol{D}$ and the quality of G' given $\boldsymbol{D}$, respectively. For each $V_k \in \boldsymbol{V}$, let $q(V_k, \boldsymbol{\rho}_G(V_k), \boldsymbol{D})$ and $q(V_k, \boldsymbol{\rho}_{G'}(V_k), \boldsymbol{D})$ be the quality of vertex V_k in G and the quality of V_k in G' given $\boldsymbol{D}$, respectively. Then,*

$$Q(G', \boldsymbol{D}) - Q(G, \boldsymbol{D}) = q(V_j, \boldsymbol{\rho}_{G'}(V_j), \boldsymbol{D}) - q(V_j, \boldsymbol{\rho}_G(V_j), \boldsymbol{D})$$

The B search heuristic now exploits the property stated in Corollary 5.2.5 for computing the gain in quality that would result from adding a new arc to the digraph in the making.

The following pseudocode summarises the B search heuristic.

```
procedure construct-digraph($\boldsymbol{V}$,$\boldsymbol{D}$,$G$)
    for each variable $V_i \in \boldsymbol{V}$ do
        $\boldsymbol{\rho}_G(V_i) := \varnothing$
    od;
    repeat
        for each (ordered) pair $V_i, V_j \in \boldsymbol{V}$ such that addition of the arc $(V_i, V_j)$
            to $G$ does not introduce a (directed) cycle do
            $\mathit{diff}(V_i, V_j) := q(V_j, \boldsymbol{\rho}_G(V_j) \cup \{(V_i, V_j)\}, \boldsymbol{D}) - q(V_j, \boldsymbol{\rho}_G(V_j), \boldsymbol{D})$
        od;
        select the pair $V_i, V_j \in \boldsymbol{V}$ for which $\mathit{diff}(V_i, V_j)$ is maximal;
        if $\mathit{diff}(V_i, V_j) > 0$ then
            $\boldsymbol{\rho}_G(V_j) := \boldsymbol{\rho}_G(V_j) \cup \{V_i\}$
        fi
    until $\mathit{diff}(V_i, V_j) \leq 0$
end
```

To conclude, we would like to point out that the learning algorithm as outlined above is not guaranteed to find a minimal I-map for the joint probability distribution underlying the process that generated the database used. It will be evident that one reason for the learning algorithm not finding such a digraph is that it incorporates a search *heuristic* that considers only a limited number of possible acyclic digraphs and therefore incidentally may skip the minimal I-maps. Yet another reason lies in the quality measure used, however. For infinite size databases, the MDL-measure can be shown to prefer minimal I-maps over all other acyclic digraphs. Unfortunately, this property is not retained for finite size databases. So, by employing the MDL-measure for a finite size database, a digraph that is not a minimal I-map can be selected, even if every possible acyclic digraph were considered.

The methods presented in this section still lie at the basis of modern score-based algorithms for unsupervised structure learning of Bayesian networks of general topology. Numerous other types of algorithm have been applied to the general problem of Bayesian network structure learning, see[Berzan, 2012] or [Stahlschmidt et al.*, 2013] for a nice overview. In addition to structure learning, various approaches to learning the probability assessment functions of a Bayesian network exist (see e.g. [Ji* et al.*, 2015] for a review). With the current interest in machine learning and the increasing availability of data sets, a lot of research in the Bayesian network community is also focussed on learning, resulting in a huge body of literature on the topic. Since the focus of this syllabus is more on knowledge representation and reasoning and not on data science, we hereby conclude the discussion of learning models from data.*

5.3 Assessing Probabilities

Only after the qualitative part of the Bayesian network in the making has been constructed and is considered robust, is its quantitative part specified. Specifying the quantitative part of a Bayesian network amounts to defining the probability assessment functions for the variables modelled in the network. The task of assessing all required probabilities tends to be by far the hardest task in Bayesian network building.

In order to facilitate the assessment task, approaches have been introduced that first attach qualitative signs to the arcs of a network's digraph, resulting in a so-called qualitative Bayesian network [Wellman, 1990, Renooij, 2001]. The signs capture the direction of change in (cumulative) conditional probability upon shifting from a lower ordered value to a higher ordered value of a conditioning variable. The benefits are that these qualitative signs are easily elicited from domain expert, for example by using the method described by [Van der Gaag & Helsper, 2004]; the ordering of probabilities for different conditioning contexts was also found to be consistent over multiple experts [Van der Gaag & *et al.*, 2012]. In addition, the signs pose constraints on the probability distributions to be assessed, which can be exploited upon further assessment [Renooij & Van der Gaag, 2002]; different combinations of qualitative signs, for example, can be an indication of disjunctive interaction patterns, such as the *noisy-or* described in Section 5.3.2 [Lucas, 2005]. In the end, though, the Bayesian network requires point probabilities for its specification.

5.3.1 Sources of Probabilistic Information

In most problem domains, at least some probabilistic information for Bayesian network quantification is readily available, be it from literature or from domain experts. The most common sources of probabilistic information are [Jensen, 1995]:

- *literature* on the domain of application (textbooks, journals, conferences);

- *(statistical) data*;
- (other) *models* of domain knowledge;
- interviews with domain *experts*.

Using literature Literature on the domain of application often provides abundant probabilistic information. Unfortunately, this information is very seldom directly amenable to encoding in the Bayesian network in the making: the information typically is not complete, it concerns variables that are not causally related, and so on. Medical literature, for example, often reports conditional probabilities of the presence of symptoms given a disease, but not always the probabilities of these symptoms occurring in the absence of disease; moreover, conditional probabilities for unobservable intermediate disease states are usually lacking. An additional, commonly found problem that prohibits direct use of probabilistic information from literature is that the characteristics of the population from which the information is derived, is not properly described or deviates seriously from the characteristics of the population for which the Bayesian network is being developed [Korver & Lucas, 1993].

Using data If the literature on the domain at hand does not provide for sufficient and reliable probability assessments for quantification of the network in the making, estimates may be obtained from statistical data or from other models of domain knowledge. The discussed algorithm for learning structure from data, for example, also computes the assessment fucntions as a (necessary) by-product. Unfortunately, experience shows that even if comprehensive data collections and models are available, they very seldom contribute significantly to the reliable quantification of detailed models[Jensen, 1995, Korver & Lucas, 1993]. In a medical data collection, for example, unobservable intermediate patho-physiological states are typically not recorded. Although techniques exist to discover such hidden or latent variables in data, their interpretation in the domain is often unclear. In case of little data, Bayesian approaches to learning unknown quantities, such as model-parameters, are most promising[Murphy, 2022]. In such approaches, a model-parameter γ_V for a variable V is itself represented as a random variable with a prior distribution. For learning the actual value of the parameter, the observations in the data are entered into the network case by case and probabilistic inference serves to update the parameter's value. The posterior is then taken as the new prior and the process is repeated. This approach, where parameters are learned from data by means of updating priors through probabilistic inference is called *Bayesian inference* and is typically used in *Probabilistic programming.*

Using domain expertise Depending on the domain and the complexity of the model under construction, we may be faced with a situation in which the majority of the probabilities required will have to be assessed by domain experts. The problems encountered when eliciting probabilities from experts are widely known [Tversky *et al.*, 1982]; an expert's assessments may for example reflect various biases and may not be properly calibrated. The next sections describe several approaches that attempt to counter these problems, at least to some extent.

5.3.2 Simplifying Probability Assessment

The task of assessing all probabilities required for quantifying a Bayesian network can often be simplified by assuming the presence of one or more *disjunctive interactions* in the network [Pearl, 1988]. A disjunctive interaction pertains to two or more causes and their common effect: it specifies that any of these causes in itself suffices to cause the effect and that the likelihood of this cause causing the effect does not diminish when several other causes are present simultaneously.

More formally, we consider a set of random variables $\boldsymbol{V}$ and an acyclic digraph $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$ with $\boldsymbol{V}_G = \boldsymbol{V}$ where $(V_1, V_0), \ldots, (V_m, V_0) \in \boldsymbol{A}_G$, for some vertices $V_0, V_1, \ldots, V_m$, $m \geq 1$. Note that the digraph G expresses that $V_1, \ldots, V_m$ model different causes of a common effect modelled by V_0. The variables $V_1, \ldots, V_m$ are said to exhibit a disjunctive interaction with respect to V_0 if the variables $V_1, \ldots, V_m$ adhere to the properties of accountability and exception independence. The property of *accountability* expresses that at least one of the causes modelled by $V_1, \ldots, V_m$ needs to be present for V_0 to adopt the value *true*, or alternatively, that the variable V_0 takes the value *false* whenever all its causes are absent. The property of accountability is stated in terms of probabilities as

$$\Pr(v_0 \mid \neg v_1 \wedge \cdots \wedge \neg v_m) = 0$$

We now turn to the property of *exception independence*. This property is best explained in terms of a *logical or-gate*. Note that in a logical or-gate, the presence of any one of the causes $V_1, \ldots, V_m$ suffices for V_0 to adopt the value *true*. Now, if the logical relationship between V_0 and its separate causes is perturbed, we say that the logical or-gate is *noisy*. In a noisy or-gate, for each cause V_i a so-called *exception mechanism* may *inhibit* the presence of the cause to result in $V_0 =$ *true*; such an exception mechanism is often termed an *inhibitor*. The exception mechanisms in a noisy or-gate may be looked upon as random variables. If the exception mechanism for a cause V_i has the value *true*, then it inhibits the occurrence of the effect $V_0 =$ *true*; if V_i's exception mechanism has the value *false*, then it does not prevent the occurrence of the effect upon presence of the cause modelled by V_i. We therefore can model the exception mechanism of a cause V_i by a random variable I_i such that

$$\Pr(v_0 \mid \neg v_1 \wedge \cdots \wedge \neg v_{i-1} \wedge (v_i \wedge i_i) \wedge \neg v_{i+1} \wedge \cdots \wedge \neg v_m) = 0$$

and

$$\Pr(v_0 \mid \neg v_1 \wedge \cdots \wedge \neg v_{i-1} \wedge (v_i \wedge \neg i_i) \wedge \neg v_{i+1} \wedge \cdots \wedge \neg v_m) = 1$$

Exception independence for $V_1, \ldots, V_m$ now is the property that random variables $I_1, \ldots, I_m$ modelling the exception mechanisms for $V_1, \ldots, V_m$, respectively, are mutually independent. Because of the analogy outlined above, a disjunctive interaction is often coined a *noisy or-gate*.

The information that $V_1, \ldots, V_m$ exhibit a disjunctive interaction with respect to V_0 is not directly amenable to encoding in a Bayesian network. The information, however, imposes strong restrictions on the probability assessment function γ_{V_0} for the vertex V_0 in the network. To support this observation, we consider the various values that have to be assessed to define the function γ_{V_0}, separately:

- for the value $\gamma_{V_0}(v_0 \mid \neg v_1 \wedge \cdots \wedge \neg v_m)$, we have

  $$\gamma_{V_0}(v_0 \mid \neg v_1 \wedge \cdots \wedge \neg v_m) = 0$$

  from the property of accountability of a disjunctive interaction;

- for $i = 1, \ldots, m$, let I_i be the inhibitor for cause V_i and let $\Pr(i_i) = q_i$; then, we have that

  $$\gamma_{V_0}(v_0 \mid \neg v_1 \wedge \cdots \wedge \neg v_{i-1} \wedge v_i \wedge \neg v_{i+1} \wedge \cdots \wedge \neg v_m) = 1 - q_i$$

- let c be an arbitrary configuration of the set of causes $\{V_1, \ldots, V_m\}$ and let $T_c = \{i \mid c \wedge v_i \not\equiv \mathsf{False}\}$; then, we have that

$$\gamma_{V_0}(v_0 \mid c) = 1 - \prod_{i \in T_c} q_i$$

  from the property of exception independence of a disjunctive interaction.

From these observations we have that, once causes $V_1, \ldots, V_m$ are known to exhibit a disjunctive interaction with respect to V_0, it suffices to assess only n probabilities to define the probability assessment function γ_{V_0} for vertex V_0.

Recall that a disjunctive interaction adheres to the properties of accountability and exception independence. We take a closer look at the property of accountability. Recall that this property states that all causes of an effect are known and explicitly modelled in the Bayesian network in the making. We observe that this property is hardly ever satisfied in real-life applications. Since incompleteness is inherent to any model of domain knowledge, a Bayesian network will never be complete: there will always be possible causes of an effect that are not explicitly modelled in the network. Unfortunately, if the property of accountability is not satisfied, it is not possible to exploit the model of the noisy or-gate as described above — not even it the property of exception independence is satisfied.

Consider the variables $V_1, \ldots, V_m$ modelling different causes of a common effect expressed by V_0. Now, assume that the property of exception independence is satisfied and that the property of accountability is not: we have that

$$\Pr(v_0 \mid \neg v_1 \wedge \cdots \wedge \neg v_m) = p$$

where $p > 0$. To be able to nevertheless exploit the model of the noisy or-gate, we can extend the Bayesian network in the making with an extra random variable in such a way that the property of accountability is enforced and the property of exception independence is retained. To this end, we introduce a new variable V_{m+1} into the network as modelling an extra cause of the effect V_0 — this variable then is taken to capture all causes of V_0 that are not yet modelled in the original network. For this variable V_{m+1}, we then have

$$\Pr(v_0 \mid \neg v_1 \wedge \cdots \wedge \neg v_m \wedge \neg v_{m+1}) = 0$$

and

$$\Pr(v_0 \mid \neg v_1 \wedge \cdots \wedge \neg v_m \wedge v_{m+1}) = p$$

Furthermore, we take V_{m+1} to be independent of $V_1, \ldots, V_m$. Note that the introduction of the variable V_{m+1} has not altered the meaning of the original causal mechanism. Also note that the property of accountability is now satisfied by $V_1, \ldots, V_{m+1}$ and that the model of the noisy or-gate applies.

Although the introduction of an extra variable as outlined before allows for exploiting the model of the noisy or-gate in the case where the property of accountability is not satisfied, adding an extra variable to the network is not satisfactory from a knowledge representational point of view: the additional variable has no well-defined meaning and therefore in a sense pollutes the Bayesian network. The model of the *leaky* noisy or-gate now allows for capturing all yet unmodelled causes of a common effect *implicitly* [Henrion, 1989]. Consider once more the variables $V_0, V_1, \ldots, V_m$ and assume that the property of exception independence is satisfied and the property of accountability is not; recall that we have that

$$\Pr(v_0 \mid \neg v_1 \wedge \cdots \wedge \neg v_m) = p$$

where $p > 0$. The probability p is called a *leak probability* and is looked upon as expressing the probability that the effect v_0 occurs *spontaneously.* Now, by introducing an extra inhibitor I_0 with probability $\Pr(i_0) = q_0 = 1 - p$ similar observations apply to the leaky noisy or-gate as for the noisy or-gate: the information that the variables $V_0, V_1, \ldots, V_m$ constitute a leak noisy or-gate again imposes strong restrictions on the definition of the probability assessment function γ_{V_0} for the variable V_0:

- for the probability $\gamma_{V_0}(v_0 \mid \neg v_1 \wedge \cdots \wedge \neg v_m)$, we have

$$\gamma_{V_0}(v_0 \mid \neg v_1 \wedge \cdots \wedge \neg v_m) = p$$

- for $i = 1, \ldots, m$, let I_i be the inhibitor for cause V_i and let $\Pr(i_i) = q_i$; then, we have that

$$\gamma_{V_0}(v_0 \mid \neg v_1 \wedge \cdots \wedge \neg v_{i-1} \wedge v_i \wedge \neg v_{i+1} \wedge \cdots \wedge \neg v_m) = 1 - q_i$$

- let c be an arbitrary configuration of the set of causes $\{V_1, \ldots, V_m\}$, let $T_c = \{i \mid c \wedge v_i \not\equiv \mathsf{False}\}$, and let $q_0 = 1 - p$; then, we have that

$$\gamma_{V_0}(v_0 \mid c) = 1 - q_0 \cdot \prod_{i \in T_c} \left(\frac{q_i}{q_0} \right)$$

  from the property of exception independence of a disjunctive interaction.

Note that the values for the probability assessment function γ_{V_0} are defined much in the same way as in the model of the noisy or-gate. Also note that, since in the Bayesian network in the making the effect of a cause cannot be isolated from the spontaneous occurrence of the effect, the probability assessments of the effect given a *single* cause are adjusted to account for this spontaneous leak before they are combined.

5.3.3 Eliciting Probabilities from Experts

The field of *decision analysis* offers various techniques for the elicitation of judgemental probabilities from experts[Von Winterfeldt & Edwards, 1986, Morgan & Henrion, 1990]. We briefly review the two techniques that are most often used for eliciting probabilities for Bayesian networks. The simplest technique is the use of a numerical *probability scale.* A probability scale is a horizontal or vertical line with the endpoints denoting a 0% and a 100% chance, respectively, and a few numerical anchors in between, for example to denote a 50% chance; Figure 5.2 illustrates the basic idea. For each probability required, a domain expert is asked to indicate his or her assessment on a separate scale. In communicating a probability to be assessed to a domain expert, the probability is often transcribed verbally in terms of frequencies. The expert is asked to envisage one hundred cases within a specific context and assess the number of cases that exhibit a certain characteristic. Experience shows that the use of a probability scale along with the frequency method provides experts little to go by and may result in highly inaccurate probability assessments [Van der Gaag *et al.*, 1999].

A more elaborate technique for the elicitation of judgemental probabilities is the use of *reference lotteries.* A domain expert is presented with a choice between two lotteries, one

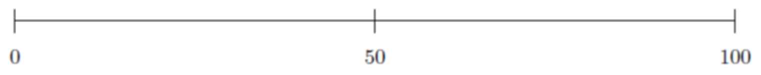


Figure 5.2: A numerical scale for probability elicitation.

of which pertains to the probability to be assessed and the other one serves as a reference. The reference lottery yields a desired reward with probability p and a less desired outcome with probability $1 - p$. The second lottery yields the same desired reward if a specific case exhibits a certain characteristic and the less desired outcome otherwise.

Example 5.3.1 Suppose that, in a medical domain of application, an expert has to assess the probability of a specific patient with metastatic cancer showing an increased level of serum calcium. The domain expert is presented, for instance, with a choice between a reference lottery and the lottery that yields \$10,000 if the patient upon examination shows an increased level of serum calcium and \$1 if the level of serum calcium is not increased in the patient. Figure 5.3 depicts this choice between lotteries. □

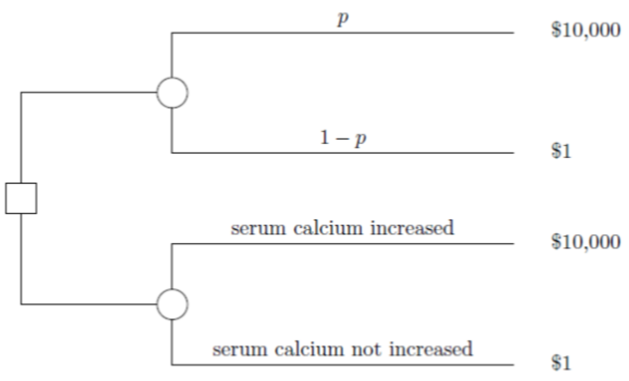


Figure 5.3: An example reference lottery.

The domain expert is asked to adjust the value of p in the reference lottery until he or she is indifferent between the two lotteries. The resulting value of p then is taken to be the conditional probability that had to be assessed. Experience shows that the use of reference lotteries for eliciting probabilities from domain experts may avert to at least some extent the problems of bias and poor calibration that are typically found in human probability assessment. The use of lotteries, however, tends to be quite time-consuming and, in fact, often turns out to be infeasible for probability elicitation for Bayesian networks as a result of the large size and complexity of a network in the making.

While the use of lotteries for probability elicitation on the one hand tends to be infeasible for quantifying a Bayesian network, the use of a probability scale on the other hand tends to yield assessments that are too much inaccurate. The use of a probability scale, nonetheless, serves to yield an assessment, be it an inaccurate one, for every conditional probability required. These assessments may then be used as a starting point for further refinement. More details on probability elicitation methods, including a method designed by Utrecht University researchers, which is based on the use of a verbal-numerical probability scale, can be found in [Renooij, 2001b].

5.3.4 A Procedure for Probability Refinement

The assessments obtained for a Bayesian network are inevitably inaccurate, due to incompleteness of data and partial knowledge of the problem under study. Particularly assessments obtained from experts are known to be highly inaccurate [Tversky *et al.*, 1982]. The inaccuracies in the probability assessments for a Bayesian network influence the reliability of the network's output. *Sensitivity analysis* is a general technique for studying the effects of the uncertainties in the parameters of a mathematical model on this model's outcome

[Morgan & Henrion, 1990]. For a Bayesian network, sensitivity analysis provides, for example, for studying the effects of the uncertainties in the assessments for the network's conditional probabilities on a probability of interest. There are various different types of sensitivity analysis. For a Bayesian network, the simplest type of sensitivity analysis amounts to systematically varying the assessment for one of the network's probabilities while keeping all other assessments fixed. Such an analysis serves to reveal the effect of just the conditional probability whose assessment is being varied, on a probability of interest. A sensitivity analysis in which a single assessment is varied, is termed a *one-way sensitivity analysis.* In a *two-way sensitivity analysis* of a Bayesian network, two probability assessments are varied simultaneously. In addition to the separate effects of variation of the two assessments, a two-way sensitivity analysis reveals the joint effect of their variation on a probability of interest. In essence, it is also possible to systematically vary more than two probability assessments at the same time. The results of such an analysis, however, are often hard to interpret.

Sensitivity analysis of a Bayesian network provides for studying the effects of the uncertainties in the various probability assessments of the network on a probability of interest. Studying these effects can to a large extent serve to support the elicitation of probabilities, as it gives detailed insight into the level of accuracy that is required for the various probabilities of the network, and as a result provides for focusing further elicitation efforts. We describe an *elicitation procedure* in which, alternatingly, sensitivity analyses are performed of a Bayesian network in the making and probability assessments are refined; this procedure is summarised in Figure 5.4. We elaborate on the various different steps of the procedure separately.

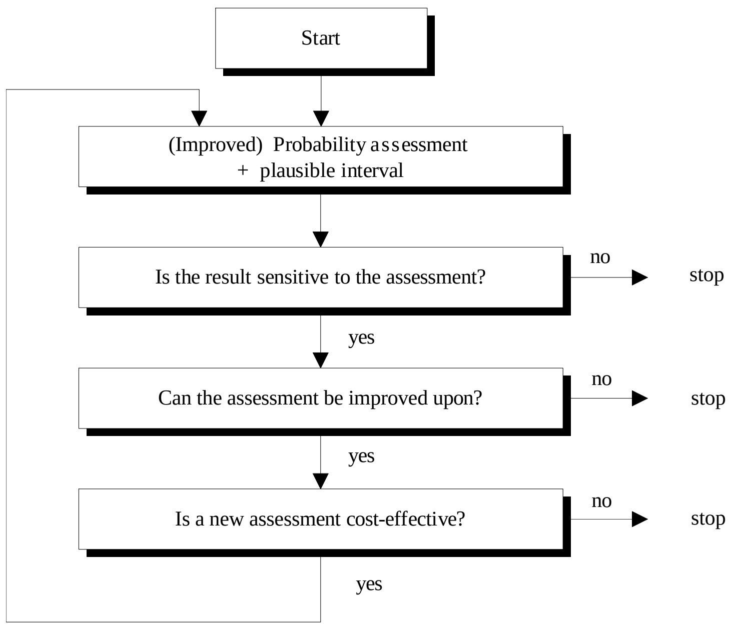

Start
(Improved) Probability assessment + plausible interval
Is the result sensitive to the assessment?
no
stop
yes
Can the assessment be improved upon?
no
stop
yes
Is a new assessment cost-effective?
no
stop
yes

Figure 5.4: A procedure for probability elicitation for Bayesian networks.

Step 1 In the first step of the elicitation procedure, *initial assessments* are acquired for all conditional probabilities for a Bayesian network in the making. As we have argued before, for most domains of application, experts will have to provide the majority of these initial assessments. In this phase of the construction of the Bayesian network, we apply an elicitation technique that aims at acquiring assessments *within a short period of time.* To this end, in a limited number of interview sessions, experts are asked to assess all probabilities required off the top of their heads, for example using a numerical, or verbal-numerical, probability scale. In addition, they are asked to indicate *plausible intervals* along with their assessments, that define a range of values in which the 'true' probability lies with reasonable certainty. These intervals should be pessimistic rather than optimistic to ensure that the uncertainties in the various different assessments are not underestimated. Since experts are thus allowed to express the uncertainties in their assessments, they will tend to be less reluctant to provide numerical statements than when they feel compelled to give exact numbers.

Instead of eliciting all probabilities required from domain experts, initial assessments for at least some of these probabilities may be obtained from data, if available. If the data at hand is known to be imperfect, incomplete, or biased, then the assessments should be supplemented with fairly large plausible intervals to capture the uncertainties involved.

Step 2 The probability assessments obtained in the first step of the elicitation procedure will in general be highly uncertain and should have considerably large plausible intervals. For some of the probabilities, these initial assessments will nonetheless suffice. Other probabilities, however, will require assessments with a higher level of accuracy. The second step of our elicitation procedure is aimed at uncovering the latter probabilities. For this purpose, the Bayesian network in the making is subjected to various *sensitivity analyses*. In these analyses, every single probability assessment for the network is varied, as is every pair of assessments. These analyses require a lot of computation and generate a huge amount of data from which we need to decide which probability assessments require further attention. This computational and informational burden, including the technical details of sensitivity analysis, is further discussed in Section 6.1.

If performing all sensitivity analyses is too time-consuming to be practicable, the analyses can be restricted to a moderate number of assessments. To this end, experts may be asked to indicate the probability assessments that are the most likely to affect a probability of interest or, alternatively, to indicate a number of conceptually related probability assessments.

Step 3 In the second step of the elicitation procedure, various probability assessments have been identified that induce a considerably large effect on a probability of interest (see Section 6.1 on criteria for determining what 'large effect' means). These assessments are potential candidates for further refinement. In the third step of the procedure, the extent to which these assessments can actually be refined is determined; this depends on the current uncertainty in the assessment, indicated by the width of its plausible interval, and on the elicitation techniques used to arrive at the assessment. A probability assessment with a rather small plausible interval obtained from applying elaborate elicitation techniques may not lend itself to further refinement. An assessment that is not yet very certain, on the other hand, may be more easily improved upon. Refinement of such an uncertain assessment, however, is only actually possible if reliable sources of probabilistic information remain to be explored; examples of such sources of information include further literature search, the use of a panel of experts, the use of more elaborate elicitation techniques, and the study of data, for instance collected in a prospective study. From among the potential candidates for further refinement, therefore, the assessments are identified that have a considerable plausible interval and for which yet unexplored sources of probabilistic information are available.

Step 4 The probability assessments that have been selected in the third step of the elicitation procedure are assessments that *can* be refined. To actually refine these assessments, an investment of time and money is required. In the fourth step of the elicitation procedure it is investigated for each of the selected assessments whether refinement is *cost-effective*, by weighing the costs in terms of money and time to be invested with the benefits of higher accuracy. The benefits of having an assessment of higher accuracy in the Bayesian network in the making may be a higher accuracy of a computed probability of interest and an improved performance more in general. For example, for a Bayesian network that is to be used for diagnostic purposes, performance may be measured as the percentage of correctly diagnosed cases. Refining a probability assessment for this network would only be worthwhile if it would increase the number of correct diagnoses (see Chapter6 on evaluation). Once a Bayesian network in the making exhibits satisfactory overall behaviour, refining assessments may be found to be no longer cost-effective.

Recursive step The probability assessments that have been identified in the fourth step of the elicitation procedure are known to induce a considerable effect on a probability of interest; moreover, any such assessment can be cost-effectively refined. For these assessments, the entire elicitation procedure is recursively repeated. The probabilities concerned are assessed anew in the first step of the next cycle of the procedure, using yet unexplored sources of probabilistic information. Since the plausible intervals of the initial assessments for these probabilities do not underestimate the uncertainties involved, refinement is likely to result in smaller plausible intervals for the new assessments. In the second step, once again several sensitivity analyses are performed. These analyses should not only focus on the new assessments, but also on previously investigated assessments as their effect on a probability of interest upon variation may have changed as a result of the refinement of the network. In addition, analyses may be performed with respect to assessments that have not been investigated before. By recursively refining probability assessments, the performance of the Bayesian network in the making is likely to gradually improve. The elicitation procedure is stopped as soon as the costs of further elicitation outweigh the improvement in the network's performance or higher accuracy can no longer be attained due to lack of knowledge. You have now finished constructing the first prototype of your network and are ready for evaluating its practical use.

Exercises

* Exercise 5.1

In constructing the qualitative part of a Bayesian network for a domain of application, a knowledge engineer aims at a digraph that is as sparse as possible. Give at least two reasons for this goal.

* Exercise 5.2

In the construction of the qualitative part of a Bayesian network for a domain of application, a cycle may arise. Describe at least two methods for removing the cycle from the network in the making.

* Exercise 5.3

Suppose that you are asked to construct a Bayesian network for a major medical center in Utrecht, to support specialists in the diagnosis and prognostication of various types of vascular disease. The network is to be constructed by hand, with the help of a domain expert.

a. *The expert indicates that there are four important types of vascular disease; these types of disease are not mutually exclusive. Now, suppose that for pragmatical reasons, the network should have just one hypothesis variable. Explain how the various types of disease can be modelled in the network's digraph.*

b. *Suppose that you decide to model a single-valued discrete domain variable by means of* two *random variables. Explain how the relationship between these two variables can be modelled.*

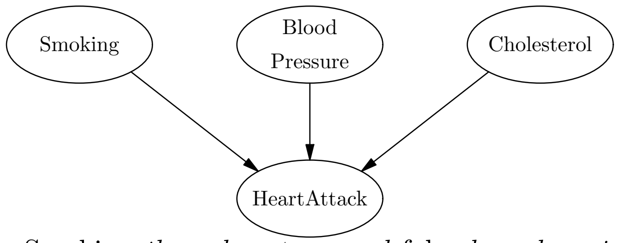
c. *After the digraph of the network has been completed, it becomes clear that it requires simply too many probabilities for its quantification. Describe at least four ways to reduce the number of probabilities required. For each of these, describe in which situations it can lead to a substantial reduction of the number of probabilities required and in which situations it cannot.*

* Exercise 5.4

Suppose that for the hospitals in the Netherlands, a Bayesian network is being built for the diagnosis of acute cardiac disorders and that the network is being handcrafted with the help of domain experts.

a. *For eliciting the topology of the digraph of the network in the making, the concept of causality is used as a heuristic guiding rule. Give an example showing that the concept of causality is not always suitable for this purpose.*

Suppose that the digraph of the Bayesian network in the making comprises the following subgraph:



For the variable Smoking *the values* true *and* false *have been identified, denoted as s and $\neg s$, respectively; for the variable* BloodPressure *the values b and $\neg b$ have been identified, for the variable* Cholesterol *the values c and $\neg c$, and for the variable* HeartAttack *the values h and $\neg h$.*

b. *Suppose that from an emergency medical center in New York, a data collection is available with the medical records of 12738 patients with heart disease. Explain why this data collection cannot be used just like that for the assessment of the probabilities required for the variable HeartAttack in the network fragment above.*

c. *Before commencing with the elicitation of the required probabilities, you would like to ask the domain expert whether or not a* disjunctive interaction *may be assumed for the variable* HeartAttack *and its parents. Which questions would you pose?*

d. *Now suppose that the domain expert indicates that the variables* Smoking, BloodPressure *and* Cholesterol *satisfy the property of exception independence with respect to the variable* HeartAttack. *The domain expert further assesses the following prob-*

abilities:

$$Pr(h \mid \neg s \wedge \neg b \wedge \neg c) = 0.05$$
$$Pr(h \mid s \wedge \neg b \wedge \neg c) = 0.6$$
$$Pr(h \mid \neg s \wedge b \wedge \neg c) = 0.8$$
$$Pr(h \mid \neg s \wedge \neg b \wedge c) = 0.9$$

Is it possible to specify a complete probability assessment function for the variable HeartAttack *on the basis of the available information ? If no, then which information is missing; if yes, then give the complete assessment function.*

* Exercise 5.5

Contrary to other exercises, this exercise is not representative of an exam question. It is mainly used to illustrate the kind of issues you can run into when you try to gather probabilistic information from various sources. Moreover, in answering the question you can once again practice al the BN basics: how is the network defined, how can I appy different rules from probability theory while exploiting the independences encoded in the graph? Be sure to draw the described graph.

Suppose that with the help of a domain expert, a digraph of a Bayesian network is constructed consisting of the variables V_1, V_2, V_3, and V_4, and the arcs (V_1, V_2), (V_2, V_3), and (V_2, V_4). From the literature on the domain under study, the following probabilistic information is available about the four variables:

$$\Pr(v_1 \wedge v_2) = 0.25$$
$$\Pr(\neg v_1 \wedge \neg v_2) = 0.3$$
$$\Pr(v_2 \wedge \neg v_3) = 0.2$$
$$\Pr(v_2 \wedge v_3) = 0.25$$
$$\Pr(\neg v_2 \wedge v_3) = 0.15$$
$$\Pr(v_3 \mid \neg v_1 \wedge v_2 \wedge v_4) = 0.4$$
$$\Pr(v_4) = 0.8$$

No further information has been found and the information provided is not completely coherent. Construct the assessment functions for as many variables as possible from the available information. Note that some assessment functions can be constructed in multiple ways, possibly resulting in different probabilities due to the incoherencies.

* Exercise 5.6

Consider the following procedure for eliciting probability assessments from domain experts:

- *the expert is asked to use his own keywords and phrases to indicate his assessments for the various probabilities;*
- *the expert is then asked to rank order the keywords and phrases he used;*
- *the knowledge engineer subsequently associates numerical probabilities with the various keywords and phrases.*

Describe the strengths and weaknesses of this procedure.

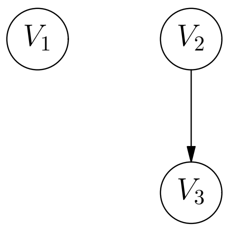
* Exercise 5.7

a. *Let* $\boldsymbol{V} = \{V_1, V_2, V_3\}$ *be a set of binary random variables. Let* $\boldsymbol{D}$ *be a database over* $\boldsymbol{V}$ *comprising the following cases:*

$v_1 \wedge \neg v_2 \wedge \neg v_3$	$\neg v_1 \wedge v_2 \wedge v_3$
$v_1 \wedge v_2 \wedge v_3$	$\neg v_1 \wedge \neg v_2 \wedge \neg v_3$
$v_1 \wedge \neg v_2 \wedge \neg v_3$	$v_1 \wedge \neg v_2 \wedge \neg v_3$
$v_1 \wedge v_2 \wedge v_3$	$\neg v_1 \wedge \neg v_2 \wedge v_3$
$v_1 \wedge \neg v_2 \wedge \neg v_3$	$\neg v_1 \wedge \neg v_2 \wedge \neg v_3$
$\neg v_1 \wedge \neg v_2 \wedge v_3$	$v_1 \wedge v_2 \wedge v_3$
$\neg v_1 \wedge v_2 \wedge \neg v_3$	$v_1 \wedge v_2 \wedge v_3$
$v_1 \wedge \neg v_2 \wedge \neg v_3$	$v_1 \wedge v_2 \wedge \neg v_3$
$v_1 \wedge \neg v_2 \wedge \neg v_3$	$v_1 \wedge v_2 \wedge v_3$
$v_1 \wedge \neg v_2 \wedge v_3$	$v_1 \wedge v_2 \wedge v_3$

There is no other information available on the variables V_1, V_2 *and* V_3*.*

Suppose that the database $\boldsymbol{D}$ *is exploited for automated construction of a Bayesian network. For constructing the network, the B search heuristic is used in combination with the MDL quality measure. Now suppose that at some stage, the following digraph is constructed:*



Compute the difference in quality that is achieved by adding the arc (V_1, V_2) *to this digraph.*

b. *In practical applications, the B search heuristic is applied not only in combination with the MDL quality measure, but also in combination with other quality measures. An example of such a measure is the* Akaike information criterion*; this criterion is defined as*

$$Q_A(G, \boldsymbol{D}) = \log P(G) - N \cdot H(G, \boldsymbol{D}) - K$$

where $G, \boldsymbol{D}, P, N, H$ *and* K *have the same meaning as in the MDL quality measure. Now, for a given database* $\boldsymbol{D}$*, let* G_A *be the digraph that is yielded by the B search heuristic with the Akaike information criterion and let* G_{MDL} *be the digraph that is yielded by the heuristic with the MDL quality measure. Describe in which respect the digraphs* G_A *and* G_{MDL} *will differ in general.*

* Exercise 5.8

Consider the automated construction of a Bayesian network for the domain of classical swine fever.

a. *Suppose you have available a sufficiently large, but incomplete data set. To use this data set for learning a Bayesian network, we need to cope with these 'missing values'. Describe at least two approaches to doing so.*

b. *Indicate what the effects of filling in missing values can be on the structure learnt with MDL and the B-search algorithm.*

c. *In the domain of classical swine fever, numerous variables have more than two values. The penalty term $\frac{1}{2} \cdot K \cdot \log N$ discussed with the MDL measure assumes all variables are binary-valued. Does it make sense to adapt this penalty term for non-binary variables? If not: explain why not. If so: explain how to adapt the penalty term and what the effect of this adaptation on your structure will be.*

* Exercise 5.9

Suppose we want to model a disjunctive interaction that is more restrictive than noisy-OR: the effect V_0 can only be present when exactly one uninhibited *cause V_i is present, instead of when at* least *one uninhibited cause is present. That is, we want to model a noisy-XOR gate.*

Since a noisy-XOR gate captures a disjunctive interaction, albeit more restricted than the noisy-OR, the properties of accountability and exception independence still apply. Now, let c be any configuration for the m causes $V_1,\ldots,V_m$. Let T_c denote the set of indices of present causes in c, that is, $T_c = \{i \mid c \wedge v_i \not\equiv \mathsf{False}\}$.

Note that for a noisy-XOR, we still have that if a single *cause is present ($|T_c| = 1$), then the effect is present, unless the cause is inhibited, so*

$$\gamma(v_0 \mid \neg v_1 \wedge \ldots \wedge \neg v_{i-1} \wedge v_i \wedge \neg v_{i+1} \wedge \ldots \wedge \neg v_m) = 1 - q_i$$

where $i = 1, \ldots, m$ and q_i is the probability that cause V_i is inhibited.

Consider an arbitrary configuration c for $V_1,\ldots,V_m$ with at least two *causes present ($|T_c| \geq 2$). Now the effect should occur only if all but one of the present causes are inhibited. Explain in detail how you would compute* $\Pr(v_0 \mid c)$ *for a noisy-XOR gate.*

* Exercise 5.10

a. *What is the rationale behind indirect probability elicitation methods such as probability wheels, betting models and lottery models?*

b. *What is a general drawback of all three methods?*

c. *What is the main difference between a betting and a lottery model? Give drawbacks of these specific models.*

d. *What is the rationale behind the tailored tool for assessing probabilities for a Bayesian network?*

* Exercise 5.11

Consider a Bayesian network $\mathcal{B} = (G, \Gamma)$, with acyclic digraph $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$. Let $\boldsymbol{V}_G = \{X_1, X_2, X_3, Y\}$ correspond to a set of binary-valued variables, with value assignments $x_i, \neg x_i$, denoting X_i = true and X_i = false, $i = 1, 2, 3$, respectively. Similarly, $y, \neg y$ denote Y = true and Y = false, respectively. In addition, let $\boldsymbol{A}_G = \{X_1 \rightarrow Y, X_2 \rightarrow Y, X_3 \rightarrow Y\}$, i.e. X_1, X_2 and X_3 are direct parents of Y in the graph.

Suppose that Y is defined as the disjunction of its three parents X_1, X_2 and X_3. (Note: here a true logical disjunction is meant; this has nothing to do with a disjunctive interaction/ noisy-or!)

a. *Specify the complete assessment function γ_Y for Y, for example in a CPT (conditional probability table):*

X_1	X_2	X_3	$\gamma_Y(Y \mid X_1 \wedge X_2 \wedge X_3)$	
			$\neg y$	y
false	*false*	*false*	...	...
...	...	...	...	...

One of the modelling techniques related to the structure of a Bayesian network is referred to as parent divorcing. *For network $\mathcal{B}$, parent divorcing would amount to introducing an intermediate variable H between Y and a* true subset $\boldsymbol{X}$ *of its parents X_1, X_2 and X_3. As a result, variables $\boldsymbol{X}$ are 'divorced' from (and no longer co-parents with) the remaining parents of Y.*

b. *Perform parent divorcing on network $\mathcal{B}$, in the following two steps:*

   b1. *select a set $\boldsymbol{X} \subset \pi_G(Y)$ with two or more variables to divorce from the remaining parents $\pi_G(Y) \setminus \boldsymbol{X}$ of Y and draw the digraph G' with $\boldsymbol{V}_{G'} = \boldsymbol{V}_G \cup \{H\}$ that results after parent divorcing;*

   b2. *specify the new assessment functions γ_H and γ_Y for the new intermediate, binary-valued variable H, and for variable Y, respectively. Explain your answers.*

c. *Answer the following questions concerning the described technique of parent divorcing:*

   c1. *What is the intended purpose or benefit of parent divorcing?*

   c2. *When would parent divorcing be unadvisable?*

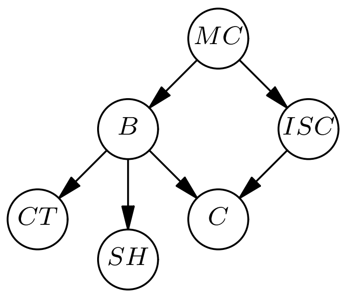
Chapter 6

Bringing Bayesian Networks into Practice

In various domains of application, ranging from medicine to meteorology, decision-support systems are being developed that build upon a Bayesian network for their knowledge representation. The previous chapter has dealt with the construction of such a Bayesian network. In this chapter, we discuss two issues that are concerned with enabling actual use of a Bayesian network in practice. Before a Bayesian network is introduced into practice, it should be analysed whether the network behaves as expected, and produces acceptable output. There are very few standard validity tests for Bayesian networks, especially for expert-elicited networks. Existing tests or analyses focus on the graphical structure (d-separation analyses), or on concepts such as 'sensitivity to findings' and 'sensitivity to model parameters' [Jensen & Nielsen, 2007]. Recently, a validation framework was introduced that suggests to include more than above-mentioned criteria for reliability and predictive validity [Pitchforth & Mengersen, 2013]. In Section 6.1, we will discuss sensitivity analysis as a general method for studying a Bayesian network's robustness. In Section 6.2, we discuss the evaluation of the practical value of a Bayesian network, by providing some measures for establishing a network's quality.

As experience with applying the Bayesian network framework increased, it became apparent that, although the framework offers many advantages over earlier approaches to automated reasoning with uncertainty, it lacks with regard to *intelligent control over reasoning.* A Bayesian network provides for computing probabilities only, whereas in practice often decisions, such as what diagnostic tests to perform (known as test-selection, or adaptive testing) or what therapy to instil, are to be based on the computed probabilities. The provision of control over reasoning is generally considered one of the main contributions of artificial intelligence research to automated reasoning: knowledge-based systems thank their success to a large extent to their ability to apply specialised knowledge for pruning search spaces and for selectively gathering evidence. In Section 6.3, we describe a problem-solving architecture that allows for automated control over reasoning with Bayesian networks.

A crucial factor in bringing Bayesian networks into practice is user-acceptance. For a user to accept a system built on the network, it should not only be proven valid and reliable, but the system should also be able to explain and motivate the outcomes to the user. Section 6.4 very briefly addresses explanation of Bayesian networks: a topic which has received too little attention over the past decades, but will hopefully profit from current interests in *explainable AI.*



$\gamma(mc) =$	0.20	$\gamma(isc \mid mc) =$	0.80
		$\gamma(isc \mid \neg mc) =$	0.20
$\gamma(b \mid mc) =$	0.20		
$\gamma(b \mid \neg mc) =$	0.05	$\gamma(ct \mid b) =$	0.95
		$\gamma(ct \mid \neg b) =$	0.10
$\gamma(c \mid b, isc) =$	0.80		
$\gamma(c \mid \neg b, isc) =$	0.80	$\gamma(sh \mid b) =$	0.80
$\gamma(c \mid b, \neg isc) =$	0.80	$\gamma(sh \mid \neg b) =$	0.60
$\gamma(c \mid \neg b, \neg isc) =$	0.05		

Figure 6.1: An example Bayesian network

6.1 Sensitivity Analysis

The reliability of the output of a Bayesian network can be investigated by studying its robustness. Robustness pertains to the extent to which the network's conditional probabilities influence the output when deviations from the specified assessments are assumed. In a medical application, for example, erroneous diagnoses or non-optimal treatment recommendations may result from building upon inaccurate assessments. For gaining detailed insight in output robustness, a Bayesian network can be subjected to a model-parameter sensitivity analysis. In the previous chapter, sensitivity analysis was mentioned as part of a probability elicitation procedure. Sensitivity analysis, in addition, can be used as a technique underlying model-parameter tuning, that is, changing probability assessments so that the network gives the desired output. Perhaps most importantly, sensitivity analysis can give insight into the range of probability assessments, or even evidence profiles, for which the outcome of the network is valid.

A sensitivity analysis of a mathematical model basically amounts to stepwise variation of one or more model parameters and computing the output of the model in each step. This is a very demanding process from a computational point of view. Fortunately, the output probability of interest in a Bayesian network relates to the model parameters varied by a simple mathematical function which can be computed quite efficiently from the network under study [Kjærulff & Van der Gaag, 2000, Coupé & Van der Gaag, 2002]. The form of this function heavily depends on how the variables corresponding to the model parameters of interest and output of interest are located relative to each other in the network, as well as on the evidence entered.

In this syllabus we restrict the discussion to *one-way* and *two-way* sensitivity analyses, where only one or two independent model parameters (i.e. from different conditional distributions), respectively, are varied simultaneously. Upon varying a model parameter, we have to make sure that the probabilities pertaining to the same conditional distribution from which we take the model parameter continue to sum to one. Upon varying a single model parameter, we adopt the standard assumption of *proportional scaling*: the other model parameters are co-varied such that their mutual proportional relationship is kept constant; other co-variation schemes can also be applied (see [Renooij, 2014] for different schemes and their properties). We will consider the general form of the functions, followed by some properties, criteria for selecting interesting model parameters, and examples.

Example 6.1.1 *As a running example we now consider the network in Figure 6.1, describing a piece of (fictitious and incomplete) medical information. The fragment describes the problems of metastatic cancer (denoted MC) for an arbitrary patient, with regard to the development of a brain tumour. Metastatic cancer may be detected by an increased* *level of serum calcium (ISC). The presence of a brain tumour (B) may be established from a CT scan (CT). Severe headaches (SH) are indicative of the presence of a brain tumour. Both a brain tumour and an increased level of serum calcium are likely to ultimately cause a patient to fall into a coma (C). The strengths of these dependences are described by the conditional probabilities. The probabilities specified for the variable ISC, for example, express that knowing whether or not metastatic cancer is present has a considerable influence on the probability of an increased level of serum calcium in an arbitrary patient. On the other hand, severe headaches are expressed as quite common in both patients with and without a brain tumour. Severe headaches thus have a low predictive value for a brain tumour. From the conditional probabilities specified for vertex C, we see that in the absence of both a brain tumour and an increased level of serum calcium, there is only a very small probability of a patient falling into a coma. The presence of either one of these causes in an arbitrary patient, however, suffices to render the probability of this patient falling into a coma in the near future quite high. Note that the two causes do not contribute to this probability independently: if one of the causes is present, then the presence of the other cause has no further influence on the probability of a patient falling into a coma. The two causes are said to exhibit a (negative)* synergistic influence *on their common effect.*

6.1.1 What to Analyse?

A sensitivity analysis with respect to a *prior* output probability of interest allows for assessing the quality and robustness of a Bayesian network in its reflecting a prior probability distribution for the domain of application. In the presence of case-specific observations, however, a Bayesian network may show very different sensitivities. To reveal these sensitivities, a sensitivity analysis may be performed with respect to a *posterior* output probability, conditioned on case-specific evidence. Such an analysis allows for validating the network's behaviour for specific cases or profiles, for example, profiles of typical patient populations in a medical application.

It is clear that a full-blown sensitivity analysis quickly becomes infeasible as it requires analysing the relation between each model parameter, or combination of model parameters, in the network, and every possible output probability of interest. The number of different output probabilities is exponential in the number of variables in the network. Fortunately, not every theoretically possible output probability will be of interest in the domain of application. In addition, for a given probability of interest, not every model parameter has to be varied. In fact, only model parameters pertaining to variables in the *sensitivity set* for a variable of interest need to be included in the sensitivity analysis. Given evidence $\boldsymbol{e}$ for a set of variables $\boldsymbol{E}$, the sensitivity set for a variable of interest A contains all variables whose model parameters may affect the posterior probability distribution for A upon variation. More precisely, the sensitivity set is the set of variables $V \in \boldsymbol{V}$ for which *none* of the following holds:

- $V \notin \boldsymbol{\rho}^*_G(A)$ and $\boldsymbol{\sigma}^*_G(V) \cap \boldsymbol{E} = \emptyset$;
- $V \in \boldsymbol{\rho}^*_G(A)$ and $\langle \{V\} \cup \boldsymbol{\rho}_G(V) \,|\, \boldsymbol{E} \,|\, \{A\} \rangle^d_G$;
- $V \notin \boldsymbol{\rho}^*_G(A)$, $\langle \{V\} \cup \boldsymbol{\rho}_G(V) \,|\, \boldsymbol{E} \,|\, \{A\} \rangle^d_G$ and $\boldsymbol{\sigma}^*_G(V) \cap \boldsymbol{E} \neq \emptyset$;

A sensitivity set clearly depends to a large extent on the case-specific observations that have been entered into the network. In our example Bayesian network, for instance, once the presence or absence of a metastatic cancer has been established in a patient, varying the probability assessments for the variable *ISC* can no longer influence $\Pr(b)$. The sensitivity set can be readily identified using d-separation properties of the network's graph extended with auxiliary nodes (more details can be found in [Coupé & Van der Gaag, 2002]) or by applying the Bayes-Ball algorithm [Shachter, 1998] to the original graph (details are given in [Meekes *et al.*, 2015]).

6.1.2 One-way Sensitivity Analysis

For a Bayesian network, sensitivity analysis basically amounts to establishing, for each of the network's conditional probabilities, the *sensitivity function* that expresses a probability of interest in terms of the model parameter under study.

In the sequel, we denote the probability of interest by $\Pr(A = a \mid \boldsymbol{e})$, or $\Pr(a \mid \boldsymbol{e})$ for short, where $\boldsymbol{e}$ denotes the available evidence. The network's parameters are denoted by $x = \gamma(b_i \mid \boldsymbol{\rho})$, where b_i is a value of a variable B and $\boldsymbol{\rho}$ is a combination of values for the predecessors of B. We use $f_{\Pr(a|\boldsymbol{e})}(x)$ to denote the function that expresses the probability $\Pr(a \mid \boldsymbol{e})$ in terms of model parameter x; we often omit the subscript for f, as long as ambiguity cannot occur.

Defining the sensitivity function

Under the assumption of proportional scaling, any sensitivity function is a quotient of two functions that are (multi-)linear in the model parameters under study. The numerator of the quotient describes the probability $\Pr(a \wedge \boldsymbol{e})$ as a function of the model parameters and the denominator describes $\Pr(\boldsymbol{e})$ as a function of the model parameters. More formally, in a one-way analysis, the function takes the form

$$f(x) = \frac{c_1 \cdot x + c_2}{c_3 \cdot x + c_4}$$

where the constants c_j, $j = 1, \ldots, 4$, are built from the assessments for the numerical model parameters that are not being varied.

If no evidence is entered into a Bayesian network, then the prior probability of interest relates linearly to any network parameter. Moreover, if the model parameter under study pertains to a variable that is an ancestor of the variable of interest in the network's qualitative part, and the network parameter's variable has no observed descendants, then the sensitivity function reduces to a linear function as well ($c_3 = 0$). For any model parameter associated with a variable outside the sensitivity set of the variable of interest, the sensitivity function is constant. We conclude that a sensitivity function is either a *linear* function or a fragment of a *rectangular hyperbola*:

$$f(x) = \frac{r}{x - s} + t, \text{ with } s = -\frac{c_4}{c_3}, \quad t = \frac{c_1}{c_3}, \quad \text{and} \quad r = \frac{c_2}{c_3} + s \cdot t$$

A rectangular hyperbola has two branches and two asymptotes. Figure 6.2 illustrates the locations of the possible hyperbola branches relative to the two asymptotes. For $r < 0$, the branches lie in the second (II) and fourth (IV) quadrants relative to the asymptotes $x = s$ and $f(x) = t$; for $r > 0$, the branches are found in the first (I) and third (III) quadrants. Since any sensitivity function is continuous for $x \in [0, 1]$, a hyperbolic sensitivity function is actually a fragment of one of the four possible hyperbola branches.

Computing the sensitivity function

There basically exist three methods for computing the constants of a sensitivity function. In the first method, the constants are determined by computing from the network the probability of interest for up to three values for the model parameter under study, and subsequently solving the resulting system of linear equations [Coupé & Van der Gaag, 2002]; for the network computations, any of the standard propagation algorithms can be used. A

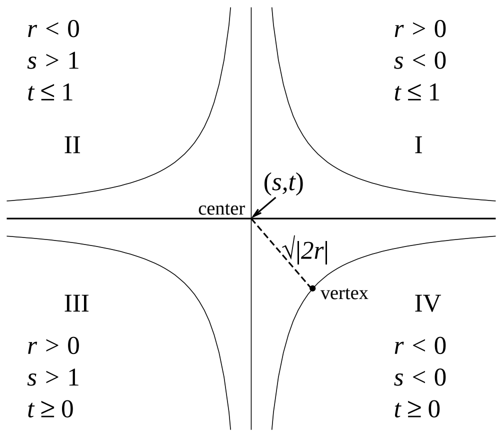


Figure 6.2: Hyperbolas and their constants (the constraints on s and t are specific for sensitivity functions)

tailored version of the propagation algorithm that builds upon junction trees can establish the constants of a sensitivity function more efficiently [Kjærulff & Van der Gaag, 2000] by basically exploiting the fact that the constants can be expressed as sums of multiplications of network parameters. Both these approaches are illustrated with the example below. A third approach, the *differential approach*, is based on the observation that the constants of the linear expressions in the numerator and denominator of the sensitivity function can be obtained from the first derivatives of these expressions [Darwiche, 2000].

Example 6.1.2 We illustrate performing a *one-way sensitivity analysis* of our example network. We begin by taking the prior $\Pr(c)$ for our probability of interest. We address the one-way analyses with respect to the model parameters $\gamma(b \mid mc)$, $\gamma(isc \mid mc)$, and $\gamma(isc \mid \neg mc)$, respectively. The results of these three prior analyses are shown in Figure 6.3. The figure displays, for example, the probability of interest $\Pr(c)$ as a function of the network parameter $x = \gamma(isc \mid \neg mc)$, that is,

$$f_{\Pr(c)}(x) = 0.57 \cdot x + 0.21$$

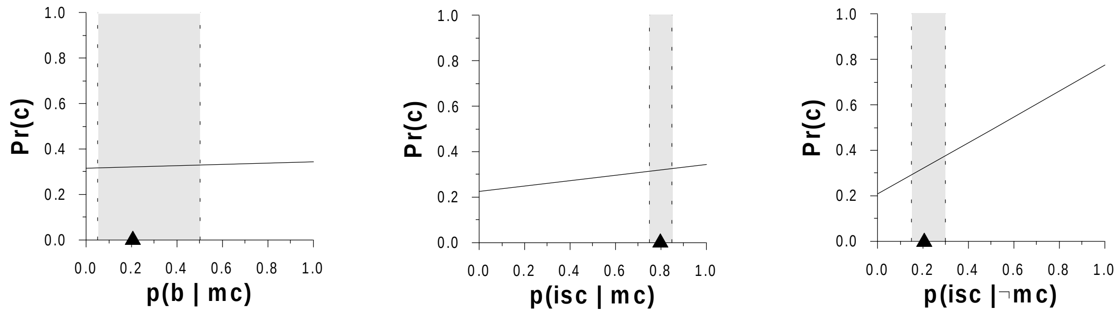


Figure 6.3: A one-way sensitivity analysis of the example Bayesian network for output probability $\Pr(c)$ and three different model parameters (note that the labels on the x-axes use p rather than γ).

Analytical approach We can *analytically* derive this expression, by marginalisation, from the factorisation of the joint distribution of our example network:

$$\begin{aligned}
\Pr(c) &= \sum_{c_{MC}}\sum_{c_B}\sum_{c_{ISC}}\sum_{c_{CT}}\sum_{c_{SH}} \Pr(c_{MC}, c_B, c_{ISC}, c, c_{CT}, c_{SH}) \\
&= \sum_{c_{MC}}\sum_{c_B}\sum_{c_{ISC}}\sum_{c_{CT}}\sum_{c_{SH}} \gamma(c \mid c_B, c_{ISC}) \cdot \gamma(c_B \mid c_{MC}) \cdot \gamma(c_{ISC} \mid c_{MC}) \cdot \gamma(c_{MC}) \cdot \\
&\quad \cdot \gamma(c_{CT} \mid c_B) \cdot \gamma(c_{SH} \mid c_B) \\
&= \sum_{c_{MC}}\sum_{c_B}\sum_{c_{ISC}} \gamma(c \mid c_B, c_{ISC}) \cdot \gamma(c_B \mid c_{MC}) \cdot \gamma(c_{ISC} \mid c_{MC}) \cdot \gamma(c_{MC}) \cdot \\
&\quad \cdot \sum_{c_{CT}} \gamma(c_{CT} \mid c_B) \cdot \sum_{c_{SH}} \gamma(c_{SH} \mid c_B)
\end{aligned}$$

Note that the latter two factors each sum to 1 and can therefore be omitted. We now write out the summations over c_{ISC} and c_{MC} to reveal our x.

$$\begin{aligned}
\Pr(c) &= \sum_{c_{MC}}\sum_{c_B}\sum_{c_{ISC}} \gamma(c \mid c_B, c_{ISC}) \cdot \gamma(c_B \mid c_{MC}) \cdot \gamma(c_{ISC} \mid c_{MC}) \cdot \gamma(c_{MC}) \\
&= \sum_{c_{MC}}\sum_{c_B} \gamma(c \mid c_B, isc) \cdot \gamma(c_B \mid c_{MC}) \cdot \gamma(isc \mid c_{MC}) \cdot \gamma(c_{MC}) \\
&\quad + \sum_{c_{MC}}\sum_{c_B} \gamma(c \mid c_B, \neg isc) \cdot \gamma(c_B \mid c_{MC}) \cdot \gamma(\neg isc \mid c_{MC}) \cdot \gamma(c_{MC}) \\
&= \sum_{c_B} \gamma(c \mid c_B, isc) \cdot \gamma(c_B \mid mc) \cdot \gamma(isc \mid mc) \cdot \gamma(mc) \\
&\quad + \sum_{c_B} \gamma(c \mid c_B, isc) \cdot \gamma(c_B \mid \neg mc) \cdot \gamma(isc \mid \neg mc) \cdot \gamma(\neg mc) \\
&\quad + \sum_{c_B} \gamma(c \mid c_B, \neg isc) \cdot \gamma(c_B \mid mc) \cdot \gamma(\neg isc \mid mc) \cdot \gamma(mc) \\
&\quad + \sum_{c_B} \gamma(c \mid c_B, \neg isc) \cdot \gamma(c_B \mid \neg mc) \cdot \gamma(\neg isc \mid \neg mc) \cdot \gamma(\neg mc) \\
&= \sum_{c_B} \gamma(c \mid c_B, isc) \cdot \gamma(c_B \mid mc) \cdot \gamma(isc \mid mc) \cdot \gamma(mc) \\
&\quad + \sum_{c_B} \gamma(c \mid c_B, isc) \cdot \gamma(c_B \mid \neg mc) \cdot x \cdot \gamma(\neg mc) \\
&\quad + \sum_{c_B} \gamma(c \mid c_B, \neg isc) \cdot \gamma(c_B \mid mc) \cdot \gamma(\neg isc \mid mc) \cdot \gamma(mc) \\
&\quad + \sum_{c_B} \gamma(c \mid c_B, \neg isc) \cdot \gamma(c_B \mid \neg mc) \cdot (1 - x) \cdot \gamma(\neg mc)
\end{aligned}$$

We observe that $f_{\Pr(c)}(x) = c_1 \cdot x + c_2$ with constants c_1 and c_2 defined by

$$\begin{aligned}
c_1 &= \sum_{c_B} \big(\gamma(c \mid c_B, isc) - \gamma(c \mid c_B, \neg isc)\big) \cdot \gamma(c_B \mid \neg mc) \cdot \gamma(\neg mc) \\
c_2 &= \sum_{c_B} \gamma(c \mid c_B, isc) \cdot \gamma(c_B \mid mc) \cdot \gamma(isc \mid mc) \cdot \gamma(mc) \\
&\quad + \sum_{c_B} \gamma(c \mid c_B, \neg isc) \cdot \gamma(c_B \mid mc) \cdot \gamma(\neg isc \mid mc) \cdot \gamma(mc) \\
&\quad + \sum_{c_B} \gamma(c \mid c_B, \neg isc) \cdot \gamma(c_B \mid \neg mc) \cdot \gamma(\neg mc)
\end{aligned}$$

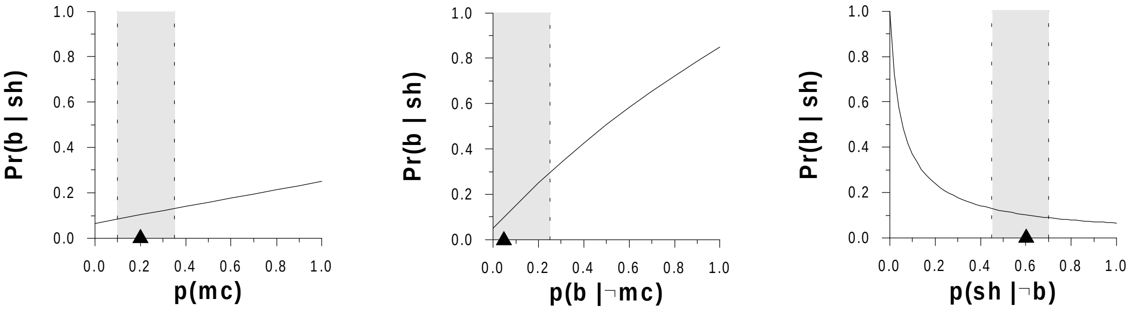


Figure 6.4: A one-way sensitivity analysis of the example Bayesian network for output probability $\Pr(b \mid sh)$ and three different model parameters (note that the labels on the x-axes use p rather than γ).

From this analytical expression and the network specification you can now straightforwardly compute the constants $c_1 \approx 0.57$ and $c_2 \approx 0.21$. Designing an algorithm that does exactly these computations exists, but is not straightforward [Kjærulff & Van der Gaag, 2000].

Approach using standard inference An *alternative approach* to the analytical approach is to use a standard inference algorithm to compute $\Pr(c)$ from the network for two different values of model parameter x, e.g. x_1 and x_2 (note that this requires actually changing the assessment function γ_{ISC}!), resulting in two different output probabilities p_1 and p_2; the constants $c_1 \approx 0.57$ and $c_2 \approx 0.21$ then follow by solving the following system of linear equations:

$$\begin{aligned} p_1 &= c_1 \cdot x_1 + c_2 \\ p_2 &= c_1 \cdot x_2 + c_2 \end{aligned}$$

Note that you choose any values for x_1 and x_2 in the zero-one interval, and subsequently compute the probabilities p_1 and p_2 from the network. The above system therefore contains two expressions with two unknowns, c_1 and c_2, and can be uniquely solved.

Next, we take for the probability of interest the *posterior* probability $\Pr(b \mid sh)$. By doing so, we assess the robustness of the *diagnosis* of a brain tumour for an arbitrary patient with a primary tumour who is suffering from severe headaches. We address the one-way analyses with respect to the model parameters $\gamma(mc)$, $\gamma(b \mid \neg mc)$, and $\gamma(sh \mid \neg b)$, respectively. The results of these posterior analyses are shown in Figure 6.4. The figure shows, for example, the probability of interest $\Pr(b \mid sh)$ as a function of the network parameter $x = \gamma(b \mid \neg mc)$, that is,

$$f_{\Pr(b|sh)}(x) \quad = \quad \frac{f_{\Pr(b \wedge sh)}(x)}{f_{\Pr(sh)}(x)} = \frac{0.64 \cdot x + 0.032}{0.16 \cdot x + 0.608} = \frac{4 \cdot x + 0.20}{x + 3.80}$$

Note that, in contrast with the prior analyses discussed before, the analyses for the posterior probability of interest reveal a non-linear relationship between the probability assessment that is being varied and the probability of interest. □

Selecting Model Parameters Deserving Attention

Sensitivity analyses for a large number of model parameters and several outputs of interest will typically result in a huge number of sensitivity functions that are to be examined. A number of criteria can be used to decide whether or not the effect of a network parameter can be considerable enough to warrant further consideration, either for refining its assessment as discussed in Chapter 5, or for determining the implications of the analysis for the employability of the network in the domain of application. Here we will briefly discuss the following selection criteria: *absolute effect*, *plausible effect*, *sensitivity value*, *vertex proximity*, and *admissible deviation.*

Absolute effect The absolute effect of changing a model parameter on an output probability is simply the absolute difference $|f(0) - f(1)|$.

Example 6.1.3 We continue our example analysis. Figure 6.3(a) shows that varying the assessment for the probability $\gamma(b \mid mc)$ from 0 to 1 has a negligible effect on the probability of interest $\Pr(c)$: the prior probability of a patient falling into a coma within the next three years increases from 0.31 to 0.34, approximately. Figure 6.3(b) shows that varying the initial assessment for the probability $\gamma(isc \mid mc)$ has a somewhat stronger effect on the probability of interest: $\Pr(c)$ now ranges from 0.22 to 0.34. From Figure 6.3(c), to conclude, it is seen that varying the assessment for the probability $\gamma(isc \mid \neg mc)$ has an even stronger effect on $\Pr(c)$: the prior probability of a coma ranges from 0.21 to 0.78. Note that the three analyses reveal the linear relationship between the probability assessment that is being varied and the probability of interest. □

Plausible effect So far we have treated the probability assessments of our Bayesian network as exact point probabilities. As for most applications, however, the initially obtained assessments are quite uncertain. If this uncertainty is captured by supplementing each probability assessment with a plausible interval that defines a range of values in which the 'true' probability lies with reasonable certainty, then we can select model parameters of interest based upon their *plausible effect* on the probability of interest. The plausible effect is now defined as the absolute effect within the plausible interval; the plausible effect is bounded by the absolute effect.

Example 6.1.4 In the prior analyses, for the three probability assessments under study in our example network, the plausible intervals are indicated in Figure 6.3 by shading. The figure shows that plausible variation of the model parameter $\gamma(isc \mid \neg mc)$ has the strongest effect on the probability of interest $\Pr(c)$. Varying the model parameters $\gamma(b \mid mc)$ and $\gamma(isc \mid mc)$, respectively, within their plausible intervals results in a rather small effect on the probability of interest. We recall from Figure 6.3 that the effect on $\Pr(c)$ of varying the assessment for $\gamma(b \mid mc)$ from 0 to 1 is smaller than the effect of varying $\gamma(isc \mid mc)$ from 0 to 1. By taking the plausible intervals into consideration, however, variation of the assessment for $\gamma(b \mid mc)$ has the stronger plausible effect. Especially since the plausible interval for this assessment is quite large, for example further elicitation efforts may better be directed at the probability $\gamma(b \mid mc)$ than at the probability $\gamma(isc \mid mc)$. □

Sensitivity value The *sensitivity value* of a model parameter x with respect to a probability of interest is defined as $|\frac{\partial f}{\partial x}(x_0)|$, the absolute value of the first derivative of the sensitivity function at the original value x_0 of the network parameter. The sensitivity value thus captures the effect of infinitely small shifts in the model parameter on the probability of interest.

Example 6.1.5 For the sensitivity function describing the posterior probability $\Pr(b \mid sh)$ as a function of the model parameter $x = \gamma(b \mid \neg mc)$ (see Figure 6.4(b)), we find for example that

$$f'(x) = \frac{0.384}{(0.16 \cdot x + 0.608)^2};$$

the model parameter has an original value of 0.05, so the sensitivity value for this model parameter is $|f'(0.05)| = 1.01$. Alternatively, if we consider the effect of varying network parameter $x = \gamma(sh \mid \neg b)$ on our posterior probability of interest (Figure 6.4(c)), then the sensitivity value equals

$$\left|\frac{-0.059}{(0.92 \cdot 0.60 + 0.064)^2})\right| = 0.155.$$

□

Vertex proximity The problem of using the sensitivity value as a measure of robustness, is that it often gives insight only in the effect of very small changes to the model parameter. The effects of larger model-parameter shifts are of course captured by the absolute and plausible effects, but can also be studied by examining the vertex proximity.

The *vertex* of a hyperbola branch is the point $(x_v, f(x_v))$ where $|f'(x_v) = 1|$. The proximity of a model parameter's original value x_0 to the x-value of the hyperbola's vertex is an indication of possible sensitivity of the output of interest to variation of the model parameter. The vertex-proximity can be easily computed from the constants of the sensitivity function:

$$x_v = \begin{cases} s + \sqrt{|r|}, & \text{if } s < 0 \\ s - \sqrt{|r|}, & \text{if } s > 1 \end{cases}$$

Example 6.1.6 Again consider our example sensitivity functions in Figure 6.4. For the hyperbola branch describing the output probability as a function of model parameter $\gamma(b \mid \neg mc)$ with original value 0.05, we have that $x_v = -3.8 + \sqrt{15} = 0.07$; the vertex of the function is therefore very close to the model parameter's original value, indicating that non-infinitesimal variation of the model parameter could possibly have large effects on the output probability of interest. For model parameter $\gamma(sh \mid \neg b)$, on the other hand, the x-value of the vertex is found around 0.19, which can be considered quite distant from the model parameter's original value of 0.60. If the original value of the latter network parameter had been 0.20, however, then its sensitivity value (0.96) would perhaps deem the model parameter irrelevant for further inspection, whereas the vertex-proximity would warn us for possibly significant effects of variation. □

Admissible deviation An *admissible deviation* for a variable of interest and a given model parameter, is a pair of real numbers (α, β) that describe the shifts to smaller values and to larger values, respectively, that are allowed in the model parameter without inducing a change in the most likely value of the variable of interest. For a model parameter with an original value of x_0, the admissible deviation (α, β) thus indicates that the model parameter can be safely varied within the interval $[x_0 - \alpha, x_0 + \beta]$. To express that the network parameter can be varied as far as the bounds of the probability interval, the symbol ∞ is used.

For establishing an admissible deviation, the intersections of a sensitivity function relating one value of the output variable of interest to a network parameter, and those pertaining to the other values of the output variable, are computed. More specifically, the admissible deviation is established by computing the x-coordinates of the points where the sensitivity function $f_{\Pr(a_i|\mathbf{e})}(x)$ for some variable of interest A, intersects with the sensitivity functions $f_{\Pr(a_j|\mathbf{e})}(x)$, $j \neq i$. We note that if the variable of interest has only two values, then the two sensitivity functions always intersect for $f(x) = 0.5$. For so-called *threshold decision making* (see Section 6.3.1), instead of considering a change in most likely value, we can determine when some outcome probability of interest becomes

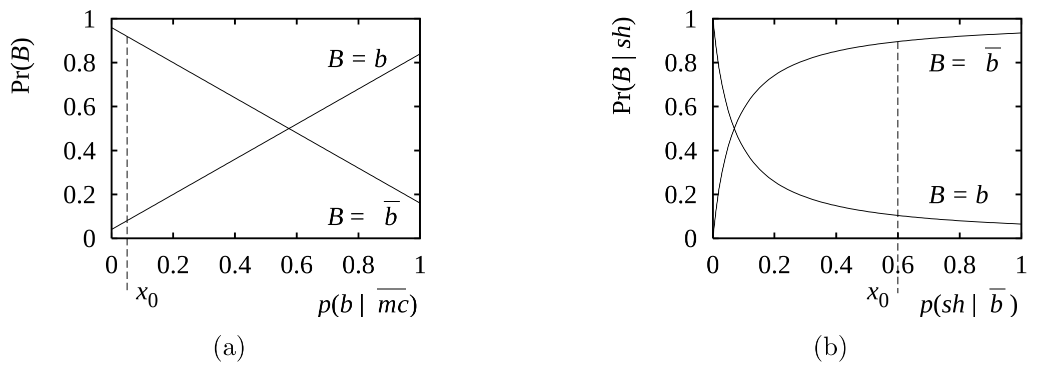


Figure 6.5: The sensitivity functions for the two possible values of the variable of interest B as a function of (a) network parameter $\gamma(b \mid \neg mc)$, and (b) network parameter $\gamma(sh \mid \neg b\,)$, given a severe headache.

smaller or larger than some pre-specified threshold; this basically amounts to establishing the intersections of a sensitivity function with the constant functions associated with the values of the threshold probabilities.

Example 6.1.7 We illustrate investigating the robustness of decisions by computing admissible deviations for our example Bayesian network. First, consider an arbitrary patient. The effects of varying model parameter $x = \gamma(b \mid \neg mc)$ with original value $x_0 = 0.05$ on the probabilities of disease $\Pr(B)$ are shown in Figure 6.5(a) and are described by

$$f_{\Pr(b)}(x) = 0.80 \cdot x + 0.04, \quad \text{and} \quad f_{\Pr(\neg b\,)}(x) = -0.80 \cdot x + 0.96.$$

The sensitivity functions intersect for $x = 0.575$, resulting in an admissible deviation of $(\infty, 0.525)$. The patient is most likely to not suffer from a brain tumour and this diagnosis is quite robust to variation of the model parameter under study. Now consider a patient with severe headaches. The effects of varying network parameter $x = \gamma(sh \mid \neg b\,)$ with original value $x_0 = 0.60$ on the posterior probabilities of disease $\Pr(B \mid sh)$ are shown in Figure 6.5(b) and are described by

$$f_{\Pr(b|sh)}(x) = \frac{0.064}{0.92 \cdot x + 0.064}, \quad \text{and} \quad f_{\Pr(\neg b\,|sh)}(x) = \frac{0.92 \cdot x}{0.92 \cdot x + 0.064}.$$

The sensitivity functions intersect for $x = 0.0696$, resulting in an admissible deviation of $(0.53, \infty)$. We conclude that the patient is most likely not to suffer from a brain tumour and this diagnosis again is quite robust to variation of the model parameter under study. $\square$

6.1.3 Two-way Sensitivity Analysis

We now address a *two-way sensitivity analysis* of a Bayesian network. In a two-way sensitivity analysis, two probability assessments are varied simultaneously to reveal their joint effect on a probability of interest. Recall that under the assumption of proportional co-variation, any sensitivity function is a quotient of two functions that are (multi-)linear in the model parameters under study. In a two-way analysis, the function is bi-linear and takes the general form

$$f(x, y) = \frac{c_1 \cdot x \cdot y + c_2 \cdot x + c_3 \cdot y + c_4}{c_5 \cdot x \cdot y + c_6 \cdot x + c_7 \cdot y + c_8}$$

where the constants c_j, $j = 1, \ldots, 8$, are again built from the assessments for the numerical network parameters that are not being varied.

As remarked before, two model parameters can have a synergistic effect on the probability of interest; this means that the joint effect of varying the two model parameters is different from the sum of their individual effects. Not every pair of varied model parameters will have such an interaction effect on a probability of interest, however. For example, any two model parameters that pertain to *incompatible* probabilities, in the sense of specifying complementary values for the same variable, will not interact. The function expressing the probability of interest in terms of two such assessments will lack a product term. A two-way sensitivity analysis involving assessments for incompatible probabilities does not reveal any unanticipated effects on a probability of interest beyond the effects shown by one-way sensitivity analyses for the two assessments separately. Any such pair of assessments can therefore be excluded from the analysis.

Two-way sensitivity analysis in Bayesian networks has received far less attention by researchers than one-way analysis. No details are as yet known about the possible shapes of the two-way functions. It is obvious though that if either one of the model parameters is fixed to an arbitrary value, then the two-way function degenerates to a one-way function in the other model parameter, which is either monotonically increasing or monotonically decreasing; as a result, the minimum and maximum function values are still found for the extreme network parameter values 0 and 1. Research, in addition, has not addressed selection criteria. Here we will therefore limit our discussion to the selection criteria that trivially apply.

Selecting Model Parameters Deserving Attention

For selecting network parameters from a two-way analysis that deserve further attention, we can use the *absolute effect*, the *plausible effect*, and the *sensitivity value* as before. In addition, we can use *contour distance* as a selection criterion.

Absolute and plausible effect The absolute effect of network parameters x and y on a probability of interest is captured by the largest absolute difference found between the function values of the sensitivity function in the 'corners' $(0, 0)$, $(0, 1)$, $(1, 0)$, and $(1, 1)$ of the domain: $\max\{f(i, j) - f(k, l) \mid i, j, k, l \in \{0, 1\}\}$. The plausible effect is defined analogously where the 'corners' of the domain are given by the plausible intervals for the two parameters.

Sensitivity value The sensitivity value of two model parameters x and y with respect to a probability of interest can be defined in terms of a directional derivative (for details, see [Bolt & Renooij, 2014]). Note that for a surface in 3D, the gradient in a certain point will typically depend on the direction under consideration, so we in fact have an infinite number of sensitivity values; the maximal gradient in a point, however, equals the length of the gradient vector in that point. The maximum 2-way sensitivity value at the original values x_0 and y_0 of the model parameters therefore equals

$$|\nabla f(x_0, y_0)| = \sqrt{\Big(\frac{\partial f}{\partial x}(x_0, y_0)\Big)^2 + \Big(\frac{\partial f}{\partial y}(x_0, y_0)\Big)^2}$$

Contour distance Two-way sensitivity functions are most easily interpreted using *contour plots*, in which contour lines connect the combinations of values for the two network parameters that result in the same value for the probability of interest. The distance between two contour lines indicates the variation necessary in the two assessments to shift the probability of interest from one contour line to another. If the contour lines are very close to one another, then a small variation in the model parameters under study suffices to have a strong effect on the probability of interest; if, in contrast, the contour lines are further apart, then the probability of interest is not very sensitive to variation of the two assessments.

Example 6.1.8 We illustrate performing a two-way sensitivity analysis for our example network. For our probability of interest, we once again take the prior probability $\Pr(c)$. We first address an analysis with respect to the model parameters $x = \gamma(b \mid mc)$ and $y = \gamma(isc \mid mc)$. The corresponding sensitivity function equals

$$\Pr(c) = -0.15 \cdot x \cdot y + 0.15 \cdot x + 0.15 \cdot y + 0.194$$

From this function it is readily seen that the two probability assessments under study upon variation have a negative interaction effect on the probability of interest. The function is depicted in Figure 6.6a. We observe from the figure that the distances between the contour lines differ, indicating that varying the model parameters $\gamma(b \mid mc)$ and $\gamma(isc \mid mc)$ simultaneously has a joint effect on the probability of interest $\Pr(c)$ beyond the effects of their separate variation; this joint effect is due to the synergistic influence of the variables B and ISC on the variable C outlined before. We further observe that the contour lines are closer to one another in the lower left part of the figure than in the upper right part. If the assessments for the network parameters $\gamma(b \mid mc)$ and $\gamma(isc \mid mc)$ are both quite small, therefore, their variation will have a stronger effect on the probability of interest than if the initial assessments have a higher value. To variation within the plausible intervals of the assessments $\gamma(b \mid mc) = 0.2$ and $\gamma(isc \mid mc) = 0.8$, as indicated by shading in Figure 6.6a, the probability of interest shows a relatively low sensitivity. We further observe that the absolute effect of their joint variation on $\Pr(c)$ is not too strong.

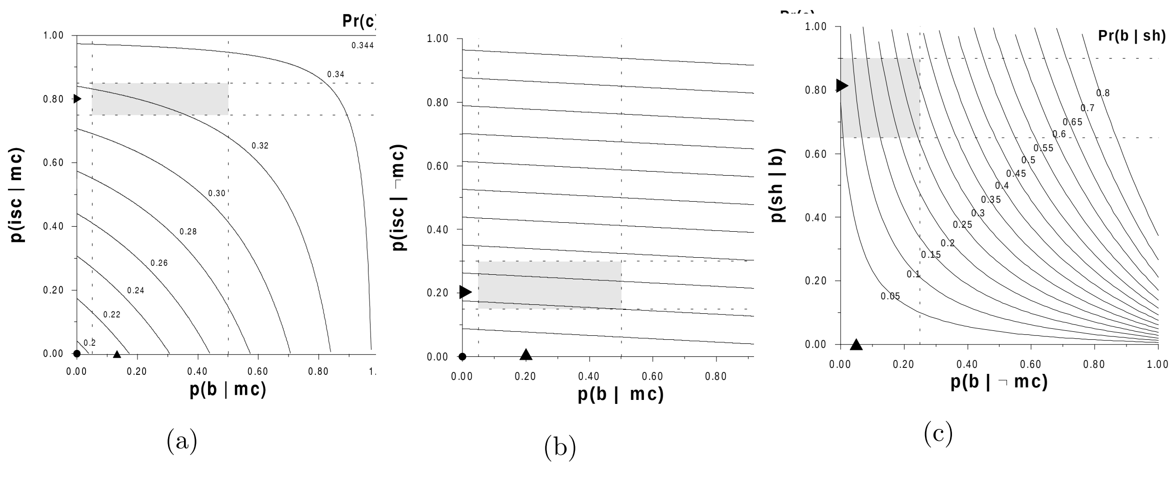


Figure 6.6: A two-say sensitivity analysis of the example Bayesian network.

Now we address an analysis pertaining to the assessments for the model parameters $\gamma(b \mid mc)$ and $\gamma(isc \mid \neg mc)$; note that the two assessments under study pertain to incompatible probabilities, since they are conditioned on two different values of the same variable. The corresponding sensitivity function equals

$$\Pr(c) = 0.03 \cdot x + 0.57 \cdot y + 0.204$$

This function is depicted in Figure 6.6b. We observe from the figure that the contour lines, once again indicating values for the probability of interest $\Pr(c)$, are equidistant. Equidistance of contour lines indicates that simultaneously varying the probability assessments under study has no joint effect on the probability of interest beyond the effects of their separate variation. The two-way analysis therefore does not provide any information in addition to the information yielded by one-way analyses for the separate assessments.

To conclude, we performed a two-way sensitivity analysis of our example network with respect to the posterior probability of interest $\Pr(b \mid sh)$. We first address the analysis of varying the assessments for the model parameters $x = \gamma(b \mid \neg mc)$ and $y = \gamma(sh \mid b)$ simultaneously. The corresponding two-way sensitivity function equals

$$\Pr(b \mid sh) = \frac{1.10005 \cdot x \cdot y - 0.00056 \cdot x + 0.0559 \cdot y - 0.00034}{x \cdot y - 0.6268 \cdot x + 0.0835 \cdot y + 0.7811}$$

In this function, the terms $-0.00056 \cdot x$ and $-0.6268 \cdot x$ pertain to the effect of variation of just the probability assessment $p(b \mid mc)$; the terms $0.0559 \cdot y$ and $0.0835 \cdot y$ pertain to the assessment $p(isc \mid mc)$. The terms $1.10005 \cdot x \cdot y$ and $x \cdot y$ with each other capture the interaction effect of the two assessments on the network's probability of interest. These terms provide information that cannot be revealed by one-way analyses with respect to the two assessments separately. The function is depicted in Figure 6.6c. Note that the contour lines are closest to one another in the lower right part of the figure, indicating a high sensitivity of the posterior probability of interest to high values for the probability $\gamma(b \mid \neg mc)$ and low values for the probability $\gamma(sh \mid b)$. For variation, within the plausible intervals of the initial assessments $\gamma(b \mid \neg mc) = 0.05$ and $\gamma(sh \mid b) = 0.8$, as indicated by shading in Figure 6.6c, however, the probability of interest is relatively stable. □

6.2 Evaluating Bayesian Networks

To establish its practical value, a real-life Bayesian network is typically subjected to an evaluation study using data from the domain of application. Such a study amounts to entering the data available for each problem case into the network and computing the most likely outcome. This outcome is then compared against a given standard of validity. The results of the study are often summarised in the percentage of correctly computed outcomes. This *percentage correct* or *accuracy* is generally taken to convey the practical value of the network. For a medical diagnostic application, for example, a percentage correct of 85% is taken to indicate that the network is likely to establish the correct diagnosis for 85 out of every 100 patients. For many applications, this percentage would convey the information that the network performs quite satisfactorily.

Unfortunately, interpretation of the percentage correct of a Bayesian network is not as straightforward as is often suggested. The percentage should be interpreted with respect to a specific data collection. Now, each data collection is likely to include errors and to reflect the biases exhibited by the experts who collected the data or as a result of considering different (sub)populations[Krak & Van der Gaag, 2014]. Moreover, the data will include the effects of random variation, especially in domains of a scientific nature. In the medical domain, for example, there is random variation in patient data, arising from biological differences between patients in the progression of pathological processes and from differences in the physicians' interpretation of symptoms and signs [Fletcher *et al.*, 1996]. When two outcomes are almost equally likely for a patient, chance determines, to at least some extent, which outcome is entered into the patient's medical record as the most likely one. Random variation may thus affect a network's percentage correct, but the extent to which it does so is not expressed by the percentage.

Bayesian networks in essence do not yield a deterministic outcome. Rather, they produce a posterior probability distribution for their outcome variable. To determine a percentage correct, therefore, an additional classification rule is required to map a probability distribution into a single value for the outcome variable of interest. The distribution computed by the Bayesian network reveals the extent of uncertainty in the outcome and reflects the network's doubt as to the most likely outcome. This, however, is not taken into account in the percentage correct. To incorporate the network's doubt in the assessment of its practical value, evaluation *scores* from the field of statistical forecasting can be used. The use and interpretation of such a score is illustrated by means of an evaluation study of a real-life Bayesian network for the staging of oesophageal cancer.

6.2.1 The Percentage Correct and its Shortcomings

With the help of two experts in gastrointestinal oncology from the Netherlands Cancer Institute, Antoni van Leeuwenhoekhuis, a Bayesian network for the staging of oesophageal cancer has been constructed. The network captures the state-of-the-art knowledge about oesophageal cancer in the *oesophagus network*; for details on the network and its construction, see [Van der Gaag *et al.*, 2002]. For studying the ability of the oesophagus network to correctly predict the stage of a patient's cancer, the medical records of 156 patients diagnosed with oesophageal cancer are available. For each patient, between 6 and 21 of the 25 symptoms and test results modelled in the network are available. For each patient, also the *stage* of his or her cancer, as established by the attending physician, is recorded. This stage can be either I, IIA, IIB, III, IVA, or IVB, in the order of advanced disease. The tumour's stage is indicative of the effects and complications to be expected from the different available therapeutic alternatives.

In general, to establish the practical value of a Bayesian network, for each case the outcome with highest posterior probability is determined from the network and compared against a given standard of validity. The results of such an evaluation study are summarised in the percentage of cases for which the network yields the correct outcome as the most likely one. To establish the practical value of the oesophagus network, for each patient all diagnostic symptoms and test results available were entered and the most likely stage for the patient's cancer was computed from the Bayesian network. The computed stage was then compared with the stage recorded in the data. The results from this comparison are shown in the matrix of Figure 6.7. The numbers on the diagonal of the matrix are the numbers of patients per stage for whom the network yields the same stage as the one recorded in the data. Taking the stages from the medical records as a standard of validity, we find that the network establishes the correct stage for 133 of the 156 patients, that is, the network has a percentage correct of 85%.

As Bayesian networks in general, the oesophagus network in essence does not produce a deterministic outcome. Rather, it yields a probability distribution for its outcome variable. More specifically, it yields, for each patient, a posterior probability distribution

		network						
		I	IIA	IIB	III	IVA	IVB	*total*
data	I	**2**	0	0	0	0	0	2
	IIA	0	**37**	0	1	0	0	38
	IIB	0	1	**0**	3	0	0	4
	III	1	10	0	**36**	0	0	47
	IVA	0	0	0	4	**35**	0	39
	IVB	0	0	0	3	0	**23**	26
	total	3	48	0	47	35	23	156

Figure 6.7: The results from the evaluation study, expressed in terms of the numbers of correctly and incorrectly staged patients.

patient 1, stage IVA	
stage I	0
stage IIA	0
stage IIB	0.0159
stage III	0.0882
stage IVA	0.8245
stage IVB	0.0714

patient 2, stage III	
stage I	0
stage IIA	0
stage IIB	0.0002
stage III	0.3616
stage IVA	0.3498
stage IVB	0.2884

patient 3, stage III	
stage I	0.0222
stage IIA	0.3753
stage IIB	0.0459
stage III	0.3714
stage IVA	0.0916
stage IVB	0.0936

Figure 6.8: The posterior probabilities of the six stages for three different patients.

over the six possible stages of his or her cancer. As an example, Figure 6.8 shows the distributions that are computed for three different real patients. Now, for some patients the computed posterior distribution clearly points to a single most likely stage. For other patients, however, the posterior distribution can reveal considerable uncertainty. It is not unlikely, therefore, that incorrect conclusions of a network can be attributed to the effect of random variation, rather than to, for example, a modelling error. In the percentage correct reported for a network, however, the distribution of uncertainty over the various different outcomes is not taken into account. For example, for the patients shown in Figure 6.8, the oesophagus network's outcome is classified simply as correct for the first two patients and as incorrect for patient 3.

6.2.2 An Evaluation Score

As mentioned previously, Bayesian networks typically yield a probability distribution for their outcome variable. While for some cases from the domain of application the computed posterior distribution will point to a single most likely outcome, it may reveal considerable uncertainty for other cases. The percentage correct as a summary of evaluation results does not take these uncertainties into account. For assessing the practical value of a network, however, not just the most likely outcome but also the posterior distribution over the various possible outcomes should be taken into consideration.

Bayesian networks basically are probabilistic *forecasters*, as they repeatedly present predictions for an outcome variable in terms of probabilities. For the oesophagus network, for example, the posterior probability distribution that is computed for a specific patient can be looked upon as a forecast for the stage of this patient's cancer. Establishing the practical value of a Bayesian network thus amounts to assessing its quality as a forecaster. The quality of a probabilistic forecaster is often expressed in terms of its *calibration*, that is, the degree to which its forecasts match the true distribution of outcomes. For the oesophagus network, more specifically, we can say that it is (empirically) *well calibrated* if, among the patients for whom the network predicts a specific stage S with probability x_S, the proportion of patients who in fact have stage S, denoted x'_S, equals x_S. The smaller the difference between x_S and x'_S, that is, the closer the network's distribution matches the true distribution, the better calibrated the network is [Dawid, 1985, DeGroot & Fienberg, 1983]. Building upon this concept of calibration, various scores for expressing the quality of a forecaster have been developed in the field of statistics.

Among the best-known evaluation scores is the *Brier score* [Panofsky & Brier, 1968]. The basic idea of this score is illustrated for our oesophagus network. For each patient i, the network yields a forecast of posterior probabilities p_{ij} over the stages $j = \text{I}, \ldots, \text{IVB}$. The Brier score B_i of this forecast is defined as

$$B_i = \sum_{j=\text{I},\ldots,\text{IVB}} (p_{ij} - s_{ij})^2$$

where $s_{ij} = 1$ if the medical record of patient i states stage j, and $s_{ij} = 0$ otherwise. If the network would yield the correct stage with certainty, that is, if the network would yield a correct deterministic forecast for the patient, then the associated Brier score would be equal to 0. If the network would yield an incorrect deterministic forecast, the score would be 2. For the forecast for a single patient, therefore, the Brier score ranges between 0 and 2, and the better the forecast, the lower the score. The Brier scores for the network's forecasts for the three patients from Figure 6.8 now are:

$$B_1 = 0.04 \qquad B_2 = 0.61 \qquad B_3 = 0.56$$

These scores reveal that the quality of the forecast for the first patient is very good, as expected. The other scores show that the forecasts for the patients 2 and 3 are of less quality. Recall that for patient 3 the forecast is unequivocal as a result of two stages being almost equally likely, among which is the correct stage. For patient 2, there is even more uncertainty in the forecast, as there are three almost equally likely stages. These observations are reflected in the associated Brier scores: the score B_3 for patient 3 indicates higher quality than the score B_2 for the second patient. While in terms of the numbers of correctly and incorrectly staged patients the forecast for patient 2 is correct and the forecast for patient 3 is incorrect, the Brier score results in a more balanced and, hence, a more insightful quality assessment. Figure 6.9 summarises the Brier scores averaged over all, correctly and incorrectly staged, patients.

		network					
		I	IIA	IIB	III	IVA	IVB
data	I	**0.21**	–	–	–	–	–
	IIA	–	**0.28**	–	1.52	–	–
	IIB	–	1.17	–	0.98	–	–
	III	1.40	0.89	–	**0.26**	–	–
	IVA	–	–	–	0.75	**0.08**	–
	IVB	–	–	–	0.87	–	**0.06**

Figure 6.9: The results from the evaluation study, expressed in terms of average Brier scores.

The quality of the oesophagus network as a forecaster can now be expressed in an overall score that is computed from the scores of the separate forecasts for our collection of patients. For n patients, the overall Brier score B is defined as

$$B = \frac{1}{n} \sum_{i=1,\ldots,n} B_i$$

It is readily seen that the overall Brier score again ranges between 0 and 2, and the better the forecaster, the lower the score. For the oesophagus network, an overall Brier score of 0.29 is found. To interpret this number, we compare the score with the overall scores obtained for two *uninformed* forecasters. The first forecaster gives a uniform probability distribution for each patient; this forecaster has an overall Brier score of 0.83. The second forecaster gives for each patient the prior distribution over the stages recorded in the data; this forecaster has an overall Brier score of 0.76 and is therefore slightly more informed than the uniform forecaster. The much lower Brier score of the oesophagus network now conveys the information that the network is quite informed and indeed builds upon its knowledge of oesophageal cancer to arrive at relatively good forecasts.

The percentage correct and the Brier score provide two ways of evaluating the practical value of a Bayesian network. The percentage correct treats outcomes as deterministic, whereas the Brier score takes the actual uncertainties into account.

6.3 A Problem-Solving Architecture

In this section, we describe a problem-solving architecture that allows for automated control over reasoning. The main purpose of exerting control over reasoning is to shape efficient and intelligent problem-solving behaviour. Exerting control involves monitoring and reflecting upon the reasoning process as it develops and taking decisions as to how it should proceed. To this end, strategic knowledge about the domain at hand is employed. As this knowledge may be non-probabilistic in nature, control over Bayesian network reasoning generally cannot be implemented in the framework in itself. The Bayesian network framework can therefore be embedded in a general *problem-solving architecture* [Van der Gaag & Wessels, 1994a].

The Bayesian network problem-solving architecture is composed of two *layers*. The first layer offers the Bayesian network formalism and a variety of associated algorithms. The algorithms in this layer are characterised by their operating on a Bayesian network directly: various algorithms for probabilistic inference are comprised in the layer as are algorithms for example for reading independences from the qualitative part of a network. This layer is called the *probabilistic layer* of the architecture. The second layer of the problem-solving architecture is designed to provide for control over reasoning — it is termed the *control layer*. This layer offers a variety of methods for control for different types of problem solving and provides formalisms for representing the additional knowledge used by these methods. The layer for example can offer methods for selectively gathering evidence for diagnostic applications as well as methods for intelligently pruning and focusing Bayesian network inference. These control methods are the basic building blocks for shaping complex, domain-dependent problem-solving behaviour. The two layers of the architecture are strictly separated and communicate in a highly restricted fashion. The control layer queries the probabilistic layer for information about the represented joint probability distribution and the evidence entered so far, and, based upon this information, takes strategic decisions as to how to proceed. The probabilistic layer computes and returns the information it is asked for by the control layer.

The problem-solving architecture explicitly separates probabilistic reasoning from control over reasoning. Several advantages arise from such an explicit separation. A Bayesian network can be developed and refined, without being hampered by any algorithmic issues. Moreover, the representation of the joint probability distribution on the domain at hand is not obscured by non-probabilistic knowledge. In addition, a Bayesian network can be re-used in different contexts for different purposes; a similar observation holds for the methods of control comprised in the control layer of the architecture. The architecture in addition provides for modelling decision problems. Knowledge about viable decisions and the preferences over their consequences involved in a decision problem are represented in the control layer, along with methods for solving the problem; the probabilistic layer comprises a Bayesian network that is used as a background knowledge base for providing the probabilities required. The idea of a meta-level problem-solving architecture pervades many areas of artificial intelligence research.

6.3.1 Example Application: Threshold Decision Making

As an example of the type of problems that can be modelled using the problem-solving architecture, the threshold model for decision making is considered.

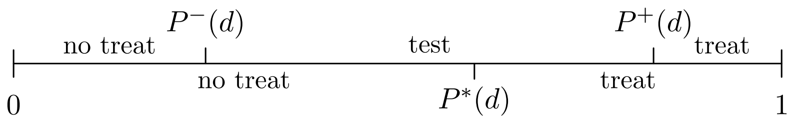


Figure 6.10: The threshold model for patient management, indicating three threshold probabilities and the various decision alternatives at a physician's disposal.

In the medical domain, Bayesian networks are often used for diagnostic purposes. A diagnostic Bayesian network typically comprises one or more variables modelling the presence of absence of disease, various variables modelling findings and results from diagnostic tests, and a number of intermediate variables modelling unobservable patho-physiological states. In the example network from Section 6.1 (Figure 6.1), for instance, the variable B models the disease of interest, being the presence or absence of a brain tumour; the variable MC models an unobservable state and the remaining variables capture findings and test results. A medical diagnostic Bayesian network is used for computing a most likely diagnosis for a patient given his or her presentation findings and test results.

The most likely diagnosis for a patient, along with its uncertainty, is generally taken by an attending physician to decide upon management of the patient. The physician may decide, for example, to start treatment right away. For the brain tumour example, the physician may decide to perform neurosurgery if a brain tumour is indicated. Alternatively, the physician may defer the decision whether or not to treat the patient until additional diagnostic information has become available, for example from a CT scan. Or, the physician may decide to withhold treatment altogether. To support choosing among these decision alternatives, the threshold model for patient management can be used. The threshold model for patient management is a typical example of something that can be implemented in the control layer of the problem-solving architecture.

The Threshold Model

The *threshold model for patient management*, or for decision making more in general, builds upon various threshold probabilities of disease [Pauker & Kassirer, 1980]. The *treatment threshold probability* of disease, written $P^*(d)$ for disease d, is the probability at which an attending physician is indifferent between giving treatment and withholding treatment. If, for a specific patient, the probability of disease $\Pr(d)$ exceeds the treatment threshold probability, that is, $\Pr(d) > P^*(d)$, then the physician will decide to treat the patient as if the disease were known to be present with certainty. Alternatively, if $\Pr(d) \leq P^*(d)$, the physician will basically withhold treatment from the patient.

As a consequence of the uncertainty concerning the presence of disease in a patient, additional information from a diagnostic test may affect an attending physician's basic management decision. If the probability of disease exceeds the treatment threshold probability, then interpreting a negative test result may result in an updated probability of disease *below* the threshold probability. Alternatively, if the pretest probability of disease falls below the treatment threshold probability, a positive test result may raise the probability of disease to a value *above* the threshold probability. To reckon with such effects, the threshold model for patient management includes another two threshold probabilities. The *no treatment-test threshold probability* of disease, written $P^-(d)$, is the probability at which the attending physician is indifferent between the decision to withhold treatment and the decision to obtain additional diagnostic information. The *test-treatment threshold probability* of disease, written $P^+(d)$, is the probability at which the physician is indifferent between obtaining additional information and starting treatment right away.

Figure 6.10 summarises the basic idea of the threshold model for patient management.

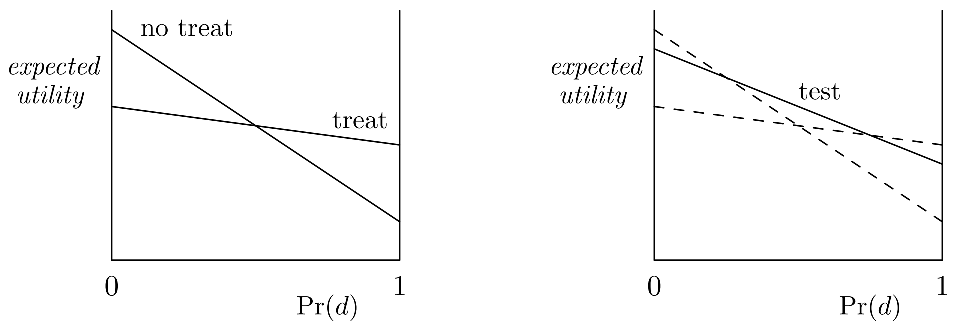


Figure 6.11: The basic idea of establishing (a) the treatment threshold probability of disease, and (b) the no-treatment-test and test-treatment threshold probabilities.

As long as the diagnostic test under consideration has not been performed, a physician has three decision alternatives at his or her disposal. If the probability of disease $\Pr(d)$ for a patient falls below the no treatment-test threshold probability, that is, if $\Pr(d) < P^{-}(d)$, then the physician will withhold treatment from the patient without gathering additional diagnostic information. If the probability of disease exceeds the test-treatment threshold probability, that is, $\Pr(d) > P^{+}(d)$, then the physician will start treatment right away. Otherwise, that is, if $P^{-}(d) \leq \Pr(d) \leq P^{+}(d)$, the physician will perform the diagnostic test. After testing, there are only two decision alternatives left. If the updated probability of disease for the patient exceeds the treatment threshold probability, the physician will start treatment; otherwise treatment will be withheld from the patient.

Determining Thresholds

The treatment threshold probability of disease $P^{*}(d)$ used in the threshold model is typically established by a physician after carefully weighing the various *utilities* involved. Utilities are numbers associated with a *utility function* and pertain to the presence or absence of disease on the one hand and giving or withholding treatment on the other hand. A utility function may be based on probabilistic information only and not involve any other information about the domain at hand. Yet, it may also incorporate non-probabilistic issues such as the cost of obtaining. From the *expected utilities* for giving and withholding treatment in view of the uncertainty concerning the presence of disease, the probability of disease at which the physician is indifferent between the two decision alternatives is readily determined; the basic idea is illustrated in Figure 6.11(a). For the brain tumour example, the physician will typically take into consideration the life expectancy for a patient, with and without a brain tumour, and the patient's attitude towards impaired health states; the physician can, for example, set the treatment threshold probability of a brain tumour at 0.15. The two threshold probabilities $P^{-}(d)$ and $P^{+}(d)$ for deciding whether or not to perform a diagnostic test are established from the test's characteristics. For the brain tumour example, a possible diagnostic test is the CT scan. Suppose this test is added to the example network as a direct descendant of the variable B, together with the conditional probabilities

$$\gamma_{CT}(ct \mid b) = 0.95,\ \gamma_{CT}(ct \mid \neg b) = 0.10$$

The physician will typically weigh the discomfort of a CT scan for a patient against the additional information yielded by the scan; the physician, for example, may set the no treatment-test threshold probability of a brain tumour at 0.045 and the test-treatment threshold probability at 0.56. The basic idea of establishing these two threshold probabilities is illustrated in Figure 6.11(b).

Beyond the Bayesian Network

While the Bayesian network framework offers algorithms for computing the probability of disease and for processing evidence, it does not provide for computing the threshold probabilities and for comparing these to computed probabilities of disease: these tasks involve knowledge that cannot be expressed in the Bayesian network formalism and require computations beyond probabilistic inference. These tasks therefore are provided for by the control layer of the problem-solving architecture. The control layer can thus provide the additional information required to support the decision between treatment, no treatment, or deferring treatment until additional diagnostic information is available. If it is decided that additional diagnostic information is to be obtained, the control layer can also be used for determining what additional information is to be obtained. A simple method for selective gathering of such information is discussed next.

6.3.2 Example Application: Selective Evidence Gathering

Another example of control over reasoning offered by the control layer is a simple method for selective gathering of evidence for diagnostic problem solving with a Bayesian network. This technique is regularly applied in intelligent tutoring systems (ITS), especially those for computerized adaptive testing (CAT), which aim to provide individual students with tailored exams based upon an assessment of their skills. [1]

In diagnostic problem solving, the objective is to identify a most likely explanation for a problem under consideration — this explanation then is the diagnosis of the problem. Establishing a diagnosis is achieved by gathering information about the manifestations of the problem at hand by applying tests to the problem. In most domains, it is not necessary to collect evidence on all possible manifestations before an accurate diagnosis is reached: information from only a few tests generally suffices. Moreover, it often is not desirable to apply all tests available as testing may be costly or damaging. In diagnostic problem solving, therefore, tests are not applied as a matter of course but instead are selected carefully. *Selective evidence gathering*, or *test planning*, now amounts to selecting the most useful tests to apply to a problem under consideration.

Tasks in Selective Evidence Gathering

In essence, selective evidence gathering is concerned with three tasks. The first of these is to select the test that is expected to yield the most useful information in the context of the evidence that is already available. When a test has been selected, the user, for example a physician, is requested to apply the test and to enter the evidence yielded. The second task of selective evidence gathering is to process this evidence. The third task is to decide whether enough evidence has been obtained as yet to confirm a diagnosis to sufficient extent. If still further information is required, the three tasks are executed recursively. We now take a closer look at these tasks in view of diagnostic problem solving with a Bayesian network. While the Bayesian network framework offers algorithms for computing probabilities and for processing evidence, thus providing for the second task of selective evidence gathering, it does not provide for valuing and selecting tests nor for deciding when to stop gathering information: these tasks involve knowledge that cannot be expressed in the Bayesian network formalism and require computations beyond probabilistic inference. These tasks therefore are provided for by the control layer of our problem-solving architecture.

[1]Search for 'computerized adaptive testing' in Google Scholar to find many papers on this topic.

Variable Roles In diagnostic problem solving, the variables from the domain at hand play different roles; for example, some variables represent test outcomes, others represent unobservable, intermediate process states. For distinguishing between different roles, the following types of vertex in the digraph of a Bayesian network are discerned [Henrion, 1989]: a *hypothesis vertex* represents one or more (mutually exclusive) hypotheses or disorders; an *evidence vertex* represents a variable whose value *can be* obtained by testing; all other vertices are *intermediate vertices*. The set of all vertices of the digraph G of a Bayesian network can thus be partitioned into three mutually disjoint sets of vertices: hypothesis vertices $\boldsymbol{H}_G$, evidence vertices $\boldsymbol{E}_G$, and intermediate vertices $\boldsymbol{I}_G$. The roles of the various vertices are modelled in the control layer of the problem-solving architecture and are not known to the probabilistic layer. In addition to knowledge concerning the roles of the vertices discerned, the control layer also specifies the additional knowledge required for assessing for each test the (expected) usefulness of information yielded by testing, and the extra knowledge involved in deciding when to stop gathering information.

Method and Assumptions for Bayesian Networks

For selective evidence gathering in diagnostic problem solving with a Bayesian network, generally two simplifying assumptions are made. First, a *myopic* approach to evidence gathering is taken, that is, evidence vertices to acquire information on are selected one by one. It is conceivable that in practical applications a non-myopic approach in which vertices are selected groupwise outperforms any method based on a myopic approach. Naively adopting a non-myopic approach, however, poses unsurmountable problems concerning computational complexity. The second simplifying assumption generally made is that the Bayesian network at hand comprises *one* hypothesis vertex H only, that is, it is assumed that all hypotheses discerned in the domain are mutually exclusive. Note that this assumption prohibits reasoning about multiple interacting disorders. Relaxing this assumption and straightforwardly applying selective evidence gathering in view of a set of hypothesis vertices also causes serious computational problems, since then all possible combinations of values for all hypothesis vertices have to be considered. In the remainder of this section, we will equally take up the two assumptions mentioned above, although research results have indicated that the simplifying assumptions may be eased to some extent [Heckerman *et al.*, 1993, Van der Gaag & Wessels, 1994b, Sent, 2005].

Selective evidence gathering for diagnostic problem solving with a Bayesian network may be envisioned as outlined below in pseudo-code. The evidence-gathering procedure takes the digraph G of a Bayesian network and the set $\boldsymbol{E}$ of all yet uninstantiated evidence vertices for its input and yields as output a diagnosis d, which corresponds with a value assignment to the hypothesis variable.

```
procedure evidence-gathering($G$,$\boldsymbol{E}$,$d$)
  enough := $false$;
  while $\boldsymbol{E} \neq \varnothing$ and not enough do
      dependent-vertices($H$,$\boldsymbol{E}$,$\boldsymbol{E'}$);
      if $\boldsymbol{E'} \neq \varnothing$ then
          select-vertex($\boldsymbol{E'}$,$E_j$);
          enter-evidence($E_j$);
          $\boldsymbol{E} := \boldsymbol{E} \setminus \{E_j\}$;
          enough := verify-enough()
      else enough := $true$
  od;
  return diagnosis(d)
end
```

In principle, for selecting from a Bayesian network an appropriate vertex to acquire information on, each yet uninstantiated evidence vertex has to be examined as to the (expected) usefulness of information yielded. To this end, several probabilities are computed from the Bayesian network. These probabilities may reveal that for some of the uninstantiated evidence vertices, entering information has no influence whatsoever on the probabilities of the values of the hypothesis vertex and therefore is utterly useless in view of establishing a diagnosis. This property holds for all vertices that are independent of the hypothesis vertex given the evidence obtained so far. Now recall that the Bayesian network formalism allows for identifying independences from the digraph of a network without having to resort to probabilistic computations. In the main evidence-gathering procedure, this property is exploited to save on the number of probabilities that has to be computed from the network. The dependent-vertices procedure is called upon to determine from a set $\boldsymbol{E}$ of uninstantiated evidence vertices the subset $\boldsymbol{E}'$ of those vertices that are not d-separated from the hypothesis vertex H given the evidence entered so far. For selecting an appropriate evidence vertex to acquire information on now only the vertices comprised in this set $\boldsymbol{E}'$ are examined. Here, we will not further elaborate on the dependent-vertices procedure; for the computational issues involved, the reader is referred to [Geiger *et al.*, 1990]. We would like to note that although the dependent-vertices procedure is called from the control layer of the problem-solving architecture, it itself is comprised in the probabilistic layer.

Once the set $\boldsymbol{E}'$ of relevant uninstantiated evidence vertices has been determined, the select-vertex procedure selects from this set the evidence vertex to best acquire information on. Recently, the interest for selective evidence gathering — or *test selection* as it is most often referred to nowadays — has revived, resulting in a number of new measures and corresponding algorithms to select those vertices for which to acquire evidence (see e.g. [Sent, 2005]). Here, we will focus only on one of the earlier approaches where a utility function is used for discriminating between the various evidence vertices. This utility function assigns to each value of every evidence vertex a numerical quantity expressing the desirability, or utility, of obtaining this value. The utility functions in use for selective evidence gathering differ considerably in the way they value information [Ben-Bassat, 1978, Glasziou & Hilden, 1989]. For selecting an appropriate evidence vertex, the select-vertex procedure may employ any such utility function. As an example, we consider here a very simple utility function tailored to binary variables [Van der Gaag & Wessels, 1994a]: the *linear-value utility function* u is defined by

$$u(E_i) = |\Pr(h \mid \widetilde{c}_{\boldsymbol{E}}) - \Pr(h \mid \widetilde{c}_{\boldsymbol{E}} \wedge E_i)|$$

for each evidence vertex $E_i \in \boldsymbol{E}'$, where H is the hypothesis vertex and $\widetilde{c}_{\boldsymbol{E}}$ denotes the partial configuration of all evidence obtained so far. Note that for an uninstantiated evidence vertex E_i, the difference between $\Pr(h \mid \widetilde{c}_{\boldsymbol{E}})$ and $\Pr(h \mid \widetilde{c}_{\boldsymbol{E}} \wedge e_i)$ indicates the confidence gained in the hypothesis h if the evidence $E_i = \textit{true}$ is observed; an analogous observation holds for the evidence $E_i = \textit{false}$. The usefulness of acquiring information on an evidence vertex, however, does not depend on a single value as it is uncertain which test result will be yielded for the problem at hand. For examining an evidence vertex, therefore, the utilities of its separate values are weighted with the probabilities that these values will be found. The result models the *expected utility* of acquiring information on the vertex. When employing the linear-value utility function u, the expected utility $\hat{u}$ for an evidence vertex $E_i \in \boldsymbol{E}'$ is computed from

$$\hat{u}(E_i) = \sum_{c_{E_i}} \Pr(c_{E_i} \mid \widetilde{c}_{\boldsymbol{E}}) \cdot u(c_{E_i})$$

To select the evidence vertex to best acquire information on, the select-vertex procedure computes the expected utilities for all relevant uninstantiated evidence vertices and then selects the vertex with highest expected utility.

Once an appropriate evidence vertex has been selected, the enter-evidence procedure prompts the user for a value for this vertex. The value obtained from the user is entered and subsequently processed in the Bayesian network at hand by the basic algorithms for probabilistic inference offered by the probabilistic layer of the problem-solving architecture. Note that the while-loop of the evidence-gathering procedure yields a sequence of prompts to the user concerning various evidence vertices.

Beyond the Bayesian Network

The last task of selective evidence gathering is to investigate whether enough evidence has been collected to justify a decision to stop further gathering of information. For this task, a *stopping criterion* is employed. Such a stopping criterion may be based on several different principles. The principle of *sufficiency of confirmation* is to stop evidence gathering as soon as a diagnosis has been confirmed to sufficient extent by the available information: the probability of a (tentative) diagnosis is compared with a pre-set threshold value, and if this probability has surpassed the threshold value and is expected not to drop considerably, evidence gathering is stopped. Note that such a stopping criterion is based on probabilistic information only. Another principle a stopping criterion may be based upon is the principle of *sufficiency of information*. This principle is to stop evidence gathering if the expected utilities of all remaining evidence vertices have dropped below a pre-set threshold value; pursuing evidence gathering then is expected not to further contribute to establishing a diagnosis. A stopping criterion based on this principle may involve both probabilistic and non-probabilistic information from the domain at hand. In the evidence-gathering procedure the two principles are combined. The principle of sufficiency of information is seen in the condition of the main while-loop of the procedure: evidence gathering is stopped if all remaining uninstantiated evidence vertices are independent of the hypothesis vertex given the evidence obtained so far. The verify-enough function completes the stopping criterion by implementing a test on sufficiency of confirmation.

6.4 Explaining Bayesian Networks

Classifying email into spam or no-spam is a task often performed by Bayesian network-based classifiers [Jin *et al.*]. For such an application, the user typically is only concerned with the accuracy of the model. If the spam-filter classifies most spam as spam and not too much legitimate email as spam, then the user does not care too much about the workings of the model or its reasons for labelling email in a certain way. Understanding of the model and its output is essential, however, for critical applications of decision-support systems concerning, for example, a patient's health or the innocence of a criminal suspect. Such systems can only be brought into practice if we are able to provide the necessary explanations.

Existing explanation methods for Bayesian networks can broadly be divided in three categories. First, the elements of the model itself can be explained. See, for instance, the work of [Lacave *et al.*, 2007] or [Koiter, 2006] in which properties of the nodes and the arcs in the network are explained. Secondly, the evidence that is instantiated in the Bayesian network can be explained by calculating, for instance, the so-called most probable explanation (MPE) or the maximum a-posteriori probability (MAP) assignment, which is the most likely configuration of a (sub)set of non-evidence variables[Pearl, 1988]. Thirdly, the reasoning chains, or crucial parts of them, that underlie the probabilistic inferences can be explained [Timmer, 2017, Van Leersum, 2015, Suermondt, 1992, Yap *et al.*, 2008] as well as local qualitative properties of these inferences such as direction of change in probability or the strength of the inference [Lacave *et al.*, 2007, Lacave & Díez, 2002, Koiter, 2006, Madigan *et al.*, 1997, Druzdzel, 1996]. The listed approaches typically provide verbal explanations and/or add visual cues to a Bayesian network graph. More recently, inspired by approaches to explaining black-box machine learning algorithms that link outputs to inputs, various types of explanation for Bayesian network that are used as classifiers have been proposed; see [Koopman & Renooij, 2021] for an overview.

Exercises

* Exercise 6.1

Consider the digraph of a Bayesian network with eight vertices E_1, E_2, E_3, E_4, E_5, H_1, H_2, and H_3, and nine arcs (E_1, E_2), (E_1, H_1), (E_1, H_2), (E_3, E_5), (H_1, E_2), (H_1, E_3), (H_2, E_3), (H_2, E_4), and (H_3, E_4).

a. *Establish the sensitivity set for the variable of interest H_1; also try to explain in an informal, intuitive way why certain variables are included in the sensitivity set and why other variables are not;*

b. *same question for variable of interest H_1 given evidence for variable E_1;*

c. *same for H_1 given evidence for E_1 and E_3;*

d. *same for H_2 given E_1, E_2, E_3, E_4 and E_5;*

Suppose all variables are binary-valued and that an observation c_{E_2} is entered for the variable E_2. We are interested in the posterior of $H_2 = true$, $\Pr(h_2 \mid c_{E_2})$. We perform a sensitivity analysis by computing $f_{\Pr(h_2|c_{E_2})}(x)$ for various network parameters $x \in \{\gamma(v_i \mid c_{\boldsymbol{\rho}_G(V_i)})\}$; complements $\gamma(\neg v_i \mid c_{\boldsymbol{\rho}_G(V_i)})$ are co-varied and excluded from explicit analysis.

e. *A sensitivity analysis of a mathematical model in general consists of constructing a sensitivity* curve*, fitted to a number of individually computed points $(p, o(p))$ for different values of an input parameter p and the corresponding output $o(p)$. Suppose we use this approach to construct such sensitivity curves for the example Bayesian network. For all network parameters and the output probability described above, we compute 10 points per curve. Suppose in addition that we do not exploit the sensitivity set. How many network propagations (calls to a standard propagation algorithm for probabilistic inference) are required for this analysis?*

f. *How many network propagations are required for computing $f_{\Pr(h_2|c_{E_2})}(x)$ if we exploit both the known functional form of the sensitivity function and restrict the analysis to all free model parameters associated with the sensitivity set for variable of interest H_2 given evidence E_2? (In your answer, take into consideration whether the calculation of f requires 2 (linear), 3 (hyperbolic, but $\Pr(c_{E_2})$ does not vary with x), or 4 (hyperbolic in general) constants).*

* Exercise 6.2

Consider the small Brain-tumour network from Figure 6.1.

a. *Suppose that only the variable B has been observed. For which variables V_i can output probabilities $\Pr(V_i \mid B)$* not *be influenced by variation of the model parameters associated with B?*

b. *Establish the minimal set of variables $\mathbf{V}$ that need be observed to render variable CT given C insensitive to variation of the model parameters of ISC.*

c. *To which model parameters does the output probability* $\Pr(c \mid sh)$ *relate linearly? Which model parameters may, upon variation, have a non-linear effect on this output probability?*

d. *Analytically express the probability* $\Pr(b)$ *as a function of model parameter* $\gamma(b \mid mc)$.

e. *Analytically express the probability* $\Pr(c \mid b \wedge mc)$ *as a function of model parameter* $\gamma(isc \mid mc)$.

* Exercise 6.3

Consider a sensitivity function $f(x)$ for some output probability and some model parameter $x = \gamma_V(v_s)$ of variable V with $n > 2$ values; we assume that the original value x_0 of $\gamma_V(v_s) \in \langle 0, 1\rangle$. Upon varying x, the values of the assessment function γ_V for the other $n-1$ values of variable V have to be co-varied. To this end, a proportional *co-variation scheme is used in which the proportion of the remaining mass $1-x$ that is assigned to $\gamma_V(v_j)$, $j \neq s$, is kept constant.*

Different co-variation schemes adhere to different properties. Two useful properties are the order-preserving *property and the* impossibility-preserving *property. A co-variation scheme is called impossibility-preserving if any $\gamma_V(v_j) = 0$, $j \neq s$, remains zero upon co-variation. Suppose the values of V are ordered according to the values of γ_V as specified in the network, i.e. $\gamma_V(v_1) \leq \ldots \leq \gamma_V(v_s) \leq \ldots \leq \gamma_V(v_n)$; a co-variation scheme is called order-preserving if this ordering is preserved during co-variation.*

Which of these two properties does the proportional co-variation scheme adhere to? Explain your answer:

I. neither order-preserving, nor impossibility-preserving
II. not order-preserving, yet impossibility-preserving
III. order-preserving, but not impossibility-preserving
IV. both order-preserving and impossibility-preserving

* Exercise 6.4

Consider a noisy-or gate modelling a disjunctive interaction between n causes $C_1, \ldots, C_n$ and effect E (all variables are binary-valued). Assume that each cause C_i, $i = 1, \ldots, n$ has a uniform *prior distribution. Show that the following formula correctly captures the sensitivity function $f_{\Pr(e)}(x)$ for $x = \Pr(c_1)$:*

$$f_{\Pr(e)}(x) = 2^{-(n-1)} \cdot \Big(\big(\sum_{c_{\mathbf{C}} \models c_1} \Pr(e \mid c_{\mathbf{C}}) \; - \sum_{c'_{\mathbf{C}} \models \neg c_1} \Pr(e \mid c'_{\mathbf{C}}) \big) \cdot x + \sum_{c'_{\mathbf{C}} \models \neg c_1} \Pr(e \mid c'_{\mathbf{C}}) \Big)$$

where $c_{\mathbf{C}} \models c_1$ represents any configuration over the set of variables $\mathbf{C} = \{C_1, \ldots C_n\}$ that is consistent with $C_1 = true$, and $c'_{\mathbf{C}} \models \neg c_1$ is any configuration over $\mathbf{C}$ consistent with $C_1 = false$.

* Exercise 6.5

One in a thousand people is susceptible to a particular heart disease. There is a test to detect this disease. The test is 100% accurate for people who have the disease and is 95% accurate for those who do not (this means that 5% of people who do not have the disease will be wrongly diagnosed as having it).

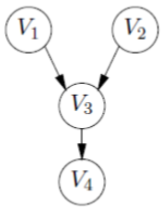
a. *Draw a Bayesian network that reflects the above described relationship between disease and test-result; give all assessment functions, or conditional probability tables, as well.*

b. *Establish the sensitivity function* $f_{\Pr(\mathit{Heart\text{-}disease}=\mathit{yes} \mid \mathit{Test}=\mathit{yes})}(x)$ *for* $x = \gamma(\mathit{Heart\text{-}disease} = \mathit{yes})$.

c. *Determine the sensitivity value and the admissible deviation for* $x_0 = 0.001$. *Comment on their interpretation in terms of the domain under consideration.*

*** Exercise 6.6**

Consider the Bayesian network $\mathcal{B} = (G, \Gamma)$ *that represents joint probability distribution* Pr *for some domain of application. Let* G *be the following digraph:*



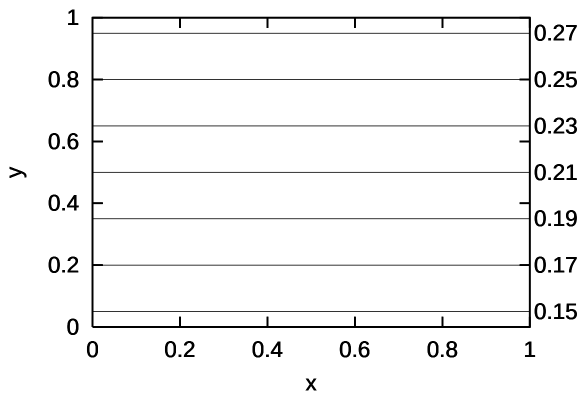
Consider a one-way sensitivity analysis of $\mathcal{B}$. *In general, the one-way sensitivity function* $f(x)$, *describing an output probability of interest in terms of a network parameter* x, *has the following form:*

$$f(x) = \frac{a \cdot x + b}{c \cdot x + d} \quad \text{where } a,\ b,\ c \text{ and } d \text{ are constants with respect to } x.$$

a. *Suppose we are interested in the posterior output probability* $\Pr(v_3 \mid v_4)$ *and how it changes upon varying model parameter* $x = \gamma(v_4 \mid \neg v_3)$. *Show that for this particular case the sensitivity function takes on the following simplified form:*

$$f_{\Pr(v_3 \mid v_4)}(x) = \frac{a}{b \cdot x + a}$$

b. *Suppose that also a two-way sensitivity analysis of network* $\mathcal{B}$ *is performed. Consider the probability of interest* $\Pr(v_3)$ *and two model parameters* $x = \gamma(v_4 \mid \neg v_3)$ *and* $y = \gamma(v_2)$. *The following graph displays projected iso-probability lines for* $f_{\Pr(v_3)}(x, y)$:



Clearly explain what this graph tells you about

- *the joint effect of varying* x *and* y *on* $\Pr(v_3)$, *and*

- *the individual effects of varying x and y on* $\Pr(v_3)$.

c. *Give an advantage and a drawback of performing a two-way sensitivity analysis as opposed to a one-way analysis.*

* Exercise 6.7

Argue that the method for establishing a sensitivity set by adding auxiliary parents to the nodes in the digraph indeed works.

* Exercise 6.8

Consider weather forecaster Piet who makes daily predictions for Rain ($\Pr(r)$ *and* $\Pr(\neg r)$*) in a country where it rains 70% of the time. Every day Piet considers whether or not it rained the day before and computes the Brier score for yesterday's prediction.*

a. *What is the average Brier score (in the long run) for Piet if he is a* uniform *forecaster and predicts* $\Pr(r) = \Pr(\neg r) = 0.5$ *for each day?*

b. *What is the average Brier score for an oesophageal cancer network that predicts uniform distributions over the 6 possible stages?*

c. *What is the average Brier score over n predictions if Piet is a* base rate *predictor, i.e. predicts* $\Pr(r) = 0.7$ *every day?*

d. *What are the Brier scores for a correct and an incorrect prediction if Piet is an opportunistic, deterministic predictor and predicts* $\Pr(r) = 1$*? And what is his average score over n predictions given that m of these are correct? What is the expected value of m and the corresponding average score?*

Chapter 7

Conclusions

Probabilistic modelling and reasoning through Probabilistic Graphical Models (PGMs) is an exciting research area. First and foremost, PGMs may be looked upon as mathematically sound computational frameworks for probabilistic inference. From this point of view, we have addressed the algorithms offered by the Bayesian network framework. We have argued that although these algorithms have an exponential worst-case time complexity, they tend to behave polynomially for most real-life Bayesian networks. However, as applications of the framework grow larger, the Bayesian networks involved increase in size accordingly. Networks comprising hundreds or even thousands of vertices are no exception. For Bayesian networks of this size, the basic algorithms for probabilistic inference inevitably slow down problem solving despite their polynomial behaviour. Research into inference algorithms therefore aims at developing more efficient algorithms. Efficiency is sought after in many different ways: existing algorithms for exact inference are further optimised, e.g. through knowledge-based pruning and focusing, any-time algorithms are proposed, and various algorithms for approximate inference, often based on simulation techniques, have been designed.

The PGM framework may also be looked upon as a framework for building knowledge-based systems. In fact, experience with developing applications of the framework is progressing rapidly. From this point of view, we have addressed the issue of building a Bayesian network for a domain of application. In many respects, building a Bayesian network resembles engineering a knowledge-based system more in general. Available knowledge-engineering methodologies are more and more supplemented with methodologies tailored to Bayesian network building. With the increasing availability of data over the past decades, the popularity of research into the possibilities for automated construction has also grown. The current interest in explainable AI will possibly awaken the interest in explanation methods for Bayesian networks as well. As experience with applying the PGM framework is building, the need for methods for knowledge-based control over reasoning is evident. We have briefly addressed this issue and outlined shaping diagnostic problem solving with a Bayesian network.

In this syllabus we have focussed mostly on the Bayesian network, which is the PGM family member that works with discrete variables and a directed graph. Other family members include more traditional statistical models such as Markov networks (undirected graph; also called Markov random fields), Kalman filters (continuous variables) and Hidden Markov models. The latter can actually be modelled as a (Dynamic) Bayesian network of fixed topology, showing that the material presented in this syllabus is applicable to a wider range of probabilistic models.

Chapter 8

Solutions, Answers and Hints

This chapter presents solutions, answers or tips for selected exercises. Often also the approach you can take to come up with a solution is given. Note that for a lot of exercise there are multiple ways to solve the problem and we present at most one.

1.2

c. We need to sum out $n-m$ variables; since all variables are binary-valued this amounts to summing over 2^{n-m} joint probabilities.

2.2

The correct answers follow from the notation definitions (see boxes in Section 2.2 and the Ch. 2 course slides): **a**-I, **b**-VI, **c**-IV, **d**-III, **e**-II, **f**-V, **g**-VII

2.3

a. *Approach:* to introduce conditional probabilities, use Definition 2.2.4. First just separate V_n from the rest and check out the result. Now apply the definition of conditional probability repeatedly. Note that this works regardless of the values of V_i and therefore you can just use the template notation.

b. *Approach:* again you can use the definition of conditional probability, together with symmetry ($\Pr(X \wedge Y) = \Pr(Y \wedge X)$) in template notation; also try with and without $\boldsymbol{Z}$.

c. We prove the marginalisation property stated in Proposition 2.2.8. Let $\boldsymbol{V}$ be a set of random variables and let Pr be a joint probability distribution on $\boldsymbol{V}$. We have to show that

$$\Pr(\boldsymbol{X}) = \sum_{c_{\boldsymbol{Y}}} \Pr(\boldsymbol{X} \wedge c_{\boldsymbol{Y}})$$

for all sets $\boldsymbol{X}, \boldsymbol{Y} \subseteq \boldsymbol{V}$. From the definition of conditional probability, we have that

$$\begin{aligned}
\sum_{c_{\boldsymbol{Y}}} \Pr(\boldsymbol{X} \wedge c_{\boldsymbol{Y}}) &= \sum_{c_{\boldsymbol{Y}}} \Pr(c_{\boldsymbol{Y}} \mid \boldsymbol{X}) \cdot \Pr(\boldsymbol{X}) = \\
&= \Pr(\boldsymbol{X}) \cdot \sum_{c_{\boldsymbol{Y}}} \Pr(c_{\boldsymbol{Y}} \mid \boldsymbol{X}) = \\
&= \Pr(\boldsymbol{X}) \cdot 1
\end{aligned}$$

by the definition of Pr, which concludes our proof.
d. *Approach:* again use the definition of conditional probability on the right-hand side of the expression. Observe that the rest of the proof follows from the marginalisation property proven above.

2.4

Prove the property for arbitrary configurations $c_{\boldsymbol{X}}$ and $c_{\boldsymbol{Y}}$ rather than using the template notation. *Approach:* (other approaches are possible!) Observe that, due to the interpretation of the logical OR,

$$\Pr(c_{\boldsymbol{X}} \vee c_{\boldsymbol{Y}}) = \Pr(c_{\boldsymbol{X}} \wedge c_{\boldsymbol{Y}}) + \sum_{c'_{\boldsymbol{Y}} \neq c_{\boldsymbol{Y}}} \Pr(c_{\boldsymbol{X}} \wedge c'_{\boldsymbol{Y}}) + \sum_{c'_{\boldsymbol{X}} \neq c_{\boldsymbol{X}}} \Pr(c'_{\boldsymbol{X}} \wedge c_{\boldsymbol{Y}})$$

Also observe that the first two terms sum to $\Pr(c_{\boldsymbol{X}})$ by marginalisation (Proposition 2.2.8). Now rewrite the last term.

2.5

Approach: (other approaches are possible!) Start with the expression on the right-hand side and apply the definition of conditional probability to initially get rid of both conditioning bars. Now try to get rid of $c_{\boldsymbol{Y}}$.

2.6

To prove the property stated in the exercise, we show that

$$\Pr(\boldsymbol{X} \mid \boldsymbol{Y} \wedge \boldsymbol{Z}) = \Pr(\boldsymbol{X} \mid \boldsymbol{Z}) \Leftrightarrow \Pr(\boldsymbol{X} \wedge \boldsymbol{Y} \mid \boldsymbol{Z}) = \Pr(\boldsymbol{X} \mid \boldsymbol{Z}) \cdot \Pr(\boldsymbol{Y} \mid \boldsymbol{Z})$$

under the assumption that both $\Pr(\boldsymbol{Y} \wedge \boldsymbol{Z}) > 0$ and $\Pr(\boldsymbol{Z}) > 0$ for all values of $\boldsymbol{Y}$ and $\boldsymbol{Z}$. From the definition of conditional probability, we have that

$$\Pr(\boldsymbol{X} \mid \boldsymbol{Y} \wedge \boldsymbol{Z}) = \Pr(\boldsymbol{X} \mid \boldsymbol{Z}) \quad \Leftrightarrow \frac{\Pr(\boldsymbol{X} \wedge \boldsymbol{Y} \wedge \boldsymbol{Z})}{\Pr(\boldsymbol{Y} \wedge \boldsymbol{Z})} = \frac{\Pr(\boldsymbol{X} \wedge \boldsymbol{Z})}{\Pr(\boldsymbol{Z})}$$

By division of both the left hand side and the right hand side of the latter expression by $\Pr(\boldsymbol{Z})$ and subsequent rearrangement of terms, we find that

$$\Pr(\boldsymbol{X} \mid \boldsymbol{Y} \wedge \boldsymbol{Z}) = \Pr(\boldsymbol{X} \mid \boldsymbol{Z}) \Leftrightarrow \frac{\Pr(\boldsymbol{X} \wedge \boldsymbol{Y} \wedge \boldsymbol{Z})}{\Pr(\boldsymbol{Z})} = \frac{\Pr(\boldsymbol{X} \wedge \boldsymbol{Z})}{\Pr(\boldsymbol{Z})} \cdot \frac{\Pr(\boldsymbol{Y} \wedge \boldsymbol{Z})}{\Pr(\boldsymbol{Z})} \Leftrightarrow$$

$$\Leftrightarrow \Pr(\boldsymbol{X} \wedge \boldsymbol{Y} \mid \boldsymbol{Z}) = \Pr(\boldsymbol{X} \mid \boldsymbol{Z}) \cdot \Pr(\boldsymbol{Y} \mid \boldsymbol{Z})$$

which concludes our proof.

3.1

For $I_{\Pr}$ you need to use definitions based on Pr.

a. Proof for the symmetry property is given in the Ch. 3 course slides.

b. A sketch of the proof of the decomposition property is in the course slides; figure out the few missing steps.

c. We show that the independence relation I_{Pr} satisfies the weak union property

$$I_{\mathrm{Pr}}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y} \cup \boldsymbol{W}) \to I_{\mathrm{Pr}}(\boldsymbol{X}, \boldsymbol{Z} \cup \boldsymbol{W}, \boldsymbol{Y})$$

for all sets of variables $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{W} \subseteq \boldsymbol{V}$, that is, we prove the third property stated in Theorem 3.1.2.

We assume that $I_{\mathrm{Pr}}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y} \cup \boldsymbol{W})$. From this observation, we have

$$\Pr(\boldsymbol{X} \mid \boldsymbol{Z} \wedge \boldsymbol{Y} \wedge \boldsymbol{W}) = \Pr(\boldsymbol{X} \mid \boldsymbol{Z})$$

by definition, that is, we have that

$$\frac{\Pr(\boldsymbol{X} \wedge \boldsymbol{Y} \wedge \boldsymbol{Z} \wedge \boldsymbol{W})}{\Pr(\boldsymbol{Y} \wedge \boldsymbol{Z} \wedge \boldsymbol{W})} = \frac{\Pr(\boldsymbol{X} \wedge \boldsymbol{Z})}{\Pr(\boldsymbol{Z})}$$

From our assumption $I_{\mathrm{Pr}}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y} \cup \boldsymbol{W})$, we further have $I_{\mathrm{Pr}}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{W})$ by the second property stated in the exercise. By definition, we therefore have that

$$\Pr(\boldsymbol{X} \mid \boldsymbol{Z} \wedge \boldsymbol{W}) = \Pr(\boldsymbol{X} \mid \boldsymbol{Z})$$

that is,

$$\frac{\Pr(\boldsymbol{X} \wedge \boldsymbol{Z} \wedge \boldsymbol{W})}{\Pr(\boldsymbol{Z} \wedge \boldsymbol{W})} = \frac{\Pr(\boldsymbol{X} \wedge \boldsymbol{Z})}{\Pr(\boldsymbol{Z})}$$

Now consider the conditional probability $\Pr(\boldsymbol{X} \mid \boldsymbol{Z} \wedge \boldsymbol{W} \wedge \boldsymbol{Y})$. By definition, we find that

$$\Pr(\boldsymbol{X} \mid \boldsymbol{Z} \wedge \boldsymbol{W} \wedge \boldsymbol{Y}) = \frac{\Pr(\boldsymbol{X} \wedge \boldsymbol{Y} \wedge \boldsymbol{Z} \wedge \boldsymbol{W})}{\Pr(\boldsymbol{Y} \wedge \boldsymbol{Z} \wedge \boldsymbol{W})}$$

From the previous observations, we further find that

$$\Pr(\boldsymbol{X} \mid \boldsymbol{Z} \wedge \boldsymbol{W} \wedge \boldsymbol{Y}) \quad = \frac{\Pr(\boldsymbol{X} \wedge \boldsymbol{Z})}{\Pr(\boldsymbol{Z})} = \frac{\Pr(\boldsymbol{X} \wedge \boldsymbol{Z} \wedge \boldsymbol{W})}{\Pr(\boldsymbol{Z} \wedge \boldsymbol{W})} = \Pr(\boldsymbol{X} \mid \boldsymbol{Z} \wedge \boldsymbol{W})$$

From $\Pr(\boldsymbol{X} \mid \boldsymbol{Z} \wedge \boldsymbol{W} \wedge \boldsymbol{Y}) = \Pr(\boldsymbol{X} \mid \boldsymbol{Z} \wedge \boldsymbol{W})$, we have by definition that $I_{\mathrm{Pr}}(\boldsymbol{X}, \boldsymbol{Z} \cup \boldsymbol{W}, \boldsymbol{Y})$. We conclude that $I_{\mathrm{Pr}}(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y} \cup \boldsymbol{W}) \to I_{\mathrm{Pr}}(\boldsymbol{X}, \boldsymbol{Z} \cup \boldsymbol{W}, \boldsymbol{Y})$.

d. *Approach:* Similar to the proof above, use the definition of I_{Pr} and that of conditional probability.

3.2

We begin our proof by observing that, since I is a semi-graphoid independence relation, it obeys the first four axioms stated in Definition 3.1.3. Now, we assume that $I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y} \cup \boldsymbol{W})$ and $I(\boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{W})$. We have that

$$I(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y} \cup \boldsymbol{W}) \to I(\boldsymbol{X}, \boldsymbol{Z} \cup \boldsymbol{W}, \boldsymbol{Y}) \to I(\boldsymbol{Y}, \boldsymbol{Z} \cup \boldsymbol{W}, \boldsymbol{X})$$

by the weak union and symmetry axioms; in conjunction with our assumption $I(\boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{W})$, we find

$$I(\boldsymbol{Y}, \boldsymbol{Z} \cup \boldsymbol{W}, \boldsymbol{X}) \wedge I(\boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{W}) \to I(\boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{W} \cup \boldsymbol{X}) \to I(\boldsymbol{X} \cup \boldsymbol{W}, \boldsymbol{Z}, \boldsymbol{Y})$$

by the contraction and symmetry axioms.

3.3

Approach: for I you need to use the four axioms based upon which the relation is defined (Definition 3.1.3). The proof proceeds similar to that of Exercise 3.2 and the example in the Ch. 3 course slides. You will need the symmetry, weak union and contraction axioms.

3.5

Note that the only axiom that can be applied to the two statements is symmetry, so the independence relation consists of four statements in total.

a. *Approach:* to find D-maps, recall from Lemma 3.2.6 that a graph without edges is always a D-map. You can now add in edges as long as you do not introduce *de*pendences that contradict the independence statements. That is, the separation properties that correspond to the statements from the independence relation should remain valid in the graph. For the first independence statement this means that V_1 and V_4 *cannot* be directly connected by an edge; moreover, if there does exist a path between V_1 and V_4, then it must pass through V_2 and/or V_3. Similar observations apply to the second statement.

This exercise requires you to draw 16 D-maps, four of which are given in the Ch. 3 course slides. You should find:

- 1 graph with 0 edges;
- 1 graph which is complete except for the 2 forbidden edges: $0.5 \cdot 4 \cdot 3 - 2 = 4$ allowable edges;
- $4 = \binom{4}{1}$ graphs with 1 of the allowable edges;
- $6 = \binom{4}{2}$ graphs with 2 allowable edges;
- $4 = \binom{4}{3}$ graphs with 3 allowable edges.

b. *Approach:* to find I-maps, recall from Lemma 3.2.6 that a complete graph is always an I-map. You can now remove edges as long as you do not introduce *in*dependences that are absent from the independence relation. That is, the separation properties that are introduced into the graph should have a corresponding statement in the independence relation.

This exercise requires you to draw the 4 I-maps that are given in the Ch. 3 course slides.

3.6

Approach: first draw yourself a graph to visualize the problem; according to the exercise, V_j must be distinct from V_i and not among V_i's neighbours.

Proof: Every chain from V_i to V_j in G will contain at least one neighbour of V_i; this neighbour will block the chain. Therefore, the set of neighbours of V_i blocks all chains between V_i and V_j and hence separates V_i and V_j.

3.7

a. The property holds: verify that any simple chain between V_1 and V_2 is blocked by V_2 and/or V_3.

b. The property doesn't hold: two vertices can only be d-separated by the empty set if there is either *no* chain connecting the vertices, or all chains include a head-to-head vertex (vertex with two incoming arcs on the chain).

c. The property $\langle\{V_2\} \mid \{V_1\} \mid \{V_3\}\rangle_G^d$ holds in the digraph G since all chains in G from V_2 to V_3 are blocked by the set of vertices $\{V_1\}$. For example, the chain V_2, V_1, V_3 from V_2 to V_3 is blocked by $\{V_1\}$ since $V_1 \in \{V_1\}$; the chain V_2, V_5, V_3 from V_2 to V_3 is blocked by $\{V_1\}$ since $\{V_5, V_6\} \cap \{V_1\} = \varnothing$.

d. Verify yourself that the property holds.

e. The property $\langle\{V_2\} \mid \{V_3, V_4\} \mid \{V_6\}\rangle_G^d$ does *not* hold in the digraph G since not every chain in G from V_2 to V_6 is blocked by the set of vertices $\{V_3, V_4\}$. For example, the chain V_2, V_5, V_6 from V_2 to V_6 is not blocked by $\{V_3, V_4\}$.

f. The property doesn't hold: two directly connected vertices cannot be d-separated (other than by themselves if you allow non-disjoint sets).

3.8

Tip: Remember to apply the axioms to get the full independence relation! In addition to symmetry, we can apply both weak-union and decomposition to the second statement. In total this will give 6 independence statements plus their 6 symmetric versions; write these down!

a. *Approach:* to find D-maps, recall from Lemma 3.2.15 that a graph without any arcs is always a D-map. You can now add in arcs as long as you do not introduce *de*pendences that contradict the independence statements. That is, the d-separation properties that correspond to the 12 statements from the relation should remain valid in the graph. For the first statement this for example means that V_1 and V_2 *cannot* be directly connected by an arc; moreover, if there does exist a chain between V_1 and V_2 then it must pass through a head-to-head vertex. Similarly, the other statements tell us that the D-map cannot have direct arcs between V_1 and V_4, and between V_2 and V_4, and indirect chains between these vertices must pass through V_3.

This exercise requires you to draw 15 D-maps, four examples of which are given in the Ch. 3 course slides. You should find:

- 1 graph with 0 arcs;
- 1 graph which is complete except for the 6 $(3 \cdot 2)$ forbidden arcs: 3 allowable arcs with only one possible direction;
- $6 = 4 \cdot 3 - 6$ graphs with 1 allowable arc;
- $7 = \binom{6}{2} - 5 - 3$ graphs with 2 allowable arcs (-5 to get head-to-heads in correct place, -3 to prevent 2 arcs between 2 variables).

b. *Apporach:* to find I-maps, recall from Lemma 3.2.15 that a complete oriented graph is always an I-map. You can now remove arcs as long as you do not introduce *in*dependences that are absent from the independence relation. That is, the d-separation properties that are introduced into the graph should have a corresponding statement in the independence relation. From the independence relation we have for example that in the I-map there has to be a chain (without head-to-head vertex) from both V_1 and V_2 to V_4. Moreover, whenever there is no arc between V_1 and V_2, all chains between V_1 and V_2 should include a head-to-head vertex.

This exercise now requires you to draw (some of the) 63(!) possible I-maps, four examples of which are given in the Ch. 3 course slides. You should find:

- 24 complete oriented graphs with $0.5 \cdot 4 \cdot 3 = 6$ arcs (not 2^6 graphs, because they are not all acyclic!);
- 28 graphs with 5 arcs;
- 10 graphs with 4 arcs;
- 1 graph with 3 arcs.

3.9

A sketch of the proof is given in the Chapter 3 (see Lemma 3.2.15). The remaining arguments are similar to those in the proof of Lemma 3.2.6 for undirected graphs; also see the Ch. 3 course slides.

3.10

Approach: try to find as simple examples as possible. In this case an independence relation over only three variables should suffice. You have multiple P-maps if there are arcs that you can reverse without violating the P-map property. Essentially this means that you do not introduce or remove head-to-head nodes.

3.11

One example, out of 20 possible minimal I-maps, is given in the Ch. 3 course slides. *Approach:* note that if an arc between V_1 and V_4 is absent, then the graph should include at least one chain between V_1 and V_4 without a head-to-head vertex. A similar observation applies to the combination V_2 and V_3.

3.12

Approach: first draw yourself a graph to visualize the problem; according to the exercise, V_j must be distinct from V_i and not among V_i's direct ancestors (parents) or descendants.

The proof now proceeds similarly to that of Exercise 3.6. Think about why the property is not stated in terms of neighbours for directed graphs.

3.13

Not necessarily; again use a simple independence relation on three variables to construct a counter example. Think about what the real difference between the languages of undirected and directed graphs for encoding independences is: the interpretation of the head-to-head connection.

3.14

Approach: try independence relations over as few variables as possible. For example, use three variables for a) and c), and four variables for b) and d). Let yourself be inspired by the example relations you have already encountered above and in the Ch. 3 course slides, and think about the true difference in expressiveness between directed and undirected graphs (head-to-head connections).

4.2

Chapter 4.1 and the Ch. 4 course slides on the probabilistic interpretation, give examples of how to perform such computations. For this example network, your computations should give $\Pr(v_1 \wedge v_2 \wedge v_3) = 0.09375$; $\Pr(v_2 \wedge v_3) = 0.24375$; $\Pr(v_1 \mid v_2 \wedge v_3) = 0.38462$ (first use definition of conditional probability); $\Pr(v_1 \vee v_2 \vee \neg v_3 \vee v_4) = 0.985$ (first transform the logical disjunction into a logical conjunction with De Morgan's Law).

4.3

a. Note that you are asked to prove that the Identity property from Lemma 4.2.8 also holds for singly connected subgraphs in which no evidence resides. For the proof: be inspired by the proof of Lemma 4.2.8 or directly use the definition of $\lambda_{V_i}(V_i)$ (Definition 4.2.11).

Also note that if there is no evidence at all in the network, then the Identity property holds in the singly connected graph as a whole. Moreover, the property also holds in directed trees for any vertex that forms the root of a sub-tree without evidence.

b. and c. the result similarly follows from the definitions of the parameters from Chapter 4.2.2.

4.4

Tip: Chapter 4.2.2 and the Ch. 4 course slides on Pearl in singly connected graphs give examples of how to perform computations using Pearl's belief propagation algorithm. Make sure you can follow these before attempting to solve this and the following exercises.

Approach: Now try doing this exercise yourself before looking at the worked out example below: this is something you need to practice and experience! A general tip: always start with the data fusion lemma for the requested probabilities and apply the computation rules only for computing those compound and message parameters required to establish the requested probabilities.Also practice for the exam: make clear which compound and message parameters are computed, how they are computed and which additional assumptions (such as special cases or the Identity property) you make, if any. A way to do this is given below.

a. Vertex V_4 applies the *data fusion lemma* to compute the probabilities $\Pr(v_4)$ and $\Pr(\neg v_4)$:

$$\begin{aligned}\Pr(v_4) &= \alpha \cdot \pi_{V_4}(v_4) \cdot \lambda_{V_4}(v_4) \\ \Pr(\neg v_4) &= \alpha \cdot \pi_{V_4}(\neg v_4) \cdot \lambda_{V_4}(\neg v_4)\end{aligned}$$

Since no evidence has been entered in the network, we have from *the identity property* that $\lambda_{V_4}(v_4) = \lambda_{V_4}(\neg v_4) = 1$. As a consequence, *no normalisation* is required upon data fusion. For the compound causal parameter $\pi_{V_4}(v_4)$, vertex V_4 now computes

$$\pi_{V_4}(v_4) = \gamma_{V_4}(v_4 \mid v_3) \cdot \pi^{V_3}_{V_4}(v_3) + \gamma_{V_4}(v_4 \mid \neg v_3) \cdot \pi^{V_3}_{V_4}(\neg v_3)$$

Due to the absence of evidence, the *causal parameter equivalence* holds. As a result, $\pi^{V_3}_{V_4}(V_3) = \pi_{V_3}(V_3)$, which results in:

$$\begin{aligned}\pi^{V_3}_{V_4}(v_3) &= \gamma_{V_3}(v_3 \mid v_1 \wedge v_2) \cdot \pi^{V_1}_{V_3}(v_1) \cdot \pi^{V_2}_{V_3}(v_2) + \\ &+ \gamma_{V_3}(v_3 \mid \neg v_1 \wedge v_2) \cdot \pi^{V_1}_{V_3}(\neg v_1) \cdot \pi^{V_2}_{V_3}(v_2) + \\ &+ \gamma_{V_3}(v_3 \mid v_1 \wedge \neg v_2) \cdot \pi^{V_1}_{V_3}(v_1) \cdot \pi^{V_2}_{V_3}(\neg v_2) + \\ &+ \gamma_{V_3}(v_3 \mid \neg v_1 \wedge \neg v_2) \cdot \pi^{V_1}_{V_3}(\neg v_1) \cdot \pi^{V_2}_{V_3}(\neg v_2)\end{aligned}$$

Again using *causal parameter equivalence* and the fact that V_1 and V_2 are *roots*, we find:

$$\begin{aligned}\pi^{V_3}_{V_4}(v_3) &= \gamma_{V_3}(v_3 \mid v_1 \wedge v_2) \cdot \gamma_{V_1}(v_1) \cdot \gamma_{V_2}(v_2) + \\ &+ \gamma_{V_3}(v_3 \mid \neg v_1 \wedge v_2) \cdot \gamma_{V_1}(\neg v_1) \cdot \gamma_{V_2}(v_2) + \\ &+ \gamma_{V_3}(v_3 \mid v_1 \wedge \neg v_2) \cdot \gamma_{V_1}(v_1) \cdot \gamma_{V_2}(\neg v_2) + \\ &+ \gamma_{V_3}(v_3 \mid \neg v_1 \wedge \neg v_2) \cdot \gamma_{V_1}(\neg v_1) \cdot \gamma_{V_2}(\neg v_2) = \\ \\ &= 0.9 \cdot 0.5 \cdot 0.4 + 0.6 \cdot 0.5 \cdot 0.4 + 0.2 \cdot 0.5 \cdot 0.6 + 0.8 \cdot 0.5 \cdot 0.6 = 0.6\end{aligned}$$

and, likewise, $\pi^{V_3}_{V_4}(\neg v_3) = 0.4$. So, $\pi_{V_4}(v_4) = 0.5 \cdot 0.6 + 0.65 \cdot 0.4 = 0.56$.
Analogously, vertex V_4 computes $\pi_{V_4}(\neg v_4) = 0.44$.
As a result we find

$$\Pr(v_4) \;=\; 0.56 \quad \text{and} \quad \Pr(\neg v_4) \;=\; 0.44$$

b. *Tip:* Re-use results from part a) where possible. Copy (part of) the graph and add the dummy vertex that enables processing the instantiation for vertex V_6 explicitly, so you cannot forget about its messages.

Vertex V_4 applies the *data fusion lemma* to compute the probabilities $\Pr^{v_6}(v_4)$ and $\Pr^{v_6}(\neg v_4)$:

$$\begin{aligned}\Pr^{v_6}(v_4) &= \alpha \cdot \pi_{V_4}(v_4) \cdot \lambda_{V_4}(v_4) \\ \Pr^{v_6}(\neg v_4) &= \alpha \cdot \pi_{V_4}(\neg v_4) \cdot \lambda_{V_4}(\neg v_4)\end{aligned}$$

Since no evidence has been entered for vertices in $\mathbf{V_4^+}$, we still have (part a.) that

$$\pi_{V_4}(v_4) = 0.56 \text{ and } \pi_{V_4}(\neg v_4) = 0.44$$

For the compound diagnostic parameter $\lambda_{V_4}(v_4)$ vertex V_4 computes the following:

$$\lambda_{V_4}(v_4) = \lambda^{V_4}_{V_5}(v_4) \cdot \lambda^{V_4}_{V_6}(v_4)$$

Since vertex V_5 is not instantiated, we have from the *identity property* that $\lambda^{V_4}_{V_5}(v_4) = 1$. For diagnostic parameter $\lambda^{V_4}_{V_6}(v_4)$ we observe that its components $\lambda_{V_6}(v_6) = 1$ and $\lambda_{V_6}(\neg v_6) = 0$ follow from the instantiation of vertex V_6 through a *dummy vertex*. As a result,

$$\begin{aligned}\lambda^{V_4}_{V_6}(v_4) &= \lambda_{V_6}(v_6) \cdot \gamma_{V_6}(v_6 \mid v_4) + + \lambda_{V_6}(\neg v_6) \cdot \gamma_{V_6}(\neg v_6 \mid v_4) = \\ &= \gamma_{V_6}(v_6 \mid v_4) = 0.6\end{aligned}$$

(Note that we left out the normalisation constant and postpone normalisation to the final data fusion step)

We thus find that $\lambda_{V_4}(v_4) = 1 \cdot 0.6 = 0.6$. Analogously, we find

$$\lambda_{V_4}(\neg v_4) = \lambda^{V_4}_{V_5}(\neg v_4) \cdot \lambda^{V_4}_{V_6}(\neg v_4) = \gamma_{V_6}(v_6 \mid \neg v_4) = 0.1$$

Filling in all the computed parameters in the data fusion lemma, we therefore find:

$$\begin{aligned}\Pr^{v_6}(v_4) &= \alpha \cdot 0.56 \cdot 0.6 = \alpha \cdot 0.336\\ \Pr^{v_6}(\neg v_4) &= \alpha \cdot 0.44 \cdot 0.1 = \alpha \cdot 0.044\end{aligned}$$

With $\alpha = \frac{1}{0.38}$ vertex V_4 finally returns

$$\begin{aligned}\Pr^{v_6}(v_4) &\approx 0.88\\ \Pr^{v_6}(\neg v_4) &\approx 0.12\end{aligned}$$

c. Entering the evidence $V_2 = \textit{false}$ cannot influence the probabilities of the values of the vertex V_6 since this vertex is already instantiated. The evidence, however, may influence the probabilities of the values of each of the other vertices: neither of V_1, V_3, V_4, and V_5 is d-separated from V_2 by the set $\{V_6\}$, the set of vertices for which evidence has been entered.

4.5

a. For computing the probabilities $\Pr(v_5)$ and $\Pr(\neg v_5)$ of its values vertex V_5 applies the *data fusion lemma*:

$$\begin{aligned}\Pr(v_5) &= \alpha \cdot \pi_{V_5}(v_5) \cdot \lambda_{V_5}(v_5)\\ \Pr(\neg v_5) &= \alpha \cdot \pi_{V_5}(\neg v_5) \cdot \lambda_{V_5}(\neg v_5)\end{aligned}$$

Since no evidence has been entered in the network as yet, we have from the *identity property* for the compound diagnostic parameter λ_{V_5} of V_5 that $\lambda_{V_5}(v_5) = 1$ and $\lambda_{V_5}(\neg v_5) = 1$. As a consequence, *no normalisation* upon data fusion is required.

For the value $\pi_{V_5}(v_5)$ of its compound causal parameter, vertex V_5 computes

$$\pi_{V_5}(v_5) = \gamma_{V_5}(v_5 \mid v_3) \cdot \pi^{V_3}_{V_5}(v_3) + \gamma_{V_5}(v_5 \mid \neg v_3) \cdot \pi^{V_3}_{V_5}(\neg v_3)$$

where

$$\begin{aligned}\pi^{V_3}_{V_5}(v_3) &= \gamma_{V_3}(v_3 \mid v_1 \wedge v_2) \cdot \pi^{V_1}_{V_3}(v_1) \cdot \pi^{V_2}_{V_3}(v_2) +\\ &\quad + \gamma_{V_3}(v_3 \mid \neg v_1 \wedge v_2) \cdot \pi^{V_1}_{V_3}(\neg v_1) \cdot \pi^{V_2}_{V_3}(v_2) +\\ &\quad + \gamma_{V_3}(v_3 \mid v_1 \wedge \neg v_2) \cdot \pi^{V_1}_{V_3}(v_1) \cdot \pi^{V_2}_{V_3}(\neg v_2) +\\ &\quad + \gamma_{V_3}(v_3 \mid \neg v_1 \wedge \neg v_2) \cdot \pi^{V_1}_{V_3}(\neg v_1) \cdot \pi^{V_2}_{V_3}(\neg v_2) =\end{aligned}$$

$\{\pi^{V_1}_{V_3}(V_1) = \pi(V_1) = \gamma_{V_1}(V_1)$ since V_1 is a *root vertex*; it has no other successors, so *causal parameter equivalence* holds; same for $\pi^{V_2}_{V_3}(V_2)\}$

$$\begin{aligned}&= \gamma_{V_3}(v_3 \mid v_1 \wedge v_2) \cdot \gamma_{V_1}(v_1) \cdot \gamma_{V_2}(v_2) +\\ &\quad + \gamma_{V_3}(v_3 \mid \neg v_1 \wedge v_2) \cdot \gamma_{V_1}(\neg v_1) \cdot \gamma_{V_2}(v_2) +\\ &\quad + \gamma_{V_3}(v_3 \mid v_1 \wedge \neg v_2) \cdot \gamma_{V_1}(v_1) \cdot \gamma_{V_2}(\neg v_2) +\\ &\quad + \gamma_{V_3}(v_3 \mid \neg v_1 \wedge \neg v_2) \cdot \gamma_{V_1}(\neg v_1) \cdot \gamma_{V_2}(\neg v_2) =\end{aligned}$$

$$= 0.2 \cdot 0.8 \cdot 0.5 + 0.6 \cdot 0.2 \cdot 0.5 + 0.5 \cdot 0.8 \cdot 0.5 + 0.1 \cdot 0.2 \cdot 0.5 = 0.35$$

and, likewise,

$$\begin{aligned}\pi^{V_3}_{V_5}(\neg v_3) &= \gamma_{V_3}(\neg v_3 \mid v_1 \wedge v_2) \cdot \gamma_{V_1}(v_1) \cdot \gamma_{V_2}(v_2) +\\ &\quad + \gamma_{V_3}(\neg v_3 \mid \neg v_1 \wedge v_2) \cdot \gamma_{V_1}(\neg v_1) \cdot \gamma_{V_2}(v_2) +\\ &\quad + \gamma_{V_3}(\neg v_3 \mid v_1 \wedge \neg v_2) \cdot \gamma_{V_1}(v_1) \cdot \gamma_{V_2}(\neg v_2) +\\ &\quad + \gamma_{V_3}(\neg v_3 \mid \neg v_1 \wedge \neg v_2) \cdot \gamma_{V_1}(\neg v_1) \cdot \gamma_{V_2}(\neg v_2) =\end{aligned}$$

$$= 0.8 \cdot 0.8 \cdot 0.5 + 0.4 \cdot 0.2 \cdot 0.5 + 0.5 \cdot 0.8 \cdot 0.5 + 0.9 \cdot 0.2 \cdot 0.5 = 0.65$$

So, $\pi_{V_5}(v_5) = 0.6 \cdot 0.35 + 0.4 \cdot 0.65 = 0.47$.
Analogously, vertex V_5 computes $\pi_{V_5}(\neg v_5) = 0.53$.

By substituting the values $\lambda_{V_5}(v_5)$, $\lambda_{V_5}(\neg v_5)$, $\pi_{V_5}(v_5)$, and $\pi_{V_5}(\neg v_5)$ in the *data fusion lemma* and eliminating the normalisation constant α we find

$$\Pr(v_5) \quad = \quad 0.47 \quad \text{and} \quad \Pr(\neg v_5) \quad = \quad 0.53$$

b. *Approach:* Chapter 4.2.2, the Ch. 4 course slides on Pearl in SCGs and the worked out example of Exercise 4.4b give examples of how to perform such computations. Also check out the tip in the answers for 4.4b! Your computations should result in $\Pr(v_5 \mid v_3) = 0.6$, $\Pr(\neg v_5 \mid v_3) = 0.4$.

c. Entering the evidence $V_3 = \mathit{true}$ cannot influence the probabilities of the values of the vertex V_4. By exploiting the d-separation criterion it is easily seen that the vertex V_4 is independent of the vertex V_3 given the empty set $\varnothing$, that is, the set of vertices for which evidence has been entered: the only chain in the digraph G between the vertices V_3 and V_4 is blocked by $\varnothing$. Since no other vertex in G is d-separated from vertex V_3 by the empty set, the evidence for V_3 may influence the probabilities of the values of all other vertices in the network.

4.6

Approach: See tips and answers for Exercises 4.4 and 4.5a

a. $\Pr(v_1 \mid v_7) = 0.6$, $\Pr(\neg v_1 \mid v_7) = 0.4$.

b. *Tip:* implement the example network in your BN software package and see what happens and whether you can explain this.

Since $\neg\langle\{V_4\} \mid \{V_7\} \mid \{V_1\}\rangle_G^d$, the probability $\Pr^{v_7,v_4}(v_1)$ can possibly differ from $\Pr^{v_7}(v_1)$. Whether the two probabilities actually differ cannot be established from graphical considerations alone and will depend on the assessment functions involved.

c. The probability of a conjunction *cannot* be *directly* computed using Pearl's algorithm, since it computes marginal or conditional distributions over a single variable only. Let's consider Pearl's algorithm as a black box[1] and explain how we can extract the necessary information for computing the probability of this conjunction. From the chain rule, we have that

$$\Pr^{v_7,v_4}(v_1 \wedge \neg v_2) = \Pr^{v_7,v_4}(v_1 \mid \neg v_2) \cdot \Pr^{v_7,v_4}(\neg v_2)$$

Both factors on the right-hand side are queries of the form $\Pr(V_i \mid \boldsymbol{e})$ that can be directly computed using a single run of Pearl's algorithm each.

However, given the independences implied by the digraph, we can do these computations more efficiently if we apply the chain rule to the vertices in a different order:

$$\begin{aligned}
\Pr^{v_7,v_4}(v_1 \wedge \neg v_2) &= \Pr^{v_7,v_4}(\neg v_2 \mid v_1) \cdot \Pr^{v_7,v_4}(v_1) \\
&= \Pr(\neg v_2 \mid v_1) \cdot \Pr^{v_7,v_4}(v_1) \\
&= \gamma_{V_2}(\neg v_2 \mid v_1) \cdot \Pr^{v_7,v_4}(v_1)
\end{aligned}$$

[1] A black box is a system which can be viewed in terms of its inputs and outputs, without any knowledge of its internal workings.

In this case one of the factors can be read directly from the network's assessment function without applying Pearl.

4.7

a. *Approach:* See tips and answers for Exercises 4.4 and 4.5a.
$\Pr(v_3 \mid v_1 \wedge \neg v_6) = 0.6$, $\Pr(\neg v_3 \mid v_1 \wedge \neg v_6) = 0.4$.

b. *Approach:* This probability of a conjunction cannot be *directly* computed using Pearl's algorithm. See answer to Exercise 4.6c. and see if you can exploit independences to increase efficiency with respect of the number of times Pearl needs to be applied and the amount of evidence to process.

c. For the same reasons as above, the probability of a disjunction *cannot* be *directly* computed from the network with Pearl's algorithm.

Approach: Use De Morgan's law to turn the disjunction into a conjunction:

$$v_2 \vee v_4 \equiv \neg\neg(v_2 \vee v_4) \equiv \neg(\neg v_2 \wedge \neg v_4)$$

and handle the conjuction as in Exercise 4.6c.

4.8

First note that for all vertices V_i with $1 \leq i < k$ there is no change in assessment function and therefore the property trivially holds, since

$$|\Pr'(V_i) - \Pr(V_i)| = 0 \leq \epsilon$$

Approach: For $i \geq k$ we can prove the property for example by mathematical induction. First show that the property holds for vertex V_k; this is the base case. Then in the induction step you assume that the property holds for some vertex V_{m-1}, $k < m \leq n$, and show that it then also holds for vertex V_m.

4.9

Note: the Suermondt & Cooper heuristic is illustrated in Section 4.2.3 in Example 4.2.21. For another example of its application, see the Ch. 4 course slides.

a. You should find either $\{V_2\}$ or $\{V_3\}$.

b. *Approach:* Use Pearl extended with loop cutset conditioning, illustrations of which can also be found in Example 4.2.18 and the Ch 4 course slides.

Tip: Draw the singly connected graph that you effectively create through instantiating the loop cutset. This means that a loop cutset vertex with k neighbours is duplicated $k-1$ times such that each of its original neighbours is linked to one of the k copies. Also don't forget to include dummy vertices for each of these copies. Finally, look carefully at this new graph and only compute the parameters that you need for determining your probabilities of interest.

Below we take $\{V_2\}$ as a loop cutset. Try using $\{V_3\}$ yourself instead: in contrast to the computations below, you will also need to compute the diagnostic message parameters

$\lambda^{V_2}_{V_3}(V_2)$ for V_3 to send to V_2. Ultimately both choices should result in $\Pr(v_5) = 0.34$ and $\Pr(\neg v_5) = 0.66$.

To compute the probabilities $\Pr(v_5)$ and $\Pr(\neg v_5)$ we use loop cutset conditioning:

$$\Pr(V_5) = \Pr(V_5 \mid v_2) \cdot \Pr(v_2) + \Pr(V_5 \mid \neg v_2) \cdot \Pr(\neg v_2)$$

First we compute the probabilities for the loop cutset, *directly from the network specification*:

$$\begin{aligned}
\Pr(v_2) &= \Pr(v_2 \mid v_1) \cdot \Pr(v_1) + \Pr(v_2 \mid \neg v_1) \cdot \Pr(\neg v_1) \\
&= \gamma_{V_2}(v_2 \mid v_1) \cdot \gamma_{V_1}(v_1) + \gamma_{V_2}(v_2 \mid \neg v_1) \cdot \gamma_{V_1}(\neg v_1) \\
&= 0.9 \cdot 0.5 + 0.7 \cdot 0.5 = 0.8; \\
\Pr(\neg v_2) &= 0.2
\end{aligned}$$

Note: In the remainder of this exercise we will drop the subscripts for the compound parameters and the assessment functions.

I We now compute $\Pr(V_5 \mid v_2)$ using Pearl's algorithm (data fusion):

$$\begin{aligned}
\Pr^{v_2}(v_5) &= \alpha \cdot \pi(v_5) \cdot \lambda(v_5) \\
\Pr^{v_2}(\neg v_5) &= \alpha \cdot \pi(\neg v_5) \cdot \lambda(\neg v_5)
\end{aligned}$$

As there is no evidence in $\mathbf{V_5^-}$, $\mathbf{V_4^-}$ and $\mathbf{V_6^-}$ we have that their compound diagnostic parameters, as well as the diagnostic message parameters sent by these nodes, are all 1 (*identity property*).

For its compound causal parameter vertex V_5 computes:

$$\begin{aligned}
\pi(v_5) = \; & \gamma(v_5 \mid v_2 \wedge v_3) \cdot \pi^{V_2}_{V_5}(v_2) \cdot \pi^{V_3}_{V_5}(v_3) + \gamma(v_5 \mid \neg v_2 \wedge v_3) \cdot \pi^{V_2}_{V_5}(\neg v_2) \cdot \pi^{V_3}_{V_5}(v_3) \\
& + \gamma(v_5 \mid v_2 \wedge \neg v_3) \cdot \pi^{V_2}_{V_5}(v_2) \cdot \pi^{V_3}_{V_5}(\neg v_3) + \gamma(v_5 \mid \neg v_2 \wedge \neg v_3) \cdot \pi^{V_2}_{V_5}(\neg v_2) \cdot \pi^{V_3}_{V_5}(\neg v_3)
\end{aligned}$$

The causal messages sent by V_2 include the 0-1 messages that it receives from its *dummy* descendant to reflect the instantion of V_2, so $\pi^{V_2}_{V_5}(v_2) = 1$ and $\pi^{V_2}_{V_5}(v_2) = 0$. The causal messages sent by V_3 reflect the probabilities of V_3 given the observations 'above V_5 in the subgraph connected to V_3'; that is $\Pr(V_3 \mid v_2)$. Following the computation rule (do this!), we then find that $\pi^{V_3}_{V_5}(v_3) = \gamma(v_3 \mid v_2)$ and $\pi^{V_3}_{V_5}(\neg v_3) = \gamma(\neg v_3 \mid v_2)$ and conclude:

$$\begin{aligned}
\pi(v_5) &= 0.3 \cdot 1 \cdot 0.5 + 0.7 \cdot 0 \cdot 0.5 + 0.2 \cdot 1 \cdot 0.5 + 0.7 \cdot 0 \cdot 0.5 = 0.25; \\
\pi(\neg v_5) &= 0.75
\end{aligned}$$

We conclude by data fusion that

$$\begin{aligned}
\Pr^{v_2}(v_5) &= \alpha \cdot 0.25 \cdot 1 = 0.25; \\
\Pr^{v_2}(\neg v_5) &= \alpha \cdot 0.75 \cdot 1 = 0.75.
\end{aligned}$$

II We now compute $\Pr(V_5 \mid \neg v_2)$ using Pearl's algorithm (data fusion):

$$\begin{aligned}
\Pr^{\neg v_2}(v_5) &= \alpha \cdot \pi(v_5) \cdot \lambda(v_5) \\
\Pr^{\neg v_2}(\neg v_5) &= \alpha \cdot \pi(\neg v_5) \cdot \lambda(\neg v_5)
\end{aligned}$$

Replacing in the above computations for case **I** each occurrence of v_2 by $\neg v_2$, we find that the causal messages sent by V_2 are $\pi^{V_2}_{V_5}(v_2) = 0$ and $\pi^{V_2}_{V_5}(v_2) = 1$. The causal messages sent by V_3 are $\Pr(V_3 \mid \neg v_2)$, that is, $\pi^{V_3}_{V_5}(v_3) = \gamma(v_3 \mid \neg v_2)$ and $\pi^{V_3}_{V_5}(\neg v_3) = \gamma(\neg v_3 \mid \neg v_2)$. We conclude that

$$\begin{aligned}
\pi(v_5) &= 0.3 \cdot 0 \cdot 0.8 + 0.7 \cdot 1 \cdot 0.8 + 0.2 \cdot 0 \cdot 0.2 + 0.7 \cdot 1 \cdot 0.2 = 0.70;\\
\pi(\neg v_5) &= 0.30
\end{aligned}$$

We conclude by data fusion that

$$\begin{aligned}
\Pr^{\neg v_2}(v_5) &= \alpha \cdot 0.70 \cdot 1 = 0.70;\\
\Pr^{\neg v_2}(\neg v_5) &= \alpha \cdot 0.30 \cdot 1 = 0.30.
\end{aligned}$$

Finally, we have all ingredients to compute our original probabilities of interest:

$$\begin{aligned}
\Pr(v_5) &= 0.25 \cdot 0.8 + 0.70 \cdot 0.2 = 0.34;\\
\Pr(\neg v_5) &= 0.75 \cdot 0.8 + 0.30 \cdot 0.2 = 0.66.
\end{aligned}$$

4.10

Note: the Suermondt & Cooper heuristic is illustrated in Section 4.2.3 in Example 4.2.21. For another example of its application, see the Ch. 4 course slides.

a. $\{V_2\}$.

b. See the elaborate answer to Exercise 4.9b, especially the tip and general approach. You should find $\Pr(v_3) = 0.3$, $\Pr(\neg v_3) = 0.7$ using the loop cutset $\{V_2\}$ from part a.

c. *Note:* the exercise does *not* ask you to actually compute the probabilities $\Pr^{v_5}(V_3)$; it asks you to describe in detail the approach you would take, where you can use Pearl's basic belief propagation algorithm as a black-box.

The requested probabilities are computed using loop cutset conditioning:

$$\begin{aligned}
\Pr^{v_5}(v_3) &= \Pr^{v_2,v_5}(v_3) \cdot \Pr^{v_5}(v_2) + \Pr^{\neg v_2,v_5}(v_3) \cdot \Pr^{v_5}(\neg v_2)\\
\Pr^{v_5}(\neg v_3) &= \Pr^{v_2,v_5}(\neg v_3) \cdot \Pr^{v_5}(v_2) + \Pr^{\neg v_2,v_5}(\neg v_3) \cdot \Pr^{v_5}(\neg v_2)
\end{aligned}$$

The probabilities $\Pr^{v_2,v_5}(v_3)$, $\Pr^{\neg v_2,v_5}(v_3)$, $\Pr^{v_2,v_5}(\neg v_3)$ and $\Pr^{\neg v_2,v_5}(\neg v_3)$ are conditioned on the loop cutset and can therefore be directly computed from the network using Pearl's algorithm. The loop cutset probabilities $\Pr^{v_5}(v_2)$ and $\Pr^{v_5}(\neg v_2)$, however, are updated recursively (see page 60) using Bayes' rule:

$$\begin{aligned}
\Pr^{v_5}(v_2) &= \alpha \cdot \Pr^{v_2}(v_5) \cdot \Pr(v_2)\\
\Pr^{v_5}(\neg v_2) &= \alpha \cdot \Pr^{\neg v_2}(v_5) \cdot \Pr(v_2)
\end{aligned}$$

where α is a normalisation constant. The prior probabilities $\Pr(V_2)$ are computed directly from the network's assessment functions (already done for part b.). To update the loop cutset probabilities the necessary probabilities $\Pr^{v_2}(v_5)$ and $\Pr^{\neg v_2}(v_5)$ can be computed using Pearl's basic algorithm, since these *are* conditioned on the loop cutset.

4.11

Note: the Suermondt & Cooper heuristic is illustrated in Section 4.2.3 in Example 4.2.21. For another example of its application, see the Ch. 4 course slides.

a. You should find $\{V_1, V_2\}$ or $\{V_2, V_3\}$.

b. *Approach:* two concepts are important to be able to answer this question:

- optimality of loop cutsets: think about how this is defined;
- the use of loop cutsets: think about why the number of values would matter in the first place. When using Pearl's algorithm enhanced with loop cutset conditioning we have to apply Pearl's basic algorithm for each loop cutset configuration. This means not only loop cutset size, but also loop cutset cardinality plays a role.

c. *Approach:* in general the heuristic tries to select vertices that are possibly part of as many loops as possible. In choosing candidates, or among candidates, think about where the cardinality can be taken into account and how.

For example, if there is more than one candidate with the same degree, then choose the one with the smallest number of values. Maybe it would be even better to choose a vertex with a lower degree if the number of values is much smaller. In any case, for the sake of optimality take into account the cardinality of the loop cutset instead of the size.

4.12

Note: the Suermondt & Cooper heuristic is illustrated in Section 4.2.3 in Example 4.2.21. For another example of its application, see the Ch. 4 course slides.

a. Several loop cutsets can be found, depending on how ties in degree are broken. E.g. possible loop cutsets found by the heuristic are $\{V_2, V_3, V_7, V_8\}$ and $\{V_3, V_7, V_{10}\}$.

A loop cutset that you won't find with the heuristic is $\{V_3, V_{10}\}$. Think about why this one is not found. *Hint:* V_{10} should then be a possible candidate in the first or second round of candidates.

b. *Approach:*

- first assume that some $\boldsymbol{C}$ is a loop cutset of some graph G. Think about what kind of properties a vertex in a loop cutset has: what kind of neighbours can it have on a cyclic chain it breaks? What happens to this cyclic chain if the outgoing arcs of this vertex, if any, are removed? Use your answers to build a concise argument for concluding that graph G' is singly connected.
- now you need to prove the opposite: if G' is singly connected then $\boldsymbol{C}$ is a loop cutset for G. Assume that G' is singly connected. Think about what happens if you start adding outgoing arcs to the vertices in $\boldsymbol{C}$. Can this result in loops? If so, what do these loops look like? What is the role of a vertex from $\boldsymbol{C}$ in such a loop? Use your answers to build a concise argument for concluding that $\boldsymbol{C}$ must be a loop cutset for graph G.

Practical use of this property: note that the Suermondt & Cooper heuristic quite efficiently returns an empty loop cutset in case it is applied to a singly connected graph. As such it can be used to verify if a graph is singly connected. The above property can then be exploited to test if a given set is a loop cutset.

c. *Approach:* think about what it means for a loop cutset to be non-minimal. Also consider how you could exploit the property you had to prove in part b. to solve this issue.

4.13

a. *Approach:* recall that independences are read from the digraph by means of d-separation. You therefore have to show that *given* V_i the two graphs encode exactly the same d-separation statements, that is: $\langle \boldsymbol{X} \mid \boldsymbol{Y} \cup \{V_i\} \mid \boldsymbol{Z}\rangle^d_G$ if and only if $\langle \boldsymbol{X} \mid \boldsymbol{Y} \cup \{V_i\} \mid \boldsymbol{Z}\rangle^d_{G'}$, for all sets of variables $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z} \subseteq \boldsymbol{V}_G$.
Tip: draw some example graphs to get an idea of what is happening!

($\Rightarrow$) Consider $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z} \subseteq \boldsymbol{V}_G$ such that $\langle \boldsymbol{X} \mid \boldsymbol{Y} \cup \{V_i\} \mid \boldsymbol{Z}\rangle^d_G$. Then, by definition, each chain between a vertex from $\boldsymbol{X}$ and a vertex from $\boldsymbol{Z}$ is blocked by a vertex from $\boldsymbol{Y} \cup \{V_i\}$. Let $\boldsymbol{X'} \subseteq \boldsymbol{X}$ and $\boldsymbol{Z'} \subseteq \boldsymbol{Z}$ be such that vertex V_i blocks the chains between each pair of vertices from $\boldsymbol{X'}$ and $\boldsymbol{Y'}$. Then V_i must have at least one outgoing arcs on each of those chains. By removing all outgoing arcs from V_i, each chain in G between $\boldsymbol{X'}$ and $\boldsymbol{Z'}$ is broken, so they stay d-separated in G'. Therefore $\langle \boldsymbol{X} \mid \boldsymbol{Y} \cup \{V_i\} \mid \boldsymbol{Z}\rangle^d_{G'}$.

($\Leftarrow$) Suppose $\langle \boldsymbol{X} \mid \boldsymbol{Y} \cup \{V_i\} \mid \boldsymbol{Z}\rangle^d_{G'}$, then adding outgoing arcs to V_i cannot introduce active chains because V_i will block all newly created chains, so $\langle \boldsymbol{X} \mid \boldsymbol{Y} \cup \{V_i\} \mid \boldsymbol{Z}\rangle^d_G$. This concludes the proof. ■

b. *Approach:* you have shown in part a. that G and G' encode the same independences *given* V_i. Therefore you should try to construct a small(!) example graph where some d-separation statement holds in G and not in G' or vice-versa, if you don't condition on V_i. Since G' has less connections and therefore vertices are more easily d-separated, it is probably easiest to construct an example such that $\langle \boldsymbol{X} \mid \boldsymbol{Y} \mid \boldsymbol{Z}\rangle^d_{G'}$ but not $\langle \boldsymbol{X} \mid \boldsymbol{Y} \mid \boldsymbol{Z}\rangle^d_G$. The smallest possible $\boldsymbol{Y}$ is the emptyset, so try if you can use that.
Tip: Always make drawings of the graphs you consider!

Consider the following digraph $G = (\boldsymbol{V}_G, \boldsymbol{A}_G)$: $V_1 \to V_2 \to V_3$ and let $G' = (\boldsymbol{V}_{G'}, \boldsymbol{A}_{G'})$ with $\boldsymbol{V}_{G'} = \boldsymbol{V}_G$ and $\boldsymbol{A}_{G'} = \boldsymbol{A}_G - (V_2, V_3)$, i.e. $V_1 \to V_2 \qquad V_3$. In this example we find

$$\neg\langle \{V_1\} \mid \emptyset \mid \{V_3\}\rangle^d_G, \quad \text{yet } \langle \{V_1\} \mid \emptyset \mid \{V_3\}\rangle^d_{G'}$$

c. *Approach:* think about how removing arcs can affect the complexity of probabilistic inference. What property of graphs makes inference NP-hard, for example?

Possible answers include the following:

- parents of vertices disappear, so the number of computations per vertex will be reduced
- the graph may become disconnected; as a result evidence may have to be propagated to less vertices

4.14

a. *Approach:* this question does *not* ask you to illustrate the use of Pearl's algorithm. This means that you can just straightforwardly compute the probabilities you need from the joint distribution defined by the network, using the standard rules of probability theory. For these small example networks that are used for exercises and exams performing straightforward computations is less elaborate.

You should find MPE($c_{\boldsymbol{E}}$)= $\neg v_2 \wedge \neg v_3 \wedge \neg v_4$ and MAP($\{V_3, V_4\}, c_{\boldsymbol{E}}$)= $\neg v_3 \wedge \neg v_4$.

b. A) is the correct answer. For an explanation think about what it is that makes probabilistic inference (which underlies these computations) hard. *Hint:*

$$\arg\max_{c_{\boldsymbol{X}}} \Pr(c_{\boldsymbol{X}} \wedge c_{\boldsymbol{E}}) = \arg\max_{c_{\boldsymbol{Y}}} \sum_{c_{\boldsymbol{Y}\setminus\boldsymbol{X}}} \Pr(c_{\boldsymbol{Y}\setminus\boldsymbol{X}} \wedge c_{\boldsymbol{X}} \wedge c_{\boldsymbol{E}})$$

The two problems are actually in different complexity classes: MPE is NP-complete and MAP is NP^{PP}-complete. (Note that the property suggesting that MAP is a subconfiguration of MPE as described in answer C happens to true for this example, but it does not hold in general! The question was phrased as a general question and not one particular to the example.)

5.1

Approach: A dense graph, with many arcs, will have vertices with many direct ancestors (parents). Think about which tasks, both in construction and use of a network, become increasingly difficult when networks have vertices with large parent sets.

Possible underlying reasons include (need further explanation!):

- computational aspects
- the number of probability assessments required
- the reliability of assessments, especially when learned from data

5.2

Note: Options can be found in the Ch. 5 course slides on fine-tuning the digraph. Possible approaches include (need explanation!):

- removing arc(s)
- reversing arc(s)
- adding vertices
- clustering several vertices into one
- if it is clearly a process in time: choose a dynamic network model

5.3

Note: Examples are discussed in the lectures and can be found in Chapter 5.1 and the Ch. 5 course slides on modelling variables. The answers below are possible answers, not necessarily the only 'correct' ones. On an exam, these would require more explanation!

a. Combinations of domain values that can occur together should be combined into a single value for the network variable.

b. Add an arc between the two vertices; part of their relation is deterministic, which should be captured in the assessment function. The parent will have a value indicating its child. Both variables will probably share some neighbours.

c. Possible solutions include:

- splitting variables: this can substantially reduce the number of probabilities if each of the variables requires a much smaller set of parents and especially children;
- reduce the number of values of a variable when the variable is involved in a large number of relations;
- reduce the number of variables (make sure this doesn't result in variables with a large number of values or large neighbour sets);
- reduce the number of arcs (very weak relations may not be important enough to include);

5.4

a. *Approach:* Possible observations that you can use to construct an example include (but should be further detailed!):

- not all relations are causal (e.g. abstractions, classifications)
- causal relations, especially in time, can result in cycles
- what exactly is a causal relation?

b. Possible reasons include (but should be further explained!):

- different population with different properties;
- different definition of variables;

c. *Approach:* think about what the assumptions underlying a disjunctive interaction are. Can you think of questions to ask a domain expert to (directly or indirectly) verify that these assumptions (almost) hold? You should assume your domain expert is a layman when it comes to BNs and probabilistic reasoning. Again: there is no single correct answer to this question. Be creative and, most importantly, motivate your answers.

d. The available information suffices for specifying a complete probability assessment function for the variable *HeartAttack*. Since the property of exception independence is satisfied and sufficient probabilities have been assessed, the model of the *leaky noisy or-gate* can be exploited. The complete assessment function is defined by

$$\begin{array}{lcl}
\gamma_H(h \mid s \wedge b \wedge c) & = & 1 - 0.95 \cdot \frac{0.4}{0.95} \cdot \frac{0.2}{0.95} \cdot \frac{0.1}{0.95} = 0.991 \\
\gamma_H(h \mid \neg s \wedge b \wedge c) & = & 1 - 0.95 \cdot \frac{0.2}{0.95} \cdot \frac{0.1}{0.95} = 0.980 \\
\gamma_H(h \mid s \wedge \neg b \wedge c) & = & 1 - 0.95 \cdot \frac{0.4}{0.95} \cdot \frac{0.1}{0.95} = 0.958 \\
\gamma_H(h \mid s \wedge b \wedge \neg c) & = & 1 - 0.95 \cdot \frac{0.4}{0.95} \cdot \frac{0.2}{0.95} = 0.916 \\
\gamma_H(h \mid \neg s \wedge \neg b \wedge c) & = & 0.9 \\
\gamma_H(h \mid \neg s \wedge b \wedge \neg c) & = & 0.8 \\
\gamma_H(h \mid s \wedge \neg b \wedge \neg c) & = & 0.6 \\
\gamma_H(h \mid \neg s \wedge \neg b \wedge \neg c) & = & 0.05
\end{array}$$

and the complementary function values.

5.5

Full assessment functions for all variables, except for variable V_4 can be established. Note that as a result of the incoherence in the information, different approaches to computing the requested probabilities can result in different numbers! In this case, it seems easiest to start from the distribution $\Pr(V_2 \wedge V_3)$ and the probabilities that can be computed from it.

5.6

Note: possible strengths and weaknesses are listed below (but would need explanation!), but by no means are intended to be exhaustive. Try to imagine yourself as knowledge engineer or as domain expert and use what you have learned in this course about probability elicitation to come up with your own answers. Be creative and, most importantly, motivate your answers.

Possible strengths include:

- domain expert can use own familiar words, relevant to domain
- rank order is natural, because expert's own

Possible weaknesses include:

- interpretation of words is known to be strongly context-dependent
- words may not cover the entire probability range

5.7

a. *Approach:* See the Ch. 5 course slides for some example computations of MDL quality.

The result of your computation, using ${}^{10}\log$, should be approximately -0.546. If you find a difference of around 0.12 you probably made the following common mistake: dividing by $N(V_2)$ rather than $N(V_1)$. Given your result: will the arc be added or not?

b. *Approach:* Note that the two measures differ in the penalty term alone. Think about situations in which the penalty term is the dominant term in the expression. Think about the conditions under which the two penalty terms are equivalent or not.

Note that N plays a role and that for larger N MDL will penalizes more than Akaike (exact value of N depends on which base of the logarithm you use). What will happen as a result?

5.8

a. There are various alternatives. Some actually fill in the missing values: EM (Expectation Maximisation); using a default value like '99'; using different defaults to distinguish between 'unknown' and 'irrelevant'. Some just use the available data: either only entire cases without missing values, or only configurations required for the entropy computations.

b. *Approach:* think about what happens to the ingredients of the MDL score for the above-mentioned alternatives (number of cases considered, occurrences of various values...).

c. Yes it makes sense. If you think about the reason for incorporating the penalty term (prevent adding too many parents to a vertex) and the reasons for which we prefer sparse graphs, you can motivate this. To adapt the term: think about what it's role is and why the number of values for vertices matter.

5.9

Approach: Start with assuming that only two of the m causes are present: v_i and v_j. When will the effect **FAIL** to occur in this situation?: if v_i and V_j are both inhibited, or if neither are inhibited. This happens with probability…? If you have established this: generalize to any number of present causes.

5.10

a. the assumption underlying the use of these methods is that assessment that are indirectly derived from a decision are more reliable than directly elicited probability estimates.

b. they are highly time consuming

c. with lotteries you always win a prize; with bets you either win or lose. The latter results e.g. in different behaviour, depending on a person's attitude towards risk.

d. it was designed to elicit large numbers of rough estimates in little time, aided by verbal cues. This enables to quickly complete the network specification. From a complete specification we can compute probabilities and hence perform sensitivity analyses to investigate if some assessments need further refinement.

5.11

a. *Approach:* the CPT for Y should represent a logical OR. The variables X_i are binary-valued, so any configuration with $c_{X_1} \vee c_{X_2} \vee c_{X_3} \equiv \top$ should give a $Y = \mathit{true}$ with probability 1.

b. *Approach:* since Y should model a logical disjunction of the X_i, it should still do so after parent divorcing. To accomplish this, both vertices H and Y should represent logical disjunctions of their parents in their assessment functions. This is accomplished in the same way as in part a.

Tip: Verify that when you marginalize out H you end up with the probabilities from part a. Parent divorcing should not change the distribution defined over Y and X_i.

c. Parent divorcing is intended to reduce the number of parents (direct ancestors) of a vertex, thereby decreasing the size of the assessment function. This has benefits both from the perspective of construction as from a computational perspective (which ones?). *Note*: this becomes more effective if parent sets are larger. A possible drawback is in including additional variables that may not have a clear interpretation in the domain.

6.1

Approach: For a.–d.: use the method with auxiliary parents and d-separation; this is easier than directly using the definition of sensitivity set!

a. $\{E_1, H_1\}$

b. $\{H_1\}$

c. $\{E_3, H_1, H_2\}$

d. $\{E_2, E_3, E_4, H_1, H_2, H_3\}$

e. There are 40 model parameters, 20 of which are free, so 200 network propagations are required if we assume 10-step variation per function (i.e. each function is evaluated in 10 different points).

f. The sensitivity set for H_2 now consists of $\{E_1, E_2, H_1, H_2\}$. For model parameters associated with H_2, the functions are linear; for the other variables the functions are hyperbolic and require no more than 3 network propagations each (verify that $\Pr(c_{E_2})$, represented in the denominator of the sensitivity function, indeeds varies with the parameter, which means we need one less constant). Then a total of 25 propagations are required.

6.2

a. B blocks the influence of varying its parameters to variables CT and SH and therefore variation has no effect on the output probabilities $\Pr(ct \mid B)$, $\Pr(sh \mid B)$, and their complements.

b. *Note:* The question basically is: for which set of observations, in addition to C, will variable ISC no longer be contained in the sensitivity set of CT. This is the case, for example, when B is observed in addition to C.

c. Linear: model parameters of ISC and C; non-linear: model parameters of MC, B and SH. Variation of model parameters of CT has no effect.

d. *Approach:* In general, you need to follow the approach from Example 6.1.2 in Chapter 6.1.2. For this specific question, however, you can save yourself a lot of work by observing that $\Pr(b) = \Pr(b \wedge mc) + \Pr(b \wedge \neg mc)$ and then using the chain rule. (See also the example sensitivity function in the Ch. 6 course slides)

$$\begin{aligned}
\Pr(b) &= \Pr(b \mid mc) \cdot \Pr(mc) + \Pr(b \mid \neg mc) \cdot \Pr(\neg mc) \\
&= \gamma(mc) \cdot p(b \mid mc) + \gamma(b \mid \neg mc) \cdot \gamma(\neg mc) \\
&= 0.2 \cdot p(b \mid mc) + 0.04
\end{aligned}$$

e. *Approach:* In general, you need to follow the approach from Example 6.1.2 in Chapter 6.1.2. That is, start with $\Pr(c \mid b \wedge mc) = \frac{\Pr(c \wedge b \wedge mc)}{\Pr(b \wedge mc)}$, then use marginalisation and the chain rule. As in d. consider carefully if you really need all the variables from the entire joint distribution.

$$\begin{aligned}
\Pr(c \mid b \wedge mc) &= \Pr(c \mid b \wedge mc \wedge isc) \cdot \Pr(isc \mid b \wedge mc) + \\
&\quad + \Pr(c \mid b \wedge mc \wedge \neg isc) \cdot \Pr(\neg isc \mid b \wedge mc) \\
&= \Pr(c \mid b \wedge isc) \cdot \Pr(isc \mid mc) + \Pr(c \mid b \wedge \neg isc) \cdot \Pr(\neg isc \mid mc) \\
&= \gamma(c \mid b \wedge isc) \cdot p(isc \mid mc) + \gamma(c \mid b \wedge \neg isc) \cdot (1 - p(isc \mid mc)) \\
&= (\gamma(c \mid b \wedge isc) - \gamma(c \mid b \wedge \neg isc)) \cdot p(isc \mid mc) + \gamma(c \mid b \wedge \neg isc) \\
&= (0.80 - 0.80) \cdot p(isc \mid mc) + 0.80 = 0.80
\end{aligned}$$

6.3

The correct choice is II; give a counter-example to show it's not order-preserving and prove that it is impossibility-preserving.

6.4

Approach: if you don't know where to start, first write $\Pr(e)$ in terms of assessment functions for small n, e.g. $n = 2$.

6.5

Note: below the variable *Heart-disease* will be represented by vertex D, where d is shorthand for *Heart-disease = yes* and $\neg d$ for *Heart-disease = no*. Likewise the variable $Test$ is represented by vertex T with its configurations denoted t and $\neg t$, respectively.

a. Draw vertex D as direct ancestor of vertex T. Then $\gamma_D(d) = \frac{1}{1000}$, $\gamma_T(t \mid d) = 1$ and $\gamma_T(\neg t \mid \neg d) = 0.95$. The complements follow.

b. First we write out the requested probability in terms of values of the assessment functions, replacing $\Pr(d)$ by x:

$$\begin{aligned}
\Pr(d \mid t) &= \frac{\Pr(t \mid d) \cdot \Pr(d)}{\Pr(t \mid d) \cdot \Pr(d) + \Pr(t \mid \neg d) \cdot \Pr(\neg d)} \\
&= \frac{1 \cdot x}{1 \cdot x + (1 - 0.95) \cdot (1 - x)} = \frac{x}{0.95 \cdot x + 0.05}
\end{aligned}$$

c. To determine the sensitivity value, first compute the first derivative of the sensitivity function:

$$\frac{d}{dx} x \cdot (0.95 \cdot x + 0.05)^{-1} = \frac{0.05 - 0}{(0.95 \cdot x + 0.05)^2}$$

The absolute value in $x_0 = 0.001$ equals 19.26, which is quite large. We could therefore say that the output is sensitive to variations in this parameter.

Verify that for $x_0 = 0.001$ $D = no$ is the most likely outcome given a positive test result. $D = yes$ is most likely when $\Pr(d \mid t) > 0.5$. Determine for which values of x this is the case to obtain the admissible deviation.

6.6

a. *Approach*: derive analytic expressions for a and b, using the approach from Exercise 6.2d,e, and argue why they can be considered constants.

b. Possible observations include

- the iso-probability lines are equi-distant which means that there are no synergistic or interaction effects
- the lines are horizontal which means that varying x does nothing; only variation in y results in changes in $\Pr(v_3)$

c. possbile benefit: 2-way analyses can provide insights into synergistic effects that you cannot get from 1-way analyses
possible drawbacks: computationally more expensive and more difficult to interpret the results of the analyses.

6.7

Hint: Let $\boldsymbol{V}_G$ be the set of vertices in digraph G, let $V_0 \in \boldsymbol{V}_G$ be the output variable of interest and let $\boldsymbol{E} \subseteq \boldsymbol{V}_G$ be the set of instantiated vertices. Let G^* be the digraph that results by adding an auxiliary parent X_i to every $V_i \in \boldsymbol{V}_G$ then argue:

> each vertex V_i for which $\langle\{X_i\} \mid \boldsymbol{E} \mid \{V_0\}\rangle^d_{G^*}$ obeys one of the following properties:
>
> a. $V_i \notin \rho^*_G(V_0)$ and $\sigma^*_G(V_i) \cap \boldsymbol{E} = \varnothing$
> b. $V_i \in \rho^*_G(V_0)$ and $\langle\{V_i\} \cup \rho_G(V_i) \mid \boldsymbol{E} \mid \{V_0\}\rangle^d_G$
> c. $V_i \notin \rho^*_G(V_0)$ and $\langle\{V_i\} \cup \rho_G(V_i) \mid \boldsymbol{E} \mid \{V_0\}\rangle^d_G$ and $\sigma^*_G(V_i) \cap \boldsymbol{E} \neq \varnothing$

6.8

a. 0.5;
b. $1 - 1/6$;
c. Brier score is 0.18 if it rains and 0.98 if it doesn't so on average you'll find (in the long run) a score of 0.42;
d. scores are 0 and 2 for a single (correct vs incorrect) prediction. The average over n scores of which m are correct is $2 - 2 \cdot \frac{m}{n}$; we expect Piet to get an average score of 0.6.

Bibliography

[Andreassen *et al.*, 1987] S. Andreassen, M. Woldbye, B. Falck, S.K. Andersen. MUNIN - A causal probabilistic network for interpretation of electromyographic findings. *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, 1987, pp. 366 – 372.

[Andreassen *et al.*, 1991] S. Andreassen, R. Hovorka, J. Benn, K.G. Olesen, E.R. Carson. A model-based approach to insulin adjustment. In: M. Stefanelli, A. Hasman, M. Fieschi, J. Talmon. *Proceedings of the Third Conference on Artificial Intelligence in Medicine.* Lecture Notes in Medical Informatics 44, Springer Verlag, Berlin, 1991, pp. 239 – 248.

[Baioletti *et al.*, 2011] M. Baioletti, G. Busanello, B. Vantaggi. Acyclic directed graphs representing independence models. *International Journal of Approximate Reasoning*, 52(1), 2011, pp. 2–18.

[Bellazzi *et al.*, 1991] R. Bellazzi, C. Berzuini, S. Quaglini, D.J. Spiegelhalter, M. Leaning. Cytotoxic chemotherapy monitoring using stochastic simulation on graphical models. In: M. Stefanelli, A. Hasman, M. Fieschi, J. Talmon. *Proceedings of the Third Conference on Artificial Intelligence in Medicine.* Lecture Notes in Medical Informatics 44, Springer Verlag, Berlin, 1991, pp. 227 – 238.

[Berzan, 2012] C Berzan. *An Exploration of Structure Learning in Bayesian Networks.* Honors Thesis, Department of Computer Science, Tufts University, 2012.

[Ben-Bassat, 1978] M. Ben-Bassat. Myopic policies in sequential classification. *IEEE Transactions on Computers*, vol. C-27, 1978, pp. 170 – 174.

[Blanco *et al.*, 2005] R. Blanco, I. Inza, M. Merino, J. Quiroga, P. Larranñaga. Feature selection in Bayesian classifiers for the prognosis of survival of cirrhotic patients treated with TIPS. *Journal of Biomedical Informatics*, vol. 38, 2005, pp. 376 – 388.

[Bolt & Renooij, 2014] J.H. Bolt, S. Renooij. Local sensitivity of Bayesian networks to multiple simultaneous parameter shifts. *Proceedings of the Seventh European Workshop on Probabilistic Graphical Models.* Lecture Notes in Artificial Intelligence, vol. 8754, Springer Verlag, 2014, pp. 65 – 80.

[Bolt & Van der Gaag, 2019] J.H. Bolt, L.C. van der Gaag. On minimum elementary-triplet bases for independence relations. In *ISIPTA 2019: Proceedings of Machine Learning Research*, vol. 103, 2019, pp. 32 – 37.

[Boneh *et al.*, 2006] T. Boneh, A. Nicholson, L. Sonenberg. Matilda: a visual tool for modelling with Bayesian networks. *International Journal of Intelligent Systems*, vol. 21, 2006, pp. 1127 – 1150.

[Boose & Gaines, 1988] J. Boose, B. Gaines. *Knowledge Acquisition Tools for Expert Systems*, Academic Press, London, 1988.

[Bouckaert, 1995] R.R. Bouckaert. *Bayesian Belief Networks: from Construction to Inference*, Ph.D. thesis, Utrecht University, 1995.

[Bruza & Van der Gaag, 1994] P.D. Bruza, L.C. van der Gaag. Index expression belief networks for information disclosure. *International Journal of Expert Systems: Research and Applications*, vol. 7, 1994, pp. 107 – 138.

[Buchanan & Shortliffe, 1984] B.G. Buchanan, E.H. Shortliffe. *Rule-based Expert Systems. The MYCIN Experiments of the Stanford Heuristic Programming Project.* Addison-Wesley, Reading, Massachusetts, 1984.

[Cheeseman, 1988] P. Cheeseman. An inquiry into computer understanding. *Computational Intelligence*, vol. 4, 1988, pp. 58 – 66.

[Chen & Pollino, 2012] S. Chen, C. Pollino. Good practice in Bayesian network modelling. *Environmental Modelling and Software*, vol. 37, 2012, pp. 134 – 145.

[Comtet, 1974] L. Comtet. *Advanced Combinatorics: The Art of Finite and Infinite Expansions*, Netherlands: Reidel, pp. 176-177, 1974.

[Cooper, 1990] G.F. Cooper. The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, vol. 42, 1990, pp. 393 – 405.

[Coupé & Van der Gaag, 1998] V.M.H. Coupé and L.C. van der Gaag. Practicable sensitivity analysis of Bayesian belief networks. *Proceedings of the Joint Session of the 6th Prague Symposium of Asymptotic Statistics and the 13th Prague Conference on Information Theory, Statistical Decision Functions and Random Processes*, Union of Czech Mathematicians and Physicists, 1998, pp. 81 – 86.

[Coupé & Van der Gaag, 2002] V.M.H. Coupé and L.C. van der Gaag. Properties of sensitivity analysis of Bayesian belief networks. *Annals of Mathematics and Artificial Intelligence*, vol. 36, 2002, pp. 323 – 356.

[Cousins *et al.*, 1993] S.B. Cousins, W. Chen, M.E. Frisse. A tutorial to stochastic simulation algorithms for inference in belief networks. *Artificial Intelligence in Medicine*, vol. 5, 1993, pp. 315 – 340.

[Cox, 1979] R.T. Cox. Of inference and inquiry — an essay in inductive logic. *The Maximum Entropy Formalism*, MIT Press, Cambridge, Massachusetts, 1970.

[Dagum & Luby, 1993] P. Dagum, M. Luby. Approximating probabilistic inference in Bayesian belief networks is NP-hard. *Artificial Intelligence*, vol. 60, 1993, pp. 141 – 153.

[Dal *et al.*, 2018] G.H. Dal, A.W. Laarman, P.J.F. Lucas. Parallel probabilistic inference by weighted model counting. In *Proceedings of Machine Learning Research*, vol. 72, 2018, pp. 97 – 108.

[Darwiche, 2000] A. Darwiche. A differential approach to inference in Bayesian networks. *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, Morgan Kaufmann, San Francisco, 2000, pp. 123 – 132.

[Dawid, 1985] A.P. Dawid. Calibration-based empirical probability. *Annals of Statistics*, vol. 13, 1985, pp. 1251 – 1274.

[De Dombal *et al.*, 1972] F.T. de Dombal, D.J. Leaper, J.R. Staniland, A.P. McCann, J.C. Horrocks. Computer-aided diagnosis of acute abdominal pain. *British Medical Journal*, vol. 2, 1972, pp. 9 – 13.

[De Dombal *et al.*, 1974] F.T. de Dombal, D.J. Leaper, J.C. Horrocks, J.R. Staniland, A.P. McCann. Human and computer-aided diagnosis of abdominal pain: further report with emphasis on the performance of clinicians. *British Medical Journal*, vol. 4, 1974, pp. 376 – 380.

[DeGroot & Fienberg, 1983] M.H. DeGroot and S.E. Fienberg. The comparison and evaluation of forecasters. *The Statistician*, vol. 32, 1983, pp. 12 – 22.

[Dempster *et al.*, 1977] A.P. Dempster, N.M. Laird, D.B. Rubin. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society, Series B* 39(1): 1 – 38, 1977.

[Druzdzel, 1996] M.J. Druzdzel. Qualitative verbal explanations in Bayesian belief networks. *Artificial Intelligence and Simulation of Behaviour Quarterly*, 94:43–54, 1996.

[Druzdzel & Van der Gaag, 1995] M.J. Druzdzel, L.C. van der Gaag. Elicitation of probabilities for belief networks: combining qualitative and quantitative information. *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, Morgan Kaufmann, 1995, pp. 141 – 148.

[Druzdzel & Van der Gaag, 2000] M.J. Druzdzel, L.C. van der Gaag. Building probabilistic networks: 'Where do the numbers come from?' Guest editors' introduction. *IEEE Transactions on Knowledge and Data Engineering*, 12(4):481-486, 2000.

[Finetti, 1970] B. de Finetti. *Theory of Probability*, Wiley, New York, 1970.

[Fletcher *et al.*, 1996] R.H. Fletcher, S.W. Fletcher, and E.H. Wagner. *Clinical Epidemiology. The Essentials*, 3rd ed, Williams & Wilkins, Baltimore, 1996.

[Friedman *et al.*, 1997] N. Friedman, D. Geiger, M. Goldszmidt, G. Provan, P. Langley, P. Smyth. Bayesian Network Classifiers, *Machine Learning*, 1997, pp. 131–163.

[Gasse & Aussem, 2016] M. Gasse, A. Aussem. Identifying the irreducible disjoint factors of a multivariate probability distribution. In: *Proceedings of the International Conference on Probabilistic Graphical Models*, JMLR: Workshop and Conference Proceedings, vol. 52, 2016, pp. 183–194.

[Geiger & Pearl, 1988] D. Geiger, J. Pearl. On the logic of causal models, *Proceedings of the Fourth Workshop on Uncertainty in Artificial Intelligence*, 1988, pp. 136 – 147.

[Geiger *et al.*, 1990] D.E. Geiger, T. Verma, J. Pearl. *d*-separation: from theorems to algorithms. In: M. Henrion, R.D. Shachter, L.N. Kanal, J.F. Lemmer. *Uncertainty in Artificial Intelligence 5*, Elsevier Science, Amsterdam, 1990, pp. 139 – 148.

[Glasziou & Hilden, 1989] P. Glasziou, J. Hilden. Test selection measures. *Medical Decision Making*, vol. 9, 1989, pp. 133 – 141.

[Gorry & Barnett, 1968] G.A. Gorry, G.O. Barnett. Experience with a model of sequential diagnosis. *Computers and Biomedical Research*, vol. 1, 1968, pp. 490 – 507.

[Guida & Tasso, 1989] G. Guida, C. Tasso. *Topics in Expert System Design*, North-Holland, Amsterdam, 1989.

[Heckerman *et al.*, 1992] D.E. Heckerman, E.J. Horvitz, B.N. Nathwani. Toward normative expert systems. Part 1: The Pathfinder project. *Methods of Information in Medicine*, vol. 31, 1992, pp. 90 – 105.

[Heckerman *et al.*, 1993] D.E. Heckerman, E.J. Horvitz, B. Middleton. An approximate nonmyopic computation for value of information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, 1993, pp. 292 – 298.

[Helsper & Van der Gaag, 2002] E.M. Helsper, L.C. van der Gaag. Building Bayesian networks through ontologies. In: F. van Harmelen (ed.). *Proceedings of the 15th European Conference on Artificial Intelligence.* IOS Press, Amsterdam, 2002, pp. 680 – 684.

[Henrion, 1989] M. Henrion. Some practical issues in constructing belief networks. In: L.N. Kanal, T.S. Levitt, J.F. Lemmer (eds). *Uncertainty in Artificial Intelligence 3*, Elsevier Science, North-Holland, 1989.

[Ilkou & Koutraki, 2020] E. Ilkou, M. Koutraki. Symbolic vs sub-symbolic AI methods: friends or enemies? In: *Proceedings of the CIKM Workshop*, CEUR Workshop Proceedings, vol. 2699, 2020.

[Jackson, 1990] P. Jackson. *Introduction to Expert Systems.* Addison-Wesley, Wokingham, 1990.

[Jensen, 1995] A.L. Jensen. Quantification experience of a DSS for mildew management in winter wheat. In: M.J. Druzdzel, L.C. van der Gaag, M. Henrion, F.V. Jensen (eds). *IJCAI-95 Workshop on Building Probabilistic Networks: Where Do the Numbers Come From ?*, 1995, pp. 23 – 31.

[Jensen, 1996] F.V. Jensen. *An Introduction to Bayesian Networks.* UCL Press, London, 1996.

[Jensen & Nielsen, 2007] F.V. Jensen & T.D. Nielsen. *Bayesian Networks and Decision Graphs*, 2nd edition. Springer Verlag, 2007.

[Jensen *et al.*, 1990] F.V. Jensen, J. Nielsen, H.I. Christensen. *Use of Causal Probabilistic Networks as High Level Models in Computer Vision.* Technical Report R-90-39, University of Aalborg, Denmark, 1990.

[Ji *et al.*, 2015] Ji Z., Xia Q., Meng G. A review of parameter learning methods in Bayesian network. In: Huang DS., Han K. (eds) *Advanced Intelligent Computing Theories and Applications. ICIC 2015.* Lecture Notes in Computer Science, vol 9227. Springer, Cham, 2015.

[Jin *et al.*] X. Jin, A. Xu, R. Bie, X. Shen, M. Yin. Spam email filtering with Bayesian belief network: using relevant words. *IEEE International Conference on Granular Computing, GrC 2006*, 2006, pp. 238–243.

[Kadane & Schum, 1996] J.B. Kadane, D.A. Schum. *A Probabilistic Analysis of the Sacco and Vanzetti Evidence.* John Wiley & Sons Inc., 1996.

[Karp *et al.*, 1989] R. Karp, M. Luby, N. Madras. Monte-Carlo approximation algorithms for enumeration problems. *Journal of Algorithms*, vol. 10, 1989, pp. 429 – 448.

[Kjærulff & Van der Gaag, 2000] U. Kjærulff and L.C. van der Gaag. Making sensitivity analysis computationally efficient. *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, Morgan Kaufmann, San Francisco, 2000, pp. 317 – 325.

[Koiter, 2006] J. R. Koiter. Visualizing inference in Bayesian networks. Master's thesis, Delft University of Technology, 2006.

[Koopman & Renooij, 2021] T. Koopmand and S. Renooij. Persuasive contrastive explanations for Bayesian networks. *Proceedings of the 16th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, Springer, 2021.

[Korb & Nicholson, 2010] K.B. Korb and A.E. Nicholson. *Bayesian Artificial Intelligence*, CRC Computer Science& Data Analysis (2nd ed.). Chapman & Hall (CRC Press).

[Korver & Lucas, 1993] M. Korver, P.J.F. Lucas. Converting a rule-based expert system into a belief network. *Medical Informatics*, vol. 18, 1993, pp. 219 – 241.

[Krak & Van der Gaag, 2014] T.E. Krak, L.C. van der Gaag. Knowledge-based bias correction: A case study in veterinary decision support. In *Proceedings of the European Conference on AI (ECAI 2014)*, IOS Press, 2014.

[Lacave & Díez, 2002] C. Lacave, F.J. Díez. A review of explanation methods for Bayesian networks. *Knowledge Engineering Review*, 17(2):107–127, 2002.

[Lacave *et al.*, 2007] C. Lacave, M. Luque, F.J. Díez. Explanation of Bayesian networks and influence diagrams in Elvira. *Systems, Man, and Cybernetics, Part B*, 37(4):952–965, 2007.

[Laskey & Mahoney, 2000] K.B. Laskey, S. Mahoney. Network engineering for agile belief network models. *IEEE Transactions on Knowledge and Data Engineering*, vol. 12(4), 2000, pp. 487 – 498.

[Lauritzen & Spiegelhalter, 1988] S.L. Lauritzen, D.J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society, Series B*, vol. 50, 1988, pp. 157 – 224.

[Lucas, 2005] P.J.F. Lucas. Bayesian network modelling through qualitative patterns. *Artificial Intelligence,* vol 163, 2005, pp. 233 – 263.

[Lucas & Van der Gaag, 1991] P.J.F. Lucas, L.C. van der Gaag. *Principles of Expert Systems.* Addison-Wesley, Wokingham, 1991.

[Madigan *et al.*, 1997] D. Madigan, K. Mosurski, and R. G. Almond. Graphical explanation in belief networks. In Journal of Computational and Graphical Statistics, 6(2):160?181, 1997.

[Meekes *et al.*, 2015] M. Meekes, S. Renooij, L.C. van der Gaag. Relevance of Evidence in Bayesian Networks In: S. Destercke, T. Denoeux (editors), *Proceedings of the Thirteenth European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU)*, Lecture Notes in Artificial Intelligence 9161, Springer, 2015, pp. 366 - 375.

[Morgan & Henrion, 1990] M.G. Morgan and M. Henrion. *Uncertainty, a Guide to Dealing with Uncertainty in Quantitative Risk and Policy Analysis.* Cambridge University Press, Cambridge, 1990.

[Murphy, 2022] K.P. Murphy. *Probabilistic Machine Learning: An introduction.* MIT Press, 2022.

[Neapolitan, 2003] R.E. Neapolitan. *Learning Bayesian Networks*, Prentice Hall, 2003.

[Panofsky & Brier, 1968] H.A. Panofsky and G.W. Brier. *Some Applications of Statistics to Meteorology.* The Pennsylvania State University, University Park, Pennsylvania, 1968.

[Pauker & Kassirer, 1980] S.G. Pauker and J.P. Kassirer. The threshold approach to clinical decision making. *New England Journal of Medicine*, vol. 302, 1980, pp. 1109 – 1117.

[Pearl, 1988] J. Pearl. *Probabilistic Reasoning in Intelligent Systems. Networks of Plausible Inference.* Morgan Kaufmann, Palo Alto, 1988.

[Pearl, 2009] J. Pearl *Causality: Models, Reasoning and Inference.* Cambridge University Press, 2nd edition, 2009.

[Pearl *et al.*, 1990] J. Pearl, D. Geiger, and T. Verma. The logic of influence diagrams, in: R.M. Oliver and J.Q. Smith (eds). *Influence Diagrams, Belief Nets and Decision Analysis*, John Wiley & Sons, 1990, pp. 67 – 86.

[Pearl & Paz, 1985] J. Pearl, A. Paz. GRAPHOIDS: a graph-based logic for reasoning about relevance relations. In: B. Du Boulay, D. Hogg, L. Steels (eds). *Advances in Artificial Intelligence 2*, 1985, North-Holland.

[Pearl & Verma, 1987] J. Pearl, T.S. Verma. The logic of representing dependencies by directed acyclic graphs, *Proceedings of the Sixth National Conference on Artificial Intelligence*, 1987, pp. 374 – 379.

[Peek & Ottenkamp, 1997] N.B. Peek, J. Ottenkamp. Developing a decision-theoretic network for a congenital heart disease, in: E. Keravnou, C. Garbay, R. Baud J. Wyatt (eds). *Proceedings of the 6th Conference on Artificial Intelligence in Medicine Europe.* Springer-Verlag, Berlin, 1997, pp. 157 – 168.

[Pitchforth & Mengersen, 2013] J. Pitchforth, K. Mengersen. A proposed validation framework for expert elicited Bayesian networks. *Expert Systems with Applications*, vol. 40, 2013, pp. 162 – 167.

[Pourret, Naim & Marcot, 2008] O. Pourret, P. Naim, B. Marcot (editors). *Bayesian Networks. A Practical Guide to Applications.*Wiley, England, 2008.

[Renooij, 2001] S. Renooij. *Qualitative Approaches to Quantifying Probabilistic Networks.* Ph.D. Thesis, Institute of Information and Computing Sciences, Utrecht University, The Netherlands, 2001.

[Renooij, 2001b] S. Renooij. Probability elicitation for belief networks: Issues to consider. *Knowledge Engineering Review*, vol. 16, 2001, pp. 255–269.

[Renooij, 2014] S. Renooij. Co-variation for sensitivity analysis in Bayesian networks: properties, consequences and alternatives. *International Journal of Approximate Reasoning*, vol. 55(4), 2014, pp. 1022–1042.

[Renooij & Van der Gaag, 2002] S. Renooij, L.C. van der Gaag. From qualitative to quantitative probabilistic networks. In: A. Darwiche, N. Friedman (eds), *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence (UAI)*, Morgan Kaufmann Publishers, San Francisco, 2002, pp. 422 – 429.

[Salmerón et al., 2018] A. Salmeróm, R. Rumí, H. Langseth, Th.D. Nielsen, A.L. Madsen. A review of inference algorithms for hybrid Bayesian networks. *Journal of Artificial Intelligence Research*, vol. 62, 2018, pp. 799 – 828.

[Sent, 2005] D. Sent. *Test-selection Strategies for Probabilistic Networks.* Ph.D. Thesis, Department of Information and Computing Sciences, Utrecht University, The Netherlands, 2005.

[Shachter, 1998] R.D. Shachter. Bayes-Ball: The rational pastime (for determining irrelevance and requisite information in belief networks and influence diagrams). In: G. F. Cooper, S. Moral (eds). *Uncertainty in Artificial Intelligence. Proceedings of the 14th Conference*, Morgan Kaufmann, San Francisco 1998, pp. 480– 487.

[Shafer, 1976] G. Shafer. *A Mathematical Theory of Evidence*, Princeton University Press, Princeton, 1976.

[Shafer & Pearl, 1990] G. Shafer, J. Pearl. *Readings in Uncertain Reasoning.* Morgan Kaufmann, Palo Alto, 1990.

[Shortliffe & Buchanan, 1984] E.H. Shortliffe, B.G. Buchanan. A model of inexact reasoning in medicine. In: B.G. Buchanan, E.H. Shortliffe. *Rule-based Expert Systems. The MYCIN Experiments of the Stanford Heuristic Programming Project.* Addison-Wesley, Reading, Massachusetts, 1984, pp. 233 – 262.

[Shwe *et al.*, 1991] M.A. Shwe, B. Middleton, D.E. Heckerman, M. Henrion, E.J. Horvitz, H.P. Lehmann, G.F. Cooper. Probabilistic diagnosis using a reformulation of the INTERNIST-1/QMR knowledge base I: the probabilistic model and inference algorithms. *Methods of Information in Medicine*, vol. 30, 1991, pp. 241 – 255.

[Sommerville, 1992] I. Sommerville. *Software Engineering*, Addison-Welsy, Wokingham, 1992.

[Stahlschmidt *et al.*, 2013] S. Stahlschmidt, H. Tausendteufel, W.K. Härdle. Bayesian networks for sex-related homicides: structure learning and prediction. *Journal of Applied Statistics* 40(6), 2013, pp. 1155–1171.

[Studený, 1989] M. Studený. Multiinformation and the problem of characterization of conditional-independence relations. *Problems Control Information Theory*, vol. 18, 1989, pp. 3 – 16.

[Studený, 1992] M. Studený. Conditional independence relations have no finite complete characterization, in: S. Kubik, J.A. Visek (eds). *Information Theory, Statistical Decision Functions and Random Processes*, Kluwer, Dordrecht, 1992, pp. 377 – 396.

[Studený, 1998] M. Studený. Bayesian networks from the point of view of chain graphs, in: G. F. Cooper, S. Moral (eds). *Uncertainty in Artificial Intelligence. Proceedings of the 14th Conference*, Morgan Kaufmann, San Francisco 1998, pp. 496– 503.

[Suermondt & Cooper, 1990] H.J. Suermondt, G.F. Cooper. Probabilistic inference in multiply connected belief networks using loop cutsets. *International Journal of Approximate Reasoning*, vol. 4, 1990, pp. 283 – 306.

[Suermondt & Cooper, 1991a] H.J. Suermondt, G.F. Cooper. Initialization for the method of conditioning in Bayesian belief networks. *Artificial Intelligence*, vol. 50, 1991, pp. 83 – 94.

[Suermondt & Cooper, 1991b] H.J. Suermondt, G.F. Cooper. A combination of exact algorithms for inference on Bayesian belief networks. *International Journal of Approximate Reasoning*, vol. 5, 1991, pp. 521 – 542.

[Suermondt, 1992] H.J. Suermondt. *Explanation in Bayesian Belief Networks.* PhD thesis, Department of Computer Science and Medicine, Stanford University, Stanford, 1992.

[Taroni *et al.*, 2006] F. Taroni, C. Aitken, P. Garbolino, A. Biedermann. *Bayesian Networks and Probabilistic Inference in Forensic Science*, Wiley & Sons, Chichester, 2006.

[Timmer, 2017] S.T. Timmer. *Designing and Understanding Forensic Bayesian Networks using Argumentation.* PhD thesis, Department of Information and Computing Sciences, Utrecht University, The Netherlands, 2017.

[Tversky *et al.*, 1982] D. Kahneman, P. Slovic, and A. Tversky. *Judgment under Uncertainty: Heuristics and Biases*, Cambridge University Press, Cambridge, 1982.

[Van der Gaag, 1994] L.C. van der Gaag. A pragmatic view of the certainty factor model. *The International Journal of Expert Systems: Research and Applications*, vol. 7, 1994, pp. 289 – 300.

[Van der Gaag *et al.*, 2018] L.C. van der Gaag, M. Baioletti, J.H. Bolt. A lattice representation of independence relations. In: *PGM 2018: Proceedings of Machine Learning Research*, vol. 72, 2018, pp. 487 – 498.

[Van der Gaag & Helsper, 2002] L.C. van der Gaag, E.M. Helsper. Experiences with modelling issues in building probabilistic networks. In: A. Go'mez-Pe'rez and V.R. Benjamins (eds.). *Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web. Proceedings of EKAW 2002.* LNAI vol. 2473, Springer-Verlag, 2002, pp. 21 – 26.

[Van der Gaag & Helsper, 2004] L.C. van der Gaag, E.M. Helsper. Defining classes of influences for the acquisition of probability constraints for Bayesian networks. In: R. Lo'pez de Ma'ntaras and L. Saitta (eds.). *Proceedings of the 16th European Conference on Artificial Intelligence (ECAI 2004).* IOS Press, Amsterdam, 2004, pp. 1101 - 1102.

[Van der Gaag & Meyer, 1996] L.C. van der Gaag, J.-J.Ch. Meyer. Characterising normal forms for informational independence. *Proceedings of IPMU'96: Information Processing and Management of Uncertainty in Knowledge-Based Systems*, 1996, pp. 973 – 978.

[Van der Gaag & Meyer, 1998] L.C. van der Gaag, J.-J.Ch. Meyer. Informational independence: models and normal forms. *International Journal of Intelligent Systems*, 13, 1998, pp. 83 – 109.

[Van der Gaag & Renooij, 2001] L.C. van der Gaag, S. Renooij. Analysing sensitivity data from probabilistic networks. *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*, Morgan Kaufmann, 2001, pp. 530 – 537.

[Van der Gaag & *et al.*, 2012] L.C. van der Gaag, S. Renooij, H.J.M. Schijf, A.R. Elbers, W.L. Loeffen. Experiences with Eliciting Probabilities from Multiple Experts. In: S.Greco, B. Bouchon-Meunier, G. Coletti, M. Fedrizzi, B. Matarazzo, R.R. Yager (eds), *Proceedings of the Fourteenth International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU)*, Communications in Computer and Information Sciences, vol. 299, Springer, Heidelberg, 2012, pp. 151 – 160.

[Van der Gaag *et al.*, 1999] L.C. van der Gaag, S. Renooij, C.L.M. Witteman, B. Aleman, B.G. Taal. How to elicit many probabilities. *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, Morgan Kaufmann, 1999, pp. 647 – 654.

[Van der Gaag *et al.*, 2002] L.C. van der Gaag, S. Renooij, C.L.M. Witteman, B.M.P. Aleman, and B.G. Taal. Probabilities for a probabilistic network: A case-study in oesophageal cancer. *Artificial Intelligence in Medicine*, vol. 25, 2002, pp. 123 – 148.

[Van der Gaag & Wessels, 1994a] L.C. van der Gaag, M.L. Wessels. Selective evidence gathering for diagnostic belief networks. *AISB Quarterly*, vol. 86, 1994, pp. 23 – 34.

[Van der Gaag & Wessels, 1994b] L.C. van der Gaag, M.L. Wessels. Multiple-disorder diagnosis with belief networks. *Proceedings of the Fifth Workshop on Principles of Diagnosis — DX'94*, 1994, pp. 343 – 351.

[Van Leersum, 2015] J. van Leersum. *Explaining the reasoning of Bayesian networks.* Master thesis, Department of Information and Computing Sciences, Utrecht University, The Netherlands, 2015.

[Vlek *et al.*, 2014] C.S. Vlek, H. Prakken, S. Renooij, B. Verheij. Building Bayesian networks for legal evidence with narratives: a case study evaluation. *Artificial Intelligence and Law*, 22(4), 2014, pp. 375 – 421.

[Waal & Van der Gaag, 2005] P.R. de Waal, L.C. van der Gaag. Stable Independence in Perfect Maps. *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence*, AUAI Press, 2005.

[Warner *et al.*, 1961] H.R. Warner, A.F. Toronto, L.G. Veasy, R. Stephenson. A mathematical approach to medical diagnosis: application to congenital heart disease. *Journal of the American Medical Association*, vol. 177, 1961, pp. 177 – 183.

[Wellman, 1990] M.P. Wellman. Fundamental concepts of qualitative probabilistic networks. *Artificial Intelligence*, vol. 44, 1990, pp. 257 – 303.

[Wieten *et al.*, 2019] R. Wieten, F. Bex, H. Prakken, S. Renooij. Constructing Bayesian network graphs from labeled arguments. *Proceedings of the Fifteenth European Conference on Symbolic and Quantitative Approaches to Rea- soning with Uncertainty (ECSQARU)*, Springer Lecture Notes in Arti

cial Intelligence vol. 11726. Springer Verlag, 2019.

[Von Winterfeldt & Edwards, 1986] D. von Winterfeldt, W. Edwards. *Decision Analysis and Behavioral Research.* Cambridge University Press, New York, 1986.

[Yap *et al.*, 2008] G.-E. Yap, A.-H. Tan, H.-H. Pang. Explaining inferences in Bayesian networks. *Applied Intelligence*, 29(3):263–278, 2008.

Index