Bayesian Networks for Expert Systems: Theory and Practical Applications

Chapter · March 2010

DOI: 10.1007/978-3-642-11688-9_20

3 authors, including:

Wim Wiegerinck
Radboud University
90 PUBLICATIONS **1,012** CITATIONS

SEE PROFILE

Bayesian Networks for Expert Systems, Theory and Practical Applications

Wim Wiegerinck, Bert Kappen, Willem Burgers

Abstract Bayesian network are widely accepted as models for reasoning with uncertainty. In this chapter we focus on models that are created using domain expertise only. After a short review of Bayesian networks models and common Bayesian network modeling approaches, we will discuss in more detail three applications of Bayesian networks. With these applications, we aim to illustrate the modeling power and flexibility of the Bayesian networks that goes beyond the standard textbook applications. The first network is applied in a system for medical diagnostic decision support. A distinguishing feature of this network is the large amount of variables in the model. The second one involves an application for petrophysical decision support to determine the mineral content of a well based on borehole measurements. This model differs from standard Bayesian networks by its continuous variables and nonlinear relations. Finally, we will discuss an application for victim identification by kinship analysis based on DNA profiles. The distinguishing feature in this application is that Bayesian networks are generated and computed on-the-fly based on case information.

Wim Wiegerinck
SNN Adaptive Intelligence, Geert Grooteplein 21, 6525 EZ Nijmegen, The Netherlands e-mail: w.wiegerinck@science.ru.nl

Bert Kappen
Radboud University Nijmegen, Donders Institute for Brain, Cognition and Behaviour, Geert Grooteplein 21, 6525 EZ Nijmegen, The Netherlands e-mail: b.kappen@science.ru.nl

Willem Burgers
SNN Adaptive Intelligence, Geert Grooteplein 21, 6525 EZ Nijmegen, The Netherlands e-mail: w.burgers@science.ru.nl

1 Introduction

In modeling intelligent systems for real world applications, one inevitably has to deal with uncertainty. This uncertainty is due to the impossibility to model all the different conditions and exceptions that can underlie a finite set of observations. Probability theory provides the mathematically consistent framework to quantify and to compute with uncertainty. In principle, a probabilistic model assigns a probability to each of its possible states. In models for real world applications, the number of states is so large that a sparse model representation is inevitable. A general class with a representation that allows modeling with many variables are the Bayesian networks [20, 14, 7].

Bayesian networks are nowadays well established as a modeling tool for expert systems in domains with uncertainty [22]. Reasons are their powerful but conceptual transparent representation for probabilistic models in terms of a network. Their graphical representation, showing the conditional independencies between variables, is easy to understand for humans. On the other hand, since a Bayesian network uniquely defines a joint probability model, inference — drawing conclusions based on observations — is based on the solid rules of probability calculus. This implies that the mathematical consistency and correctness of inference are guaranteed. In other words, all assumptions in the method are contained in model, i.e., the definition of variables, the graphical structure, and the parameters. The method has no hidden assumptions in the inference rules. This is unlike other types of reasoning systems such as e.g., Certainty Factors (CFs) that were used in e.g., MYCIN — a medical expert system developed in the early 1970s [24]. In the CF framework, the model is specified in terms of a number of if-then-else rules with certainty factors. Furthermore, the CF framework provides prescriptions how to invert and/or combine the if-then-else rules to do inference. These prescriptions contain implicit conditional independence assumptions which are not immediately clear from the model specification and has consequences in their application [13].

Probabilistic inference is the problem of computing the posterior probabilities of unobserved model variables given the observations of other model variables. For instance in a model for medical diagnoses, given that the patient has complaints x and y, what is the probability that he/she has disease z? Inference in a probabilistic model involve summations or integrals over possible states in the model. In a realistic application the number of states to sum over can be very large. In the medical example, the sum is typically over all combinations of unobserved factors that could influence the disease probability, such as different patient conditions, risk factors, but also alternative explanations for the complaints, etc. In general these computations are intractable. Fortunately, in Bayesian networks with a sparse graphical structure and with variables that can assume a small number of states, efficient inference algorithms exists such as the junction tree algorithm [14, 7].

The specification of a Bayesian network can be described in two parts, a qualitative and a quantitative part. The qualitative part is the graph structure of the network. The quantitative part consists of specification of the conditional probability tables or distributions. Ideally both specifications are inferred from data [15]. In practice,

however, data is often insufficient even for the quantitative part of the specification. The alternative is to do the specification of both parts by hand, in collaboration with domain experts. Many Bayesian networks are created in this way. Furthermore, Bayesian networks are often developed with the use of software packages such as Hugin (www.hugin.com) or Netica (www.norsys.com). These packages typically contain a graphical user interface (GUI) for modeling and an inference engine based on the junction tree algorithm for computation.

Although the networks created in this way can be quite complex, the scope of these software packages obviously has its limitations. In this chapter we discuss three models in which the standard approach to Bayesian modeling as outlined above was infeasible for different reasons: the large number of variables in the first model, the need to model continuous-valued variables in the second model, and the need to create models on-the-fly from data in the third application.

The first model has been developed for an application for medical diagnostic decision support (Promedas, in collaboration with UMC Utrecht). The main functionality of the application is to list the most probable diseases given the patient-findings (complaints, tests, physical examinations) that are entered. The system is aimed to support diagnosis in general internal medicine, covering a large medical domain with several specializations. However, a considerable level of detail at which the disease areas are modeled is essential for the system to be of practical use. For this application, this means that the model should contain 1000's of diseases and a factor 10 more of relations between diseases and findings. With such numbers of variables and relations, the standard modeling approach is infeasible.

The second model has been developed for an application for petrophysical decision support (in collaboration with SHELL E&P). The main function of this application is to provide a probability distribution of the mineral composition of a potential reservoir based on remote borehole measurements. In the underlying model, the number of variables is limited. However, variables are continuous valued. One of them represents the volume fractions of 13 minerals, and is therefore a 13-D continuous variable. Any sensible discretization in a standard Bayesian network approach would lead to a blow up of the state space. Due to nonlinearities and constraints, a Bayesian network with linear-Gaussian distributions [3] is also not a solution.

Finally, we will discuss an application for victim identification by kinship analysis based on DNA profiles (Bonaparte, in collaboration with NFI). Victims should be matched with missing persons in a pedigree of family members. In this application, the model follows from Mendelian laws of genetic inheritance and from principles in DNA profiling. Inference needs some preprocessing but is otherwise reasonably straightforward. In this application, however, the challenge is that the model structure depends on the family structure of the missing person. This structure will differ from case to case and a standard approach with a static network is obviously insufficient. In this application, modeling is implemented in the engine. The application generates Bayesian networks on-the-fly based on case information. Next, it does the required inferences for the matches.

The chapter is organized as follows. First, we will provide a short review of Bayesian networks in section 2. Next, in sections 3, 4 and 5 we will discuss the three

applications. In particular we will discuss the underlying Bayesian network models and the modeling approaches at a rather detailed level. Furthermore we will discuss the inference methods that we applied whenever they deviate from the standard junction tree approach. In section 6, we will end with discussion and conclusion.

2 Bayesian Networks

In this section, we first give a short and rather informal review of the theory of Bayesian networks (subsection 2.1). Furthermore in subsection 2.2, we briefly discuss Bayesian networks modeling techniques, and in particular the typical approach that is taken in most Bayesian network applications. We briefly discuss pro's and con's of this approach, and in particular why this approach does not work in the applications that we will discuss in the later sections.

2.1 Bayesian Network Theory

To introduce notation, we start by considering a joint probability distribution, or probabilistic model, $P(X_1,\ldots,X_n)$ of n stochastic variables $X_1,\ldots,X_n$. Variables X_j can be in state x_j. A state, or value, is a realization of a variable. We use shorthand notation

$$P(X_1 = x_1,\ldots,X_n = x_n) = P(x_1,\ldots,x_n) \tag{1}$$

to denote the probability (in continuous domains: the probability density) of variables X_1 in state x_1, variable X_2 in state x_2 etc.

A Bayesian network is a probabilistic model P on a finite directed acyclic graph (DAG). For each node i in the graph, there is a random variable X_i together with a conditional probability distribution $P(x_i|x_{\pi(i)})$, where $\pi(i)$ are the parents of i in the DAG, see figure 1. The joint probability distribution of the Bayesian network is the product of the conditional probability distributions

$$P(x_1,\ldots,x_n) = \prod_{i=1}^{n} P(x_i|x_{\pi(i)}) \; . \tag{2}$$

Since any DAG can be ordered such that $\pi(i) \subseteq 1,\ldots i-1$ and any joint distribution can be written as

$$P(x_1,\ldots,x_n) = \prod_{i=1}^{n} P(x_i|x_{i-1},\ldots,x_1) \; , \tag{3}$$

it can be concluded that a Bayesian network assumes

$$P(x_i|x_{i-1},\ldots,x_1) = P(x_i|x_{\pi(i)}) \; . \tag{4}$$

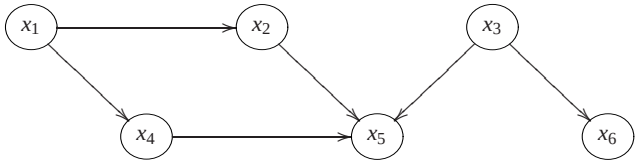

Fig. 1 DAG representing a Bayesian network $P(x_1)P(x_2|x_1)P(x_3)P(x_4|x_1)P(x_5|x_2,x_3,x_4)P(x_6|x_3)$

In other words, the model assumes: given the values of the direct parents of a variable X_i, this variable X_i is independent of all its other predecessing variables in the graph.

Since a Bayesian network is a probabilistic model, one can compute marginal distributions and conditional distributions by applying the standard rules of probability calculus. For instance, in a model with discrete variables, the marginal distribution of variable X_i is given by

$$P(x_i) = \sum_{x_1} \dots \sum_{x_{i-1}} \sum_{x_{i+1}} \dots \sum_{x_N} P(x_1, \dots, x_N) \; . \tag{5}$$

Conditional distributions such as $P(x_i|x_j)$ are obtained by the division of two marginal distributions

$$P(x_i|x_j) = \frac{P(x_i, x_j)}{P(x_j)} \; . \tag{6}$$

The bottleneck in the computation is the sum over combinations of states in (5). The number of combinations is exponential in the number of variables. A straightforward computation of the sum is therefore only feasible in models with a small number of variables. In sparse Bayesian networks with discrete variables, efficient algorithms that exploit the graphical structure, such as the junction tree algorithm [16, 14, 7] can be applied to compute marginal and conditional distributions. In more general models, exact inference is infeasible and approximate methods such as sampling have to be applied [17, 3].

2.2 Bayesian Network Modeling

The construction of a Bayesian network consists of deciding about the domain, what are the variables that are to be modeled, and what are the state spaces of each of the variables. Then the relations between the variables have to be modeled. If these are to be determined by hand (rather than by data), it is a good rule of thumb to construct a Bayesian network from cause to effect. Start with nodes that represent independent root causes, then model the nodes which they influence, and so on until we end at the leaves, i.e., the nodes that have no direct influence on other nodes.

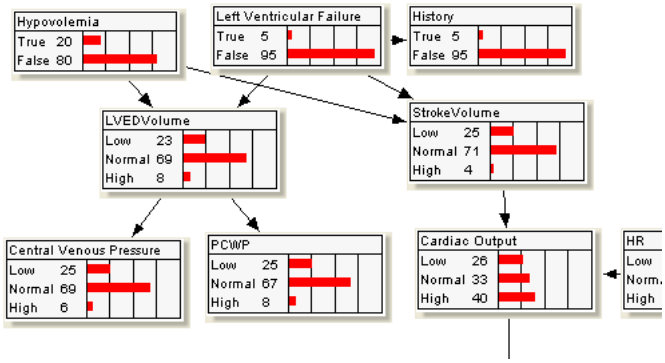

Fig. 2 Screen shot of part of the 'Alarm network' in the BayesBuilder GUI

Such a procedure often results in sparse network structures that are understandable for humans [22].

Often, models are constructed using Bayesian network software such as the earlier mentioned packages. With the use of a graphical user interface (GUI), nodes can be created. The nodes represent the variables in the system. Typically, variables can assume only values from a finite set. When a node is created, it can be linked to other nodes, under the constraint that there are no directed loops in the network. Finally — or during this process — the table of conditional probabilities are defined, often by educated guesses, and sometimes inferred from data. Many Bayesian networks that are found in literature fall into this class, see e.g., www.norsys.com/netlibrary/. In figure 2, a part of the ALARM network as represented in BayesBuilder (www.snn.ru.nl/) is plotted. The ALARM network was originally designed as a network for monitoring patients in intensive care [2]. It consists of 37 variables, each with 2, 3, or 4 states. It can be considered as a relatively large member of this class of models. An advantage of the GUI based approach is that a small or medium sized Bayesian network, i.e., with up to a few dozen of variables, where each variable can assume a few states, can be developed quickly, without the need of expertise on Bayesian networks modeling or inference algorithms.

In the next sections we will discuss three Bayesian networks for real world applications that fall outside the class of models that have been built using these modeling tools. The main reason is that the graphical user interface has no added value for these models. The first model is too complex, and would contain too many variables for the GUI. In the second one the complexity is more in the variables themselves than in the network structure. In the third model, the network consists of a few types of nodes that have simple and well defined relations among each other. However, for each different case in the application, a different network has to be generated. It does not make sense for this application to try to build these networks beforehand in a GUI.

3 Promedas, a Probabilistic Model for Medical Diagnostic Decision Support

Modern-day medical diagnosis is a very complex process, requiring accurate patient data, a profound understanding of the medical literature and many years of clinical experience. This situation applies particularly to internal medicine, because it covers an enormous range of diagnostic categories. As a result, internal medicine is differentiated in super-specializations.

Diagnosis is a process, by which a doctor searches for the cause (usually a disease) that best explains the symptoms of a patient. The search process is sequential, in the sense that patient symptoms suggest some initial tests to be performed. Based on the outcome of these tests, a tentative hypothesis is formulated about the possible cause(s). Based on this hypothesis, subsequent tests are ordered to confirm or reject this hypothesis. The process may proceed in several iterations until the patient is finally diagnosed with sufficient certainty and the cause of the symptoms is established.

A significant part of the diagnostic process is standardized in the form of protocols. These are sets of rules that prescribe which tests to perform and in which order, based on the patient symptoms and previous test results. These rules form a decision tree, whose nodes are intermediate stages in the diagnostic process and whose branches point to additional testing, depending on the current test results. The protocols are defined in each country by a committee of medical experts.

In the majority of the diagnoses that are encountered, the guidelines are sufficiently accurate to make the correct diagnosis. For these ”routine” cases, a decision support system is not needed. In 10–20 % of the cases, however, the diagnostic process is more difficult. As a result of the uncertainty about the correct diagnosis and about the next actions to perform, the decisions made by different physicians at different stages of the diagnostic process do not always agree and lack ”rationalization”. In these cases, normally a particularly specialized colleague or the literature is consulted. For these difficult cases computer based decision support may serve as an alternative source of information. In addition, a computer aided decision support system can be of help by pointing to alternative diagnoses that may be overlooked otherwise. It may thus result in an improved and more rationalized diagnostic process, as well as higher efficiency and cost-effectiveness.

Since 1996, SNN and UMC Utrecht have been developing a clinical diagnostic decision support system for internal medicine, called Promedas. In this system, patient information, such as age and gender, and findings, such as symptoms, results from physical examination and laboratory tests can be entered. The system then generates patient-specific diagnostic advice in the form of a list of likely diagnoses and suggestions for additional laboratory tests that may be relevant for a selected diagnosis.

The system is intended to support diagnostics in the setting of the outpatient clinic and for educational purposes. Its target users are general internists, super specialists (e.g., endocrinologists, rheumatologists, etc.), interns and residents, medical

students and others working in the hospital environment. Currently, a trial version of the program is installed at department of internal medicine in UMC Utrecht. It contains about 3500 diagnoses and is based on 50000 relations. The program is connected to the electronic patient records, so that physicians can easily consult the program without having to enter all the data manually. A live demo can be found on www.promedas.nl

Promedas is based on a Bayesian network. In the remainder of the section we will describe the model in further detail. We focus on the modeling part, including certain modeling approaches, model choices and methods to facilitate inference. Medical details of the model are outside the scope of this section.

3.1 Building Large Scale Probabilistic Models

For this application, in which rare diseases play an important role, data is insufficient to train the model. When modeling a Bayesian network by hand, the standard procedure is to specify a network structure of local interactions and to specify those probabilities that are needed to define these interactions quantitatively. For medium sized networks (up to 50 – 100 variables), this is doable using the methodology and Bayesian network software tools such as discussed in subsection 2.2. However, our aim was to scale up the system to 1000's of variables. For larger systems it is more difficult to keep overview, and not to get lost in the spaghetti of relations and interactions. In addition, available medical knowledge is in general limited to bivariate relations between disease and test in terms of sensitivity and specificity. Therefore we decided to take a more structured approach, in which we assume a generic structure of the model. The general assumption in this structure is that risk factors influence the probabilities of diseases and that diseases influence the probabilities of findings (symptoms, tests etc.). We furthermore restrict to models in which the parameters can be determined from the available medical knowledge of bivariate relations. In order to further facilitate modeling we have developed a database in which medical specialists can enter their knowledge in a structured and not too complicated way.

In the following, we sketch the structure of the database. Then we sketch how the Bayesian network is defined and which model choices we have made. Finally we sketch how a differential diagnosis is computed in this model.

3.1.1 Database Structure

The database contains information from which the structure of the network can be derived as well as its model parameters. In addition, the database contains meta-information, such as information about the structure of Promedas' graphical user interface. This involves mainly the grouping and naming of findings and risk factors into medical relevant categories such as complaints, physical examination, medication, lab results and subdivisions of these. In addition descriptions, annotations and

references are included. In the remainder of this subsection, however, we restrict to information that is directly relevant for the computational model.

The database contains three types of variables:

1. *Risk factors* such as occupation, drug use, past and concurrent diseases; Risk factors are coded binary (true=1/false=0).
2. *Diagnoses* such as current diseases, syndromes, drug side effects, pregnancy; Diagnoses are coded binary (true=1/false=0).
3. *Tests* or findings, such as lab tests, symptoms, physical examination etc. Tests are binary or multinomial (decreased/normal/increased/strongly increased, etc.). When the discretization is not obvious because the test is continuous by nature, then the discretization is defined in the database with cut-off points according to medical standards where possible. Discretization may depend on gender and age. The state space of the tests is such that there is always one "normal" state. Binary variables are defined such that false is the "normal" state.

Furthermore, the database contains quantifications. These are needed to model the probabilities in the Bayesian network. Quantifications can apply to single variables, and to relations between variables. Relations can be defined between risk factors and diagnoses and between tests and diagnoses. Relations can only be defined for non-normal states, e.g., between diagnosis d being true and test t being in "increased" state. The idea behind this is that relations code positive influences. The absence of the relation between diagnosis d being true and test t in "normal" state implies the assumption that the mere presence of a disease will never make the result of a test more likely to be normal than without the disease being present.

The database contains four types of quantifications:

1. *Priors.* For each diagnosis d there is prior p_d. This is the prior probability of diagnosis d being true in absence of all risk factors.
2. *Leaks.* For each test there is a so-called leak $p_{t=s}$ of each non-normal test-state. This leak is roughly interpreted as the prior probability of the test being in state $t = s$ in absence of all diagnoses. In an ideal test, the results is normal in absence of diagnoses, so any non-normal state has zero probability. In non-ideal tests, a leak causes positive probabilities of non-normal test states. Leaks are used e.g., to model the probability of a test being positive without apparent cause.
3. *Mult-factors.* For each risk–diagnosis relation there is a "mult-factor" m_{dr} by which the odds of the prior probability of diagnosis d are multiplied in the presence of the risk factor r.
4. *Senses.* For each test–diagnosis relation there is one or more "sens" $p_{dt=s}$. A sens relates a diagnosis to a non-normal test-state. This is the probability that the presence of the disease d causes the test t to be in state s (rather than the leak or other diseases). The "'sens" is closely related to sensitivity, the probability of a positive test given the presence of the disease d (regardless the leak or other diseases).

These quantifications can be age and gender dependent.

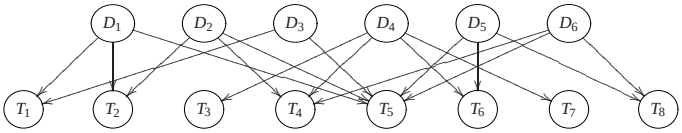

Fig. 3 Network structure in the Promedas model.

3.1.2 Network Definition

The global architecture of the diagnostic model is described by a diagnosis-layer that is connected to a layer with tests. The main assumption is that different diagnoses can coexist. Note that there are no nodes for gender, age and risk-factors. These are assumed to be observed. All other probabilities in the network are conditioned on these observations (as in e.g., (8), below). Default case is a male of 55 with all the risk-factors false. The global architecture of Promedas is similar to the QMR-DT network [25]. QMR stands for Quick Medical Reference, which is a heuristic representation with about 600 diseases and 4000 findings. The QMR-DT network, where DT stands for Decision Theoretic, is a reformulation as a two-layer Bayesian network. Main differences with Promedas are the absorption of risk factors, and the modeling of multi-valued tests in Promedas rather than the binary tests in QMR-DT. Furthermore, Promedas is based on a different knowledge base.

Diagnoses are modeled as a priori independent binary variables. Their prior probabilities (in absence of risk factors) are read from the database. In the case that a risk factor is set to true, $r = 1$, the prior of a related diagnosis is affected according to a multiplication of prior odds,

$$\frac{P(d=1|r=1)}{P(d=0|r=1)} = m_{dr} \frac{P(d=1|r=0)}{P(d=1|r=0)} , \tag{7}$$

where m_{rd} is the "mult-factor" of risk factor r in relation to diagnosis d. This implies, after rearranging terms

$$P(d=1|r=1) = \frac{m_{rd} P(d=1|r=0)}{1 + (m_{rd}-1)P(d=1|r=0)} . \tag{8}$$

The conditional distributions of tests are modeled using so-called noisy-OR and noisy-MAX gates [21]. Both will be explained below in more detail. The motivation to use these table parameterizations is that they are convenient to model because there is only one (or a few) parameter(s) for each diagnosis–test relation (rather than exponentially many as in the free form table), while on the other hand they provide a medically reasonable model that is easy to interpret [25]. An other important reason is that inference is efficient [27] as we will discuss later in this section.

To construct the noisy-OR and noisy-MAX, we first consider the deterministic OR-gate $OR(v|u_0,\ldots,u_n)$. Here, v and u_i are binary variables.

$$OR(v|u_0,\ldots,u_n) = \begin{cases} 1 & \text{if } v = \max(u_0,\ldots,u_k) \\ 0 & \text{otherwise} \end{cases} . \tag{9}$$

So $v = 1$ (true) if any of the u_i's is 1. Otherwise $v = 0$. Now the noisy-OR gate is modeled as follows (v, u_i and d_i are binary),

$$NoisyOR(v|d_1,\ldots d_n) = \sum_{\{u_0,\ldots,u_n\}} OR(v|u_0,\ldots,u_n) \prod_{i=1}^{n} P(u_i|d_i)P(u_0) . \tag{10}$$

The variables $u_0,\ldots,u_n$ can be considered as latent or auxiliary variables in this model. Furthermore, the probabilities $P(u_i = 1|d_i = 0)$ are zero in this model. The probability $P(u_0 = 1)$ is often called the 'leak'. The interpretation is that noisy-OR is a noisy version of the deterministic OR, in which there is a finite probability that (1) although all inputs $d_i = 0$, the outcome is $v = 1$ due to the leak, and (2) although there are inputs $d_i = 1$, the outcome is $v = 0$ due to the fact that $P(u_i = 0|d_i = 1)$ is non-zero. However, the more inputs $d_i = 1$, the higher the probability that the outcome is $v = 1$. In Promedas, noisy-ORs are applied for binary tests: d_i are the disease states and v is the test result. The more diseases are present, the higher the probability of a positive test result. The required probabilities to model the noisy-ORs are read from the database (leaks and senses).

Now we will construct noisy-MAX. The idea is similar as the noisy-OR gate, with in addition a winner-take-all mechanism. The idea is that if some diseases cause a test result to have a slightly increased value, and other diseases cause a test result to have a strongly increased value, the observed test result will be strongly increased. To proceed, we order the states of the test $s_0 < s_1 < \ldots s_K$, where "normal" has the lowest order (so $s_0 =$ "normal"). Next, to model diseases causing the test result to have a certain value, we define a noisy-OR gate NOR_j for each of the test-values $s_j > s_0$ (except for the "'normal" value, since diagnoses cannot cause values to be normal). The outcome of a noisy-OR gates is either 1 or 0. The outcomes of NOR_j are relabeled ($0 \to s_0$ and $1 \to s_j$) and the result is either s_0 or the value s_j.

The winner take all mechanism is modeled by the deterministic MAX-gate $MAX(t|v_1,\ldots,v_n)$. The variable t can assume all the potential values of its parent variables. The MAX-gate is defined as

$$MAX(t|v_1,\ldots,v_n) = \begin{cases} 1 & \text{if } t = \max(v_1,\ldots,v_k) \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

Noisy-MAX tables for tests $P(t|d_1,\ldots d_n)$ can be represented by NOR_j's for each of the test-values s_j, having subsets $d_{j1},\ldots,d_{jn_j}$ of diagnoses that are related to test-state $t = s_j$ as parents, combined with a deterministic MAX-gate for the winner-take-all mechanism (see figure 3),

$$P(t|d_1,\ldots d_n) = \sum_{\{v_1,\ldots,v_K\}} MAX(t|v_1,\ldots,v_k) \prod_{j=1}^{K} NOR_j(v_j|d_{j1},\ldots,d_{jn_j}) . \tag{12}$$

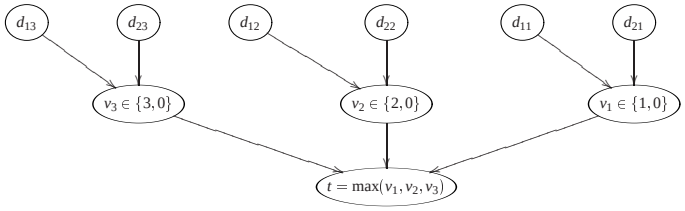

Fig. 4 Test t with ordered states $0 < 1 < 2 < 3$ are modeled as a noisy-MAX, which can itself be modeled as the MAX of the outcomes of three noisy-OR gates. In this example, diagnoses d_{ij} are connected to binary noisy-OR gates NOR_j. The outcome of a noisy-OR gate is either 1 or 0. The outcomes of NOR_j are relabeled $(0/1 \to 0/j)$ and subsequently fed into a MAX gate, which returns the maximum value.

The interpretation of the noisy-MAX model is as follows. Each of the diseases has a probability to trigger the test to be in a certain state, regardless of the presence or absence of other diseases. If different diseases have a probability to trigger the test to be in the same state, then a combination of them makes this state more likely. If different diseases trigger the test to be in different states, then the strongest state is observed. For instance if one disease triggers the body temperature to be 'increased' and another triggers the temperature to be 'strongly increased', then the model assumption is that the 'strongly increased' temperature will be observed. A drawback may be that many causes of an 'increased' temperature would in reality have an additive effect. Other models could be designed to incorporate such effect. However, such models would lack the crucial computational efficiency of the noisy-MAX model. Another issue that one could discuss is what to do with tests that have positive and negative states, such as 'decreased','normal', 'increased'. Again, other models could be designed to better incorporate the combination of a 'decreased' and an 'increased' effect, but this would also be at the expense of computational efficiency. In Promedas, we decided to be pragmatic and enforce an ordering.

3.2 Inference

The main inference task in the application is to compute the probabilities of diagnoses given the observed values of tests and risk factors. In general, inference would be computationally infeasible for networks of the size of Promedas. Therefore simplifying assumptions are introduced to make the inference task cheaper. One assumption is that all risk factors are assumed to be observed (in the application, their default value is false). This excludes any uncertainty in these variables. In this way, there will be no correlations between diagnoses through risk factors. Another simplification is to take only diagnoses into account which are connected to at least one test-node that is observed to be in a non-normal state. Other diagnoses are not of interest in the task of supporting the physician.

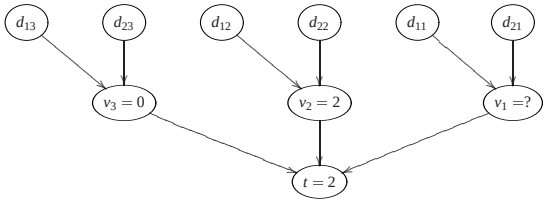

Fig. 5 Inference with noisy-MAX. Observed test value $t = 2$ implies that the outcome of $v_3 = 0$, and $v_2 = 2$. The observed test value does not give any information about v_1.

3.2.1 Efficient Inference in Noisy-MAX

Another assumption is the noisy-MAX model. As we mentioned earlier, one of the reasons to adopt this model is that inference is more efficient. There are a several properties of this model that make inference more efficient than in most other conditional probability models. See e.g. [27] for a more detailed and exposure of a general class of such models.

- *Decoupling of the parents of MAX.* If we apply the max operator over a set of variables v_i, where each v_i can have either value s_0 or s_i, with $s_0 < \ldots < s_K$, then an outcome $\max(v_1, \ldots, v_K) = s_j$ implies that all $v_k = s_0$ for $k > j$. Furthermore $v_j = s_j$ if $s_j > s_0$. The outcome does not contain any information about the variables v_k with $k < j$. See figure 5. This implies that we can take out the factor $MAX(t|v_1, \ldots, v_k)$ and decouple the intermediate variables as follows,

$$P(t = s_j|d_1, \ldots d_n) = \prod_{k=j+1}^{K} NOR_k(v_k = s_0|d_{k1}, \ldots, d_{kn_k}) \times NOR_k(v_j = s_j|d_{j1}, \ldots, d_{jn_j}) \prod_{k=1}^{j-1} \sum_{\{v_k\}} NOR_j(v_k|d_{k1}, \ldots, d_{kn_j}) \quad (13)$$

- *Decoupling of the parents of OR with outcome 'false'.* A related property is that observing that a variable modeled by a noisy-OR gate is equal to be zero, $v = 0$, implies that all states of the intermediate nodes in the noisy-OR $u_0, \ldots, u_n$ are zero. In other words, these can be considered as observed. We can remove the factor $OR(v = 0|u_0, \ldots, u_n)$ and decouple the diagnoses in (10),

$$NoisyOR(v = 0|d_1, \ldots d_n) = \prod_{i=1}^{n} P(u_i = 0|d_i) P(u_0 = 0) \,. \quad (14)$$

- *Undirected links of OR with outcome 'true'.* Straightforward expansion of OR leads to

$$OR(v = 1|u_0, \ldots, u_n) = 1 - \prod_{i=0}^{n} \delta_{u_i 0} \,. \quad (15)$$

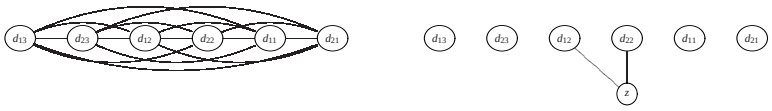

Fig. 6 Inference with noisy-MAX. Graphical structure of the undirected (moral) graph on the diagnoses which results from absorbing the evidence of observed test value $t = 2$. Left: with noisy-MAX modeled as a free form conditional probability table all the parents are connected. Right: exploiting the structure of noisy-MAX, results in a much more sparse representation. z is the auxiliary switch variable, see text.

In order to rewrite this expression, we define the auxiliary potential ψ

$$\psi(u_0, z=0) = -\delta_{u_0 0} \,, \tag{16}$$
$$\psi(u_i, z=0) = \delta_{u_i 0} \quad \text{for } i > 0, \tag{17}$$
$$\psi(u_i, z=1) = 1 \,, \tag{18}$$

where z is an auxiliary switch variable. Note that $\psi(u_0, z=0)$ is negative! With these potentials, we can decompose the *OR* as a sum-product,

$$OR(v=1|u_0,\ldots,u_n) = \sum_{\{z\}} \prod_{i=0}^{n} \psi(u_i, z) \,, \tag{19}$$

and hence, using now the auxiliary potentials defined by

$$\phi(z=0) = P(u_0=1) - 1 \,, \tag{20}$$
$$\phi(z=1) = 1 \,, \tag{21}$$
$$\phi(d_i, z=0) = 1 - P(u_i=1|d_i) \,, \tag{22}$$
$$\phi(d_i, z=1) = 1 \,, \tag{23}$$

the noisy-OR decomposes as

$$NoisyOR(v=1|d_1,\ldots,d_n) = \sum_{\{z\}} \phi(z) \prod_{i=1}^{n} \phi(d_i, z) \,. \tag{24}$$

The use of these potentials in general lead to a much smaller clique-size in the junction tree algorithm, see figure 6.

Inference in Promedas is now performed as follows. Given a set of test values, the diagnoses nodes that are related to at least one non-normal test value are selected. For these diagnoses, the present risk-factors and the evidences of the test-state-variables v_j are collected. The risk-factors and test-state-variables in normal state $v_j = s_0$ are directly absorbed in the priors of diagnoses using the mult factors and the senses in the database. The non-trivial part of the computation are the test-state-variables in non-normal state $v_j = s_j$ that are created for each non-normal

test value $t = s_j$. For these variables, undirected noisy-OR structures as in (24) are constructed using senses and leaks from the database. Standard junction tree algorithm is applied to the resulting undirected model (note that in undirected graphs, there is no coupling of the parents as preprocessing for the junction tree algorithm. In *directed* graphs, there is. This coupling is known as moralization and leads to larger cliques). The posterior probabilities of the selected diagnosis are computed and reported as the differential diagnosis (a list of the most probable diagnoses) for the case at hand.

3.3 The current application

Promedas has been further developed by Promedas B.V. Additional methods to further speed up have been implemented. However, these are outside the scope of this paper. A live demo can be found on www.promedas.nl.

3.4 Summary

Promedas is an application for medical diagnostic decision support. Its primary aim is to find a differential diagnosis based on test results (anamnesis, physical examination, lab -tests, etc.) . Given the large number of variables, a conventional Bayesian network approach is infeasible. We took a knowledge base approach in which the network is compiled from a database of relations provided by medical experts. To make computation feasible, we designed a tractable model parameterization.

4 A Petrophysical Decision Support System

Oil and gas reservoirs are located in the earth's crust at depths of several kilometers, and when located offshore, in water depths of a few meters to a few kilometers. Consequently, the gathering of critical information such as the presence and type of hydrocarbons, size of the reservoir and the physical properties of the reservoir such as the porosity of the rock and the permeability is a key activity in the oil and gas industry.

Pre-development methods to gather information on the nature of the reservoirs range from gravimetric, 2D and 3D seismic to the drilling of exploration and appraisal boreholes. Additional information is obtained while a field is developed through data acquisition in new development wells drilled to produce hydrocarbons, time-lapse seismic surveys and in-well monitoring of how the actual production of hydrocarbons affects physical properties such as the pressure and temperature. The purpose of information gathering is to decide which reservoirs can be developed

economically, and how to adapt the means of development best to the particular nature of a reservoir.

The early measurements acquired in exploration, appraisal and development boreholes are a crucial component of the information gathering process. These measurements are typically obtained from tools on the end of a wireline that are lowered into the borehole to measure the rock and fluid properties of the formation. Their is a vast range of possible measurement tools [23]. Some options are very expensive and may even risk other data acquisition options. In general acquiring all possible data imposes too great an economic burden on the exploration, appraisal and development. Hence data acquisition options must be exercised carefully bearing in mind the learnings of already acquired data and general hydrocarbon field knowledge. Also important is a clear understanding of what data can and cannot be acquired later and the consequences of having an incorrect understanding of the nature of a reservoir on the effectiveness of its development.

Making the right data acquisition decisions, as well as the best interpretation of information obtained in boreholes forms one of the principle tasks of petrophysicists. The efficiency of a petrophysicist executing her/his task is substantially influenced by the ability to gauge her/his experience to the issues at hand. Efficiency is hampered when a petrophysicists experience level is not yet fully sufficient and by the rather common circumstance that decisions to acquire particular types of information or not must be made in a rush, at high costs and shortly after receiving other information that impact on that very same decision. Mistakes are not entirely uncommon and almost always painful. In some cases, non essential data is obtained at the expense of extremely high cost, or essential data is not obtained at all; causing development mistakes that can jeopardize the amount of hydrocarbon recoverable from a reservoir and induce significant cost increases.

The overall effectiveness of petrophysicists is expected to improve using a decision support system (DSS). In practice a DSS can increase the petrophysicists' awareness of low probability but high impact cases and alleviate some of the operational decision pressure.

In cooperation with Shell E&P, SNN has developed a DSS tool based on a Bayesian network and an efficient sampler for inference. The main tasks of the application is the estimation of compositional volume fractions in a reservoir on the basis of measurement data. In addition it provides insight in the effect of additional measurements. Besides an implementation of the model and the inference, the tool contains graphical user interface in which the user can take different views on the sampled probability distribution and on the effect of additional measurements. The tool is currently under evaluation within Shell E&P.

In the remainder of this section, we will describe the Bayesian network approach for the DSS tool. We focus on our modeling and inference approach. A more detailed description of the model, in particular in relation to the petrophysical relevant quantities will be published elsewhere [5].

4.1 Probabilistic modeling

The primary aim of the model is to estimate the compositional volume fractions of a reservoir on the basis of borehole measurements. Due to incomplete knowledge, limited amount of measurements, and noise in the measurements, there will be uncertainty in the volume fractions. We will use Bayesian inference to deal with this uncertainty.

The starting point is a model for the probability distribution $P(\mathbf{v},\mathbf{m})$ of the compositional volume fractions $\mathbf{v}$ and borehole measurements $\mathbf{m}$. A causal argument *"The composition is given by the (unknown) volume fractions, and the volume fractions determine the distribution measurement outcomes of each of the tools"* leads us to a Bayesian network formulation of the probabilistic model,

$$P(\mathbf{v},\mathbf{m}) = \prod_{i=1}^{Z} P(m_i|\mathbf{v})P(\mathbf{v}) \; . \tag{25}$$

In this model, $P(\mathbf{v})$ is the so-called *prior*, the prior probability distribution of volume fractions before having seen any data. In principle, the prior encodes the generic geological and petrophysical knowledge and beliefs [26]. The factor $\prod_{i=1}^{Z} P(m_i|\mathbf{v})$ is the *observation model*. The observation model relates volume fractions $\mathbf{v}$ to measurement outcomes m_i of each of the Z tools i. The observation model assumes that *given* the underlying volume fractions, measurement outcomes of the different tools are independent. Each term in the observation model gives the probability density of observing outcome m_i for tool i given that the composition is $\mathbf{v}$. Now given a set of measurement outcomes $\mathbf{m}^o$ of a subset *Obs* of tools, the probability distribution of the volume fractions can be updated in a principled way by applying *Bayes' rule*,

$$P(\mathbf{v}|\mathbf{m}^o) = \frac{\prod_{i\in Obs} P(m_i^o|\mathbf{v})P(\mathbf{v})}{P(\mathbf{m}^o)} \; . \tag{26}$$

The updated distribution is called the *posterior* distribution. The constant in the denominator $P(\mathbf{m}^o) = \int_{\mathbf{v}} \prod_{i\in Obs} P(m_i^o|\mathbf{v})P(\mathbf{v})\mathrm{d}\mathbf{v}$ is called the *evidence*.

In our model, $\mathbf{v}$ is a 13 dimensional vector. Each component represents the volume fraction of one of 13 most common minerals and fluids (water, calcite, quartz, oil, etc.). So each component is bounded between zero and one. The components sum up to one. In other words, the volume fractions are confined to a simplex $\mathbb{S}^K = \{\mathbf{v}|0 \le v_j \le 1, \sum_k v_k = 1\}$. There are some additional physical constraints on the distribution of $\mathbf{v}$, for instance that the total amount of fluids should not exceed 40% of the total formation. The presence of more fluids would cause a collapse of the formation.

Each tool measurement gives a one-dimensional continuous value. The relation between composition and measurement outcome is well understood. Based on the physics of the tools, petrophysicists have expressed these relations in terms of deterministic functions $f_j(\mathbf{v})$ that provide the idealized noiseless measurement outcomes of tool j given the composition $\mathbf{v}$ [26]. In general, the functions f_j are nonlinear.

For most tools, the noise process is also reasonably well understood — and can be described by either a Gaussian (additive noise) or a log-Gaussian (multiplicative noise) distribution.

A straightforward approach to model a Bayesian network would be to discretize the variables and create conditional probability tables for priors and conditional distributions. However, due to the dimensionality of the volume fraction vector, any reasonable discretization would result in an infeasible large state space of this variable. We therefore decided to remain in the continuous domain.

The remainder of this section describes the prior and observation model, as well as the approximate inference method to obtain the posterior.

4.2 The prior and the observation model

The model has two ingredients: the prior of the volume fractions $P(\mathbf{v})$ and the observation model $P(m_j|\mathbf{v})$.

There is not much detailed domain knowledge available about the prior distribution. Therefore we decided to model the prior using conveniently parametrized family of distributions. In our case, $\mathbf{v} \in \mathbb{S}^K$, this lead to the Dirichlet distribution [17, 3]

$$Dir(\mathbf{v}|\alpha,\mu) \propto \prod_{j=1}^{K} v_j^{\alpha\mu_j-1} \delta\left(1-\sum_{i=1}^{K} v_i\right) . \tag{27}$$

The two parameters $\alpha \in \mathbb{R}_+$ (precision) and $\mu \in \mathbb{S}^K$ (vector of means) can be used to fine-tune the prior to our liking. The delta function — which ensures that the simplex constraint holds — is put here for clarity, but is in fact redundant if the model is constraint to $\mathbf{v} \in \mathbb{S}^K$. Additional information, e.g. the fact that the amount of fluids may not exceed 40% of the volume fraction can be incorporated by multiplying the prior by a likelihood term $\Phi(\mathbf{v})$ expressing this fact. The resulting prior is of the form

$$P(\mathbf{v}) \propto \Phi(\mathbf{v}) Dir(\mathbf{v}|\alpha,\mu) . \tag{28}$$

The other ingredient in the Bayesian network are the observation models. For most tools, the noise process is reasonably well understood and can be reasonably well described by either a Gaussian (additive noise) or a log-Gaussian (multiplicative noise) distribution. In the model, measurements are modeled as a deterministic tool function plus noise,

$$m_j = f_j(\mathbf{v}) + \xi_j , \tag{29}$$

in which the functions f_j are the deterministic tool functions provided by domain experts. For tools where the noise is multiplicative, a log transform is applied to the tool functions f_j and the measurement outcomes m_j. A detailed description of these functions is beyond the scope of this paper. The noises ξ_j are Gaussian and have a tool specific variance σ_j^2. These variances have been provided by domain experts. So, the observational probability models can be written as

$$P(m_i|\mathbf{v}) \propto \exp\left(-\frac{(m_j - f_j(\mathbf{v}))^2}{2\sigma_j^2}\right) . \tag{30}$$

4.3 Bayesian Inference

The next step is given a set of observations $\{m_i^o\}$, $i \in$ Obs, to compute the posterior distribution. If we were able to find an expression for the evidence term, i.e., for the marginal distribution of the observations $P(\mathbf{m}^o) = \int_{\mathbf{v}} \prod_{i\in \text{Obs}} P(m_i^o|\mathbf{v})P(\mathbf{v})\mathrm{d}\mathbf{v}$ then the posterior distribution (26) could be written in closed form and readily evaluated. Unfortunately $P(\mathbf{m}^o)$ is intractable and a closed-form expression does not exist. In order to obtain the desired compositional estimates we therefore have to resort to approximate inference methods. Pilot studies indicated that sampling methods gave the best performance.

The goal of any sampling procedure is to obtain a set of N samples $\{x_i\}$ that come from a given (but maybe intractable) distribution π. Using these samples we can approximate expectation values $\langle A\rangle$ of a function $A(x)$ according to

$$\langle A\rangle = \int_x A(x)\pi(x)\mathrm{d}x \approx \frac{1}{N}\sum_{i=1}^{N} A(x_i) . \tag{31}$$

For instance, if we take $A(x) = x$, the approximation of the mean $\langle x\rangle$ is the sample mean $\frac{1}{N}\sum_{i=1}^{N} x_i$.

An important class of sampling methods are the so-called Markov Chain Monte Carlo (MCMC) methods [17, 3]. In MCMC sampling a Markov chain is defined that has an equilibrium distribution π, in such a way that (31) gives a good approximation when applied to a sufficiently long chain $x_1, x_2, \ldots, x_N$. To make the chain independent of the initial state x_0, a burn-in period is often taken into account. This means that one ignores the first $M \ll N$ samples that come from intermediate distributions and begins storing the samples once the system has reached the equilibrium distribution π.

In our application we use the hybrid Monte Carlo (HMC) sampling algorithm [10, 17]. HMC is a powerful class of MCMC methods that are designed for problems with continuous state spaces, such as we consider in this section. HMC can in principle be applied to any noise model with a continuous probability density, so there is no restriction to Gaussian noise models. HMC uses Hamiltonian dynamics in combination with a Metropolis [19] acceptance procedure to find regions of higher probability. This leads to a more efficient sampler than a sampler that relies on random walk for phase space exploration. HMC also tends to mix more rapidly than the standard Metropolis Hastings algorithm. For details of the algorithm we refer to the literature [10, 17].

In our case, $\pi(\mathbf{v})$ is the posterior distribution $p(\mathbf{v}|m_i^o)$ in (26). The HMC sampler generates samples $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_N$ from this posterior distribution. Each of the N samples is a full K-dimensional vector of volume fractions constraint on $\mathbb{S}^K$. The number

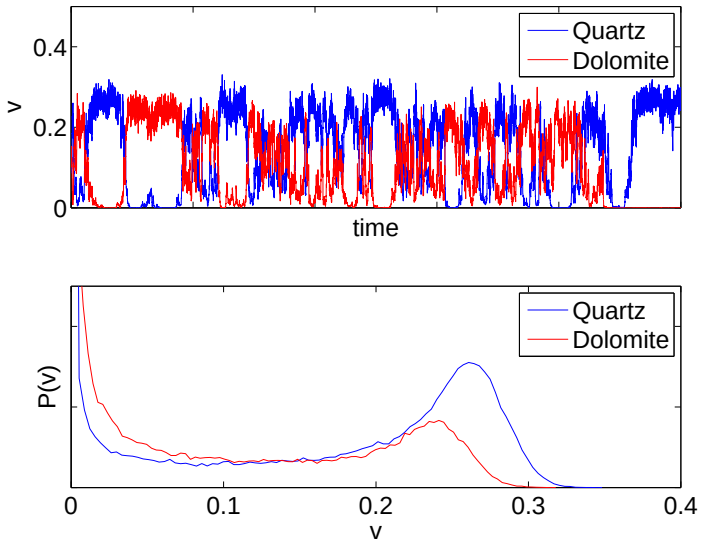

Fig. 7 Diagrams for quartz and dolomite. Top: time traces (10 000 time steps) of the volume fractions of quartz and dolomite. Bottom: Resulting marginal probability distributions of both fractions.

of samples is of the order of $N = 10^5$, which takes a few seconds on a standard PC. Figure 7 shows an example of a chain of 10 000 states generated by the sampler. For visual clarity, only two components of the vectors are plotted (quartz and dolomite). The plot illustrates the multivariate character of the method: for example, the traces shows that the volume fractions of the two minerals tend to be mutually exclusive: either 20% quartz, or 20% dolomite but generally not both. From the traces, all kind of statistics can be derived. As an example, the resulting one dimensional marginal distributions of the mineral volume fractions are plotted.

The performance of the method relies heavily on the quality of the sampler. Therefore we looked at the ability of the system to estimate the composition of a (synthetic) reservoir and the ability to reproduce the results. For this purpose, we set the composition to a certain value $\mathbf{v}^*$. We apply the observation model to generate measurements $\mathbf{m}^o$. Then we run HMC to obtain samples from the posterior $P(\mathbf{v}|\mathbf{m}^o)$. Consistency is assessed by comparing results of different runs to each other and by comparing them with the "ground truth" $\mathbf{v}^*$. Results of simulations confirm that the sampler generates reproducible results, consistent with the underlying compositional vector [5]. In these simulations, we took the observation model to generate measurement data (the generating model) equal to the observation model that is used to compute the posterior (the inference model). We also performed simulations where they are different, in particular in their assumed variance. We found that the sampler is robust to cases where the variance of the generating model is smaller than the variance of the inference model. In the cases where the variance of

the generating model is bigger, we found that the method is robust up to differences of a factor 10. After that we found that the sampler suffered severely from local minima, leading to irreproducible results.

4.4 Decision Support

Suppose that we have obtained a subset of measurement outcomes $\mathbf{m}^o$, yielding a distribution $P(\mathbf{v}|\mathbf{m}^o)$. One may subsequently ask the question which tool t should be deployed next in order to gain as much information as possible?

When asking this question, one is often interested in a specific subset of minerals and fluids. Here we assume this interest is actually in one specific component u. The question then reduces to selecting the most informative tool(s) t for a given mineral u.

We define the informativeness of a tool as the expected decrease of uncertainty in the distribution of v_u after obtaining a measurement with that tool. Usually, entropy is taken as a measure for uncertainty [17], so a measure of informativeness is the expected entropy of the distribution of v_u after measurement with tool t,

$$\begin{aligned}\langle \mathrm{H}_{u,t}|\mathbf{m}^o\rangle \equiv -\int P(m_t|\mathbf{m}^o)\int P\left(v_u|m_t,\mathbf{m}^o\right)\\ \times\log\left(P\left(v_u|m_t,\mathbf{m}^o\right)\right)\mathrm{d}v_u\mathrm{d}m_t\ .\end{aligned} \tag{32}$$

Note that the information of a tool depends on the earlier measurement results since the probabilities in (32) are conditioned on $\mathbf{m}^o$.

The most informative tool for mineral u is now identified as that tool t^* which yields in expectation the lowest entropy in the posterior distribution of v_u:

$$t^*_{u|\mathbf{m}^o} = \operatorname*{argmin}_t \langle \mathrm{H}_{u,t}|\mathbf{m}^o\rangle$$

In order to compute the expected conditional entropy using HMC sampling methods, we first rewrite the expected conditional entropy (32) in terms of quantities that are conditioned only on the measurement outcomes $\mathbf{m}^o$,

$$\begin{aligned}\langle \mathrm{H}_{u,t}|\mathbf{m}^o\rangle = -\int\int P(v_u,m_t|\mathbf{m}^o)\\ \times\log\left(P(v_u,m_t|\mathbf{m}^o)\right)\mathrm{d}v_u\mathrm{d}m_t\\ +\int P(m_t|\mathbf{m}^o)\int\log\left(P(m_t|\mathbf{m}^o)\right)\mathrm{d}m_t\ .\end{aligned} \tag{33}$$

Now the HMC run yields a set $V=\{v_1^j,v_2^j,\ldots,v_K^j\}$ of compositional samples (conditioned on $\mathbf{m}^o$). We augment these by a set $M=\{m_1^j=f_1(\mathbf{v}^j)+\xi_1^j,\ \ldots,\ m_Z^j=f_Z(\mathbf{v}^j)+\xi_Z^j\}$ of synthetic tool values generated from these samples (which are indexed by j) by applying equation (29). Subsequently, discretized joint proba-

bilities $P(v_u, m_t|\mathbf{m}^o)$ are obtained via a two-dimensional binning procedure over v_u and m_t for each of the potential tools t. The binned versions of $P(v_u, m_t|\mathbf{m}^o)$ (and $P(m_t|\mathbf{m}^o)$) can be directly used to approximate the expected conditional entropy using a discretized version of equation (33).

The outcome of our implementation of the decision support tool is a ranking of tools according to the expected entropies of their posterior distributions. In this way, the user can select a tool based on a trade-off between expected information and other factors, such as deployment costs and feasibility.

4.5 The Application

The application is implemented in C++ as a stand alone version with a graphical user interface running on a Windows PC. The application has been validated by petrophysical domain experts from Shell E&P. The further use by Shell of this application is beyond the scope of this chapter.

4.6 Summary

This chapter described a Bayesian network application for petrophysical decision support. The observation models are based on the physics of the measurement tools. The physical variables in this application are continuous-valued. A naive Bayesian network approach with discretized values would fail. We remained in the continuous domain and used the hybrid Monte Carlo algorithm for inference.

5 Bonaparte: a Bayesian Network for Disaster Victim Identification

Society is increasingly aware of the possibility of a mass disaster. Recent examples are the WTC attacks, the tsunami, and various airplane crashes. In such an event, the recovery and identification of the remains of the victims is of great importance, both for humanitarian as well as legal reasons. Disaster victim identification (DVI), i.e., the identification of victims of a mass disaster, is greatly facilitated by the advent of modern DNA technology. In forensic laboratories, DNA profiles can be recorded from small samples of body remains which may otherwise be unidentifiable. The identification task is the match of the unidentified victim with a reported missing person. This is often complicated by the fact that the match has to be made in an indirect way. This is the case when there is no reliable reference material of the missing person. In such a case, DNA profiles can be taken from relatives. Since

their profiles are statistically related to the profile of the missing person (first degree family members share about 50% of their DNA) an indirect match can be made.

In cases with one victim, identification is a reasonable straightforward task for forensic researchers. In the case of a few victims, the puzzle to match the victims and the missing persons is often still doable by hand, using a spread sheet, or with software tools available on the internet [9]. However, large scale DVI is infeasible in this way and an automated routine is almost indispensable for forensic institutes that need to be prepared for DVI.

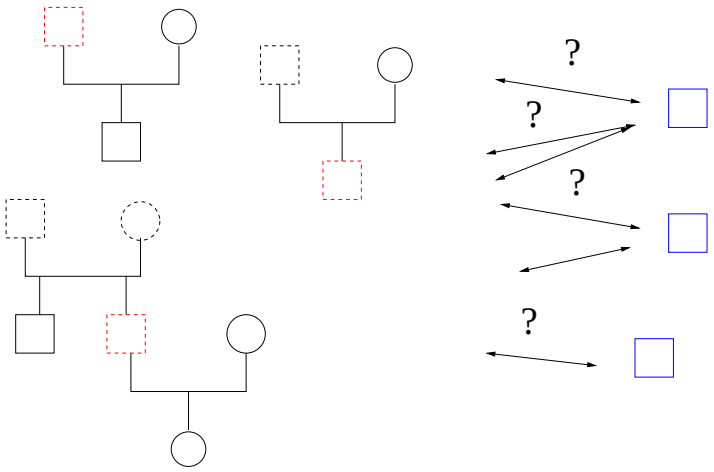

Fig. 8 The matching problem. Match the unidentified victims (blue, right) with reported missing persons (red, left) based on DNA profiles of victims and relatives of missing persons. DNA profiles are available from individuals represented by solid squares (males) and circles (females).

Bayesian networks are very well suited to model the statistical relations of genetic material of relatives in a pedigree [11]. They can directly be applied in kinship analysis with any type of pedigree of relatives of the missing persons. An additional advantage of a Bayesian network approach is that it makes the analysis tool more transparent and flexible, allowing to incorporate other factors that play a role — such as measurement error probability, missing data, statistics of more advanced genetic markers etc.

Currently, we develop software for DVI, called Bonaparte. This development is in collaboration with NFI (Netherlands Forensic Institute). The computational engine of Bonaparte uses automatically generated Bayesian networks and Bayesian inference methods, enabling to correctly do kinship analysis on the basis of DNA profiles combined with pedigree information. It is designed to handle large scale events, with hundreds of victims and missing persons. In addition, it has graphical user interface, including a pedigree editor, for forensic analysts. Data-interfaces to other laboratory systems (e.g., for the DNA-data input) will also be implemented.

In the remainder of this section we will describe the Bayesian model approach that has been taken in the development of the application. We formulate the computational task, which is the computation of the likelihood ratio of two hypotheses. The main ingredient is a probabilistic model P of DNA profiles. Before discussing the model, we will first provide a brief introduction to DNA profiles. In the last part of the section we describe how P is modeled as a Bayesian network, and how the likelihood ratio is computed.

5.1 Likelihood Ratio of Two Hypotheses

Assume we have a pedigree with an individual MP who is missing (the Missing Person). In this pedigree, there are some family members that have provided DNA material, yielding the profiles. Furthermore there is an Unidentified Individual UI, whose DNA is also profiled. The question is, is $UI = MP$? To proceed, we assume that we have a probabilistic model P for DNA evidence of family members in a pedigree. To compute the probability of this event, we need hypotheses to compare. The common choice is to formulate two hypotheses. The first is the hypothesis H_1 that indeed $UI = MP$. The alternative hypothesis H_0 is that UI is an unrelated person U. In both hypotheses we have two pedigrees: the first pedigree has MP and family members FAM as members. The second one has only U as member. To compare the hypotheses, we compute the likelihoods of the evidence from the DNA profiles under the two hypotheses,

- Under H_p, we assume that $MP = UI$. In this case, MP is observed and U is unobserved. The evidence is $E = \{DNA_{MP} + DNA_{FAM}\}$.
- Under H_d, we assume that $U = UI$. In this case, U is observed and MP is observed. The evidence is $E = \{DNA_U + DNA_{FAM}\}$.

Under the model P, the likelihood ratio of the two hypotheses is

$$LR = \frac{P(E|H_p)}{P(E|H_d)} \,. \tag{34}$$

If in addition a prior odds $P(H_p)/P(H_d)$ is given, the posterior odds $P(H_p|E)/P(H_d|E)$ follows directly from multiplication of the prior odds and likelihood ratio,

$$\frac{P(H_p|E)}{P(H_d|E)} = \frac{P(E|H_p)P(H_p)}{P(E|H_d)P(H_d)} \,. \tag{35}$$

5.2 DNA Profiles

In this subsection we provide a brief introduction on DNA profiles for kinship analysis. A comprehensive treatise can be found in e.g. [6]. In humans, DNA found in the

nucleus of the cell is packed on chromosomes. A normal human cell has 46 chromosomes, which can be organized in 23 pairs. From each pair of chromosomes, one copy is inherited from father and the other copy is inherited from mother. In 22 pairs, chromosomes are homologous, i.e., they have practically the same length and contain in general the same genes (functional functional elements of DNA). These are called the autosomal chromosomes. The remaining chromosome is the sex-chromosome. Males have an X and a Y chromosome. Females have two X chromosomes.

More than 99% of the DNA of any two humans of the general population is identical. Most DNA is therefore not useful for identification. However, there are well specified locations on chromosomes where there is variation in DNA among individuals. Such a variation is called a genetic marker. In genetics, the specified locations are called loci. A single location is a locus.

In forensic research, the short tandem repeat (STR) markers are currently most used. The reason is that they can be reliable determined from small amounts of body tissue. Another advantage is that they have a low mutation rate, which is important for kinship analysis. STR markers is a class of variations that occur when a pattern of two or more nucleotides is repeated. For example,

$$(CATG)_3 = CATGCATGCATG\,. \tag{36}$$

The number of repeats x (which is 3 in the example) is the variation among the population. Sometimes, there is a fractional repeat, e.g. $CATGCATGCATGCA$, this would be encoded with repeat number $x = 3.2$, since there are three repeats and two additional nucleotides. The possible values of x and their frequencies are well documented for the loci used in forensic research. These ranges and frequencies vary between loci. To some extend they vary among subpopulations of humans. The STR loci are standardized. The NFI uses CODIS (Combined DNA Index System) standard with 13 specific core STR loci, each on different autosomal chromosomes.

The collection of markers yields the DNA profile. Since chromosomes exist in pairs, a profile will consist of pairs of markers. For example in the CODIS standard, a full DNA profile will consist of 13 pairs, (the following notation is not common standard)

$$\overline{\mathbf{x}} = ({}^{1}x^{1}, {}^{1}x^{2}), ({}^{2}x^{1}, {}^{2}x^{2}), \ldots, ({}^{13}x^{1}, {}^{13}x^{2})\,, \tag{37}$$

in which each ${}^{\mu}x^{s}$ is a number of repeats at a well defined locus μ. However, since chromosomes exists in pairs, there will be two alleles ${}^{\mu}x^{1}$ and ${}^{\mu}x^{2}$ for each location, one paternal — on the chromosome inherited from father — and one maternal. Unfortunately, current DNA analysis methods cannot identify the phase of the alleles, i.e., whether an allele is paternal or maternal. This means that $({}^{\mu}x^{1}, {}^{\mu}x^{2})$ cannot be distinguished from $({}^{\mu}x^{2}, {}^{\mu}x^{1})$. In order to make the notation unique, we order the observed alleles of a locus such that ${}^{\mu}x^{1} \leq {}^{\mu}x^{2}$.

Chromosomes are inherited from parents. Each parent passes one copy of each pair of chromosomes to the child. For autosomal chromosomes there is no (known) preference which one is transmitted to the child. There is also no (known) correlation between the transmission of chromosomes from different pairs. Since chromo-

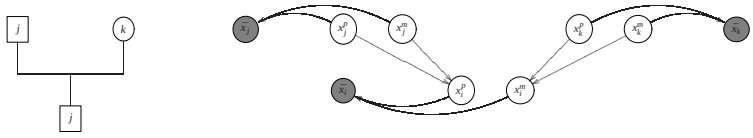

Fig. 9 A basic pedigree with father, mother, and child. Squares represent males, circles represent females. Right: corresponding Bayesian network. Grey nodes are observables. x_j^p and x_j^m represents paternal and maternal allele of individual j. See text.

somes are inherited from parents, alleles are inherited from parents as well. However, there is a small probability that an allele is changed or mutated. This mutation probability is about 0.1%.

Finally in the DNA analysis, sometimes failures occur in the DNA analysis method and an allele at a certain locus drops out. In such a case the observation is $({}^{\mu}x^1, ?)$, in which "?" is a wild card.

5.3 A Bayesian Network for Kinship Analysis

In this subsection we will describe the building blocks of a Bayesian network to model probabilities of DNA profiles of individuals in a pedigree. First we observe that inheritance and observation of alleles at different loci are independent. So for each locus we can make an independent model P_μ. In the model description below, we will consider a model for a single locus, and we will suppress the μ dependency for notational convenience.

5.3.1 Allele Probabilities

We will consider pedigrees with individuals i. In a pedigree, each individual i has two parents, a father $f(i)$ and a mother $m(i)$. An exception is when a individual is a founder. In that case it has no parents in the pedigree.

Statistical relations between DNA profiles and alleles of family members can be constructed from the pedigree, combined with models for allele transmission . On the given locus, each individual i has a paternal allele x_i^f and an maternal allele x_i^m. f and m stands for 'father' and 'mother'. The pair of alleles is denoted as $x_i = (x_i^f, x_i^m)$. Sometimes we use superscript s which can have values $\{f, m\}$. So each allele in the pedigree is indexed by (i, s), where i runs over individuals and s over phases (f, m). The alleles can assume N values, where N as well as the allele values depend on the locus.

An allele from a founder is called 'founder allele'. So a founder in the pedigree has two founder alleles. The simplest model for founder alleles is to assume that they are independent, and each follow a distribution $P(a)$ of population frequencies.

This distribution is assumed to be given. In general $P(a)$ will depend on the locus. More advanced models have been proposed in which founder alleles are correlated. For instance, one could assume that founders in a pedigree come from a single but unknown subpopulation [1]. This model assumption yield corrections to the outcomes in models without correlations between founders. A drawback is that these models may lead to a severe increase in required memory and computation time. In this chapter we will restrict ourself to models with independent founder alleles.

If an individual i has its parents in the pedigree the allele distribution of an individual given the alleles of its parents are as follows,

$$P(x_i|x_{f(i)},x_{m(i)}) = P(x_i^f|x_{f(i)})P(x_i^m|x_{m(i)}) \,, \tag{38}$$

where

$$P(x_i^f|x_{f(i)}) = \frac{1}{2}\sum_{s=f,m} P(x_i^f|x_{f(i)}^s) \,, \tag{39}$$

$$P(x_i^m|x_{m(i)}) = \frac{1}{2}\sum_{s=f,m} P(x_i^m|x_{m(i)}^s) \,. \tag{40}$$

To explain (39) in words: individual i obtains its parental allele x_i^f from its father $f(i)$. In this process, there is a 50% chance that the *parental* allele $x_{f(i)}^f$ of father $f(i)$ is transmitted and a 50% chance that the *maternal* allele $x_{f(i)}^m$ of father $f(i)$ is transmitted. A similar explanation applies to (40).

The probabilities $P(x_i^f|x_{f(i)}^s)$ and $P(x_i^m|x_{m(i)}^s)$ are given by a mutation model $P(a|b)$, which encodes the probability that allele of the child is a while the allele on the parental chromosome that is transmitted is b. The precise mutation mechanisms for the different STR markers are not known. There is evidence that mutations from father to child are in general about 10 times as probable as mutations from mother to child. Gender of each individual is assumed to be known, but for notational convenience we suppress dependency of parent gender. In general, mutation tends to decrease with the difference in repeat numbers $|a-b|$. Mutation is also locus dependent [4].

Several mutation models have been proposed, see e.g. [8]. As we will see later, however, the inclusion of a detailed mutation model may lead to a severe increase in required memory and computation time. Since mutations are very rare, one could ask if there is any practical relevance in a detailed mutation model. The simplest mutation model is of course to assume the absence of mutations, $P(a|b) = \delta_{a,b}$. Such model enhances efficient inference. However, any mutation in any single locus would lead to a 100% rejection of the match, even if there is a 100% match in the remaining markers. Mutation models are important to get some model tolerance against such case. The simplest non-trivial mutation model is a uniform mutation model with mutation rate μ (not to be confused with the locus index μ),

$$P(a|a) = 1-\mu\ , \tag{41}$$
$$P(a|b) = \mu/(N-1) \quad \text{if } a \neq b\ . \tag{42}$$

Mutation rate may depend on locus and gender.

An advantage of this model is that the required memory and computation time increases only slightly compared to the mutation free model. Note that the population frequency is in general not invariant under this model: the mutation makes the frequency more flat. One could argue that this is a realistic property that introduces diversity in the population. In practical applications in the model, however, the same population frequency is assumed to apply to founders in different generations in a pedigree. This implies that if more unobserved references are included in the pedigree to model ancestors of an individual, the likelihood ratio will (slightly) change. In other words, formally equivalent pedigrees will give (slightly) different likelihood ratios.

5.3.2 Observations

Observations are denoted as $\bar{x_i}$, or $\bar{x}$ if we do not refer to an individual. The parental origin of an allele can not be observed, so alleles $x^f = a, x^m = b$ yields the same observation as $x^f = b, x^m = a$. We adopt the convention to write the smallest allele first in the observation: $\bar{x} = (a,b) \Leftrightarrow a \leq b$. In the case of an allele loss, we write $\bar{x} = (x,?)$ where ? stands for a wild card. We assume that the event of an allele loss can be observed (e.g. via the peak hight [6]). This event is modeled by L. With $L = 1$ there is allele loss, and there will be a wild card ?. A full observation is coded as $L = 0$. The case of loss of two alleles is not modeled, since in that case we simply have no observation.

The observation model is now straightforwardly written down. Without allele loss ($L = 0$), alleles y results in an observation $\bar{y}$. This is modeled by the deterministic table

$$P(\bar{x}|y, L=0) = \begin{cases} 1 & \text{if } \bar{x} = \bar{y}\ , \\ 0 & \text{otherwise.} \end{cases} \tag{43}$$

Note that for a given y there is only one $\bar{x}$ with $\bar{x} = \bar{y}$.

With allele loss ($L = 1$), we have

$$\begin{cases} P(\bar{x} = (a,F)|(a,b), L=1) = \frac{1}{2} \\ P(\bar{x} = (b,F)|(a,b), L=1) = \frac{1}{2} \end{cases} \quad \text{if} \quad a \neq b\ , \tag{44}$$

and

$$P(\bar{x} = (a,F)|(a,a), L=1) = 1\ . \tag{45}$$

I.e., if one allele is lost, the alleles (a,b) leads to an observation a (then b is lost), or to an observation b (then a is lost). Both events have 50% probability. If both alleles are the same, so the pair is (a,a), then of course a is observed with 100% probability.

5.4 Inference

By multiplying all allele priors, transmission probabilities and observation models, a Bayesian network of alleles x and DNA profiles of individuals $\bar{x}$ in a given pedigree is obtained. Assume that the pedigree consists of a set of individuals $\mathscr{I} = 1,\ldots,K$ with a subset of founders $\mathscr{F}$, and assume that allele losses L_j are given, then this probability reads

$$P(\{\bar{x},x\}_{\mathscr{I}}) = \prod_j P(\bar{x}_j|x_j,L_j) \prod_{i\in\mathscr{I}\setminus\mathscr{F}} P(x_i|x_{f(i)},x_{m(i)}) \prod_{i\in\mathscr{F}} P(x_i)\ . \tag{46}$$

Under this model the likelihood of a given set DNA profiles can now be computed. If we have observations $\bar{x}_j$ from a subset of individuals $j \in \mathscr{O}$, the likelihood of the observations in this pedigree is the marginal distribution $P(\{\bar{x}\}_{\mathscr{O}})$, which is the marginal probability

$$P(\{\bar{x}\}_{\mathscr{O}}) = \sum_{x_1}\ldots\sum_{x_K} \prod_{j\in\mathscr{O}} P(\bar{x}_j|x_j,L_j) \prod_{i\in\mathscr{I}\setminus\mathscr{F}} P(x_i|x_{f(i)},x_{m(i)}) \prod_{i\in\mathscr{F}} P(x_i)\ . \tag{47}$$

This computation involves the sum over all states of allele pairs x_i of all individuals.

In general, the allele-state space can be prohibitively large. This would make even the junction tree algorithm infeasible if it would straightforwardly be applied. Fortunately, a significant reduction in memory requirement can be achieved by "value abstraction": if the observed alleles in the pedigree are all in a subset A of M different allele values, we can abstract from all unobserved allele values and consider them as a single state z. If an allele is z, it means that it has a value that is not in the set of observed values A. We now have a system in which states can assume only $M+1$ values which is generally a lot smaller than N, the number of a priori possible allele values. This procedure is called value abstraction [12]. The procedure is applicable if for any $a \in A$, $L \in \{0,1\}$, and $b_1,b_2,b_3,b_4 \notin A$, the following equalities hold

$$P(a|b_1) = P(a|b_2) \tag{48}$$
$$P(\bar{x}|a,b_1,L) = P(\bar{x}|a,b_2,L) \tag{49}$$
$$P(\bar{x}|b_1,a,L) = P(\bar{x}|b_2,a,L) \tag{50}$$
$$P(\bar{x}|b_1,b_2,L) = P(\bar{x}|b_3,b_4,L) \tag{51}$$

If these equalities hold, then we can replace $P(a|b)$ by $P(a|z)$ and $P(\bar{x}|a,b)$ by $P(\bar{x}|a,z)$ etc. in the abstracted state representation. The conditional probability of z then follows from

$$P(z|x) = 1 - \sum_{a\in A} P(a|x) \tag{52}$$

for all x in $A \cup z$. One can also easily check that the observation probabilities satisfy the condition. The uniform mutation model satisfies condition (48) since $P(a|b) =$

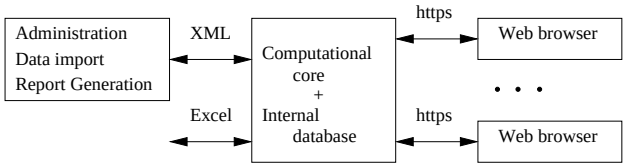

Fig. 10 Bonaparte's basic architecture

$\mu/(N-1)$ for any $a \in A$ and any $b \notin A$. Note that condition (48) does not necessarily holds for a general mutation model, so value abstraction could then not be applied.

Using value abstraction as a preprocessing step, a junction tree-based algorithm can straightforwardly applied to compute the desired likelihood. In this way, likelihoods and likelihood ratios are computed for all loci, and reported to the user.

5.5 The application

Bonaparte has been designed to facilitate large scale matching. The application has a multi-user client-server based architecture, see fig. 10. Its computational core and the internal database runs on a server. All match results are stored in internal database. Rewind to any point in back in time is possible. Via an XML and secure https interfaces, the server connects to other systems. Users can login via a web-browser so that no additional software is needed on the clients. The current version Bonaparte is now under user-validation. A live demo version will be made available on www.dnadvi.nl.

5.6 Summary

Bonaparte is an application of Bayesian networks for victim identification by kinship analysis based on DNA profiles. The Bayesian networks are used to model statistical relations between DNA profiles of different individuals in a pedigree. By Bayesian inference, likelihood ratios and posterior odds of hypotheses are computed, which are the quantities of interest for the forensic researcher. The probabilistic relations between variables are based on first principles of genetics. A feature of this application is the automatic, on-the-fly derivation of models from data, i.e., the pedigree structure of a family of a missing person.

6 Discussion

Human decision makers are often confronted with highly complex domains. They have to deal with various sources of information and various sources of uncertainty. The quality of the decision is strongly influenced by the decision makers experience to correctly interpret the data at hand. Computerized decision support can help to improve the effectiveness of the decision maker by enhancing awareness and alerting the user to uncommon situations that may have high impact. Rationalizing the decision process may alleviate some of the decision pressure.

Bayesian networks are widely accepted as a principled methodology for modeling complex domains with uncertainty, in which different sources of information are to be combined, as needed in intelligent decision support systems. However, many of the examples of Bayesian networks as described in literature — models with a few dozen of variables, each with a few states, and fixed relations — may suggest a limitation in the expressive power of the methodology [18].

In this chapter we described three Bayesian networks for real-world applications. These models are based on the same principled methodology as standard Bayesian networks, but go beyond the above mentioned limitations. The Promedas model has several orders of magnitudes more variables. The petrophysical model has continuous-valued variables. The Bonaparte model as well as the Promedas model have non-static relations.

Fundamental differences of these models with most standard Bayesian networks are (1) the model development approach and (2) the operational power and flexibility of the applications. Standard Bayesian networks are often developed using off-the-shelf GUI-based software. An advantage of this approach is that small or medium sized Bayesian networks can be developed quickly, without the need of expertise on Bayesian networks modeling or inference algorithms. The models described in this chapter, on the other hand, have been developed from scratch, based on first principles and with customized implementations of inference algorithms (junction tree based, or approximate such as the HMC method). This development approach requires more expertise, but it has more flexibility as it is not constrained by the development software and can better handle the various problems posed by the applications, such as the large number of variables, the continuous-valued variables, and on-the-fly model derivation from data, etc.

We have discussed in detail three applications of Bayesian networks. With these applications, we aimed to illustrate the modeling power of the Bayesian networks that goes beyond the standard textbook applications. The applications domains of the models (medicine, petrophysics and forensics) demonstrate that Bayesian networks can be applied in a wide variety of domains with different types of domain requirements.

Finally, we would like to stress that the Bayesian network technology is only one side of the model. The other side is the domain knowledge, which is maybe even more important for the model. Therefore Bayesian network modeling always requires a close collaboration with domain experts. And even then, the model is of course only one of many ingredients of an application, such as user-interface, data-

management, user-acceptance etc. which are all essential to make the application a success.

Acknowledgments

The presented work was partly carried out with support from the Intelligent Collaborative Information Systems (ICIS) project, supported by the Dutch Ministry of Economic Affairs, grant BSIK03024. The research for the Promedas project has been supported by the Dutch Technology Foundation STW, applied science division of NWO and the Technology Program of the Ministry of Economic Affairs. We thank Kees Albers and Martijn Leisink (SNN), Jan Neijt (UMC Utrecht), Mirano Spalburg (Shell E & P), Klaas Slooten and Carla Bruijning (NFI) for their collaboration. Finally, we thank the anonymous reviewers for their useful comments.

References

1. Balding, D., Nichols, R.: DNA profile match probability calculation: how to allow for population stratification, relatedness, database selection and single bands. Forensic Science International **64**(2-3), 125–140 (1994)
2. Beinlich, I., Suermondt, H., Chavez, R., Cooper, G., et al.: The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks. In: Proceedings of the Second European Conference on Artificial Intelligence in Medicine, vol. 256. Berlin: Springer-Verlag (1989)
3. Bishop, C.: Pattern recognition and machine learning. Springer (2006)
4. Brinkmann, B., Klintschar, M., Neuhuber, F., Hühne, J., Rolf, B.: Mutation rate in human microsatellites: influence of the structure and length of the tandem repeat. The American Journal of Human Genetics **62**(6), 1408–1415 (1998)
5. Burgers, W., Wiegerinck, W., Kappen, H., Spalburg, M.: A Bayesian petrophysical decision support system for estimation of reservoir compositions. Submitted
6. Butler, J.: Forensic DNA typing: biology, technology, and genetics of STR markers. Academic Press (2005)
7. Castillo, E., Gutierrez, J.M., Hadi, A.S.: Expert Systems and Probabilistic Network Models. Springer (1997)
8. Dawid, A., Mortera, J., Pascali, V.: Non-fatherhood or mutation? A probabilistic approach to parental exclusion in paternity testing. Forensic science international **124**(1), 55–61 (2001)
9. Drábek, J.: Validation of software for calculating the likelihood ratio for parentage and kinship. Forensic Science International: Genetics **3**(2), 112–118 (2009)
10. Duane, S., Kennedy, A., Pendleton, B., Roweth, D.: Hybrid Monte Carlo Algorithm. Phys. Lett. B **195**, 216 (1987)
11. Fishelson, M., Geiger, D.: Exact genetic linkage computations for general pedigrees. Bioinformatics **18**(Suppl 1), S189–S198 (2002)
12. Friedman, N., Geiger, D., Lotner, N.: Likelihood computations using value abstraction. In: Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence, pp. 192–200. Morgan Kaufmann Publishers (2000)
13. Heckerman, D.: Probabilistic interpretations for mycin's certainty factors. In: L. Kanal, J. Lemmer (eds.) Uncertainty in artificial intelligence, pp. 167–96. North Holland (1986)
14. Jensen, F.: An Introduction to Bayesian networks. UCL Press (1996)
15. Jordan, M.: Learning in graphical models. Kluwer Academic Publishers (1998)

16. Lauritzen, S., Spiegelhalter, D.: Local computations with probabilities on graphical structures and their application to expert systems. Journal of the Royal Statistical Society. Series B (Methodological) pp. 157–224 (1988)
17. MacKay, D.: Information theory, inference and learning algorithms. Cambridge University Press (2003)
18. Mahoney, S., Laskey, K.: Network engineering for complex belief networks. In: Proc. 12th Conf. on Uncertainty in Artificial Intelligence, pp. 389–396. Morgan Kaufmann (1996)
19. Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., Teller, E.: Equation of state calculations by fast computing machines. The journal of chemical physics **21**(6), 1087 (1953)
20. Pearl, J.: Probabilistic Reasoning in Intelligent systems: Networks of Plausible Inference. Morgan Kaufmann Publishers, Inc. (1988)
21. Pradhan, M., Provan, G., Middleton, B., Henrion, M.: Knowledge engineering for large belief networks. In: Proc. Tenth Conf. on Uncertainty in Artificial Intelligence, pp. 484–490 (1994)
22. Russell, S., Norvig, P., Canny, J., Malik, J., Edwards, D.: Artificial intelligence: a modern approach. Prentice Hall (2003)
23. Schlumberger: Log Interpretation Principles/Applications. Schlumberger Limited (1991)
24. Shortliffe, E., Buchanan, B.: A model of inexact reasoning in medicine. Mathematical Biosciences **23**(3-4), 351–379 (1975)
25. Shwe, M., Middleton, B., Heckerman, D., Henrion, M., E.J., H., Lehman, H., Cooper, G.: Probabilistic Diagnosis Using a Reformulation of the Internist-1/ QMR Knowledge Base. Methods of Information in Medicine **30**, 241–55 (1991)
26. Spalburg, M.: Bayesian uncertainty reduction for log evaluation. SPE International (2004). SPE88685
27. Takinawa, M., D'Ambrosio, B.: Multiplicative factorization of noisy-MAX. In: Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence UAI99, pp. 622–30 (1999)