BUILDING BAYESIAN NETWORKS: ELICITATION, EVALUATION, AND LEARNING

by

Haiqin Wang

M.S., University of Pittsburgh

M.S., Institute of Automation, Chinese Academy of Sciences

B.S., University of Science and Technology of China

Submitted to the Graduate Faculty of
Arts and Sciences in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy

University of Pittsburgh

2004

UNIVERSITY OF PITTSBURGH

FACULTY OF ARTS AND SCIENCES

This dissertation was presented

by

Haiqin Wang

It was defended on

December, 2004

and approved by

Marek J. Druzdzel, Associate Professor, Intelligent Systems Program

Gregory F. Cooper, Associate Professor, Intelligent Systems Program

Michael Lewis, Associate Professor, Information Sciences

Irina Rish, Research Scientist, IBM Watson Research Center

Dissertation Director: Marek J. Druzdzel, Associate Professor, Intelligent Systems Program

BUILDING BAYESIAN NETWORKS: ELICITATION, EVALUATION, AND LEARNING

Haiqin Wang, PhD

University of Pittsburgh, 2004

As a compact graphical framework for representation of multivariate probability distributions, Bayesian networks are widely used for efficient reasoning under uncertainty in a variety of applications, from medical diagnosis to computer troubleshooting and airplane fault isolation. However, construction of Bayesian networks is often considered the main difficulty when applying this framework to real-world problems. In real world domains, Bayesian networks are often built using a knowledge engineering approach. Unfortunately, eliciting knowledge from domain experts is a very time-consuming process, and could result in poor-quality graphical models when not performed carefully. Over the last decade, the research focus is shifting more towards learning Bayesian networks from data, especially with increasing volumes of data available in various applications, such as biomedical, internet, and e-business, among others.

Aiming at solving the bottle-neck problem of building Bayesian network models, this research work focuses on elicitation, evaluation and learning Bayesian networks. Specifically, the contribution of this dissertation involves the research in the following five areas: a) graphical user interface tools for efficient elicitation and navigation of probability distributions, b) systematic and objective evaluation of elicitation schemes for probabilistic models, c)valid evaluation of performance robustness, i.e., sensitivity, of Bayesian networks, d) the sensitivity inequivalent characteristic of Markov equivalent networks, and the appropriateness of using sensitivity for model selection in learning Bayesian networks, e) selective refinement for learning probability parameters of Bayesian networks from limited data with availability of expert knowledge. In addition, an efficient algorithm for fast sensitivity analysis is developed based on a relevance reasoning technique. The implemented algorithm runs very fast and makes d) and e) more affordable for real domain practice.

To my parents, sister, brothers, and David

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my advisor, Professor Marek Druzdzel, for his generous support over the years, and for giving me so much freedom to explore the various areas in the probabilistic AI field. His continuous encouragement throughout and after my school years made me more and more confident on the UAI research. His enthusiasm and ambition on the research with a very practical approach will always inspire me toward the next milestone in my professional career.

I owe my hearted thanks to Dr. Irina Rish for hiring me into the internship program of IBM Watson Research Center in 2001, where I started to attack the problem of learning Bayesian networks using sensitivity as model selection criterion. I was deeply impressed by her research methodology. Her wonderful mentoring skills and friendship are always precious gifts to me.

I would like to thank my other committee members who have also been very supportive. Professor Gregory Cooper's thought-provoking questions and insightful comments have directly guided my research work on selective learning parameters for Bayesian networks. Professor Michael Lewis has enthusiastically offered review and recommendation for my paper on evaluation of probability elicitation to one of the most prestigious IEEE journal and finally led to its publication.

Also, I would like to thank my Boeing Colleagues, Oscar Kipersztok, Cathy Kitto, Grace Bahandari, Alice Chen, Guijun Wang, Shan Luh etc for their support and encouragement to fulfil the dissertation. Special thanks go to Oscar who has been working with me for many years including a very productive internship at the beginning, and showed me the secret weapons for success in industrial transition of scientific research.

Many thanks to all DSL members whom I have had privileges and pleasures to work with, especially Denver Dash, Tsaiching Lu, Adam Zagorechi for thoughtful discussions and interesting debates, Tomasz Sowinski for his neat code reusable in programming on sensitivity analysis algorithms, Changhe Yuan for his generous support on obtaining data for testing algorithms, and Yan Lin, Agnieszka Onisko for their great friendships.

I am very appreciative to Keena Walker for her always instant response and kindly help on the administrative matters during my dissertation study.

Finally, I am in debt to my dear parents, sister and brothers for endless love and support in exploring the colorful outside world. I am thankful to my friends for much-needed diversions. And of course, I give my appreciation to my beloved son, David, who erases any tiredness and depression with his beautiful smile, innocent mind, and offer of his favorite snacks.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1.0 INTRODUCTION

Bayesian networks provide a graphical framework for compact representation of multivariate probability distributions and efficient reasoning under uncertainty. Graphical probabilistic models are widely used in various applications, including medical diagnosis, computer troubleshooting, traffic control, airplane failure isolation, speech recognition, and error-correcting codes, to name a few. However, construction of Bayesian networks is often considered the main difficulty when applying this framework to real-world problems. Aiming at solving the bottle-neck problem of building Bayesian models, this research work focuses on three modelling areas: elicitation, evaluation and learning. The following sections discuss the challenges in these areas and briefly introduce the solutions explored by the thesis work.

1.1 BAYESIAN NETWORKS

Bayesian networks (also called *belief networks*) provide an increasingly popular graphical framework for Bayesian reasoning, a probabilistic approach to inference that basically combines prior knowledge with observed data using the *Bayes' rule*:

$$P(H|D) = \frac{P(D|H)P(H)}{P(D)}, \tag{1.1}$$

where $P(H)$ is the *prior* probability of hypothesis H, $P(D)$ is the prior probability of observing data D, $P(D|H)$ (called *likelihood*) is the probability of observing D if hypothesis H holds, and $P(H|D)$ is the *posterior* probability of H after observing data D.

Formally, a Bayesian network B is a pair (G, Θ), where G is a directed acyclic graph in which nodes represent random variables of interest (e.g., the temperature of a device, the gender of a patient, a feature of an object, occurrence of an event) and edges denote probabilistic dependencies. Since the directed edges are often interpreted as direct causal influences between the variables, Bayesian networks are also called *causal networks*. In addition to G, the parameters Θ define the probability distributions that specify the strength of the conditional dependencies between the variables in B.

Let $\mathbf{X} = \{X_1, X_2, \ldots, X_n\}$[1] be a set of random variables modeled in B, and let $\Theta = \{\theta_{X_i|\boldsymbol{Pa_i}}\}$ be the set of parameters that represent conditional probability distributions for each node X_i given its parents $\boldsymbol{Pa_i}$ (direct predecessors of X_i in the graph) in G, i.e., $\theta_{X_i|\boldsymbol{Pa_i}} = p(X_i|\boldsymbol{Pa_i})$. The distributions $p(X_i|\boldsymbol{Pa_i})$, associated with each node X_i, are called *local probability distributions* [29]. As a compact representation framework, Bayesian network factors a joint probability distribution over $\boldsymbol{X}$ into a product of local distributions:

$$p(X_1, \ldots, X_n) = \prod_{i=1}^{n} p(X_i|\boldsymbol{Pa_i}) \ .$$

Typically, Bayesian networks are defined for unrestricted multinomial variables, i.e., discrete variables with finite number of states. Thus, local probability distributions are represented by $(m+1)$-dimensional *conditional probability tables (CPTs)*, where m is the number of parents, and each entry $p(X_i|\boldsymbol{Pa_i})$ corresponds to a particular value assignment to X_i and its parents $\boldsymbol{Pa_i}$. It is easy to see that X_i's probability matrix for each node is exponential in the number of its parents.

Figure 1.1 shows an example of Bayesian network. It is a small fragment of *HEPAR II* [53] network built for medical diagnosis for liver diseases. The causal relationship between liver disorder to possible causes (e.g., gallstone, alcoholism) and to symptoms (e.g., fatigue, jaundice) can be read directly from the links in the graph. In this network, node *Disorder* has three binary parents: *Alcoholism*, *Hepatotoxic medications*, and *Gallstones*, each of which is a causal factor contributing to each of six possible liver disorders. There are totally 48 probability parameters to define node *Disorder* conditioned on its parent configurations. For a root node (i.e., a node having no parents), the prior probability distribution is defined over the node's outcomes. *HEPAR II* includes 72 variables and requires over 2000 numerical parameters for its full quantification.

[1] In the dissertation, variables will be denoted by capital letters, e.g., X; value of variables will be denoted by lower case letters, e.g., x; set of variables will be denoted in bold font, e.g., $\boldsymbol{Pa_i}$.

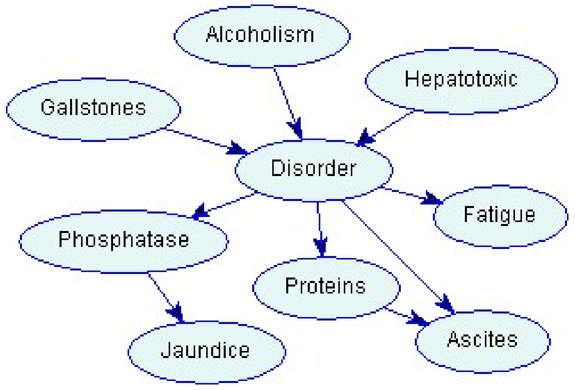


Figure 1.1: An example of a Bayesian network.

1.2 CHALLENGES IN BUILDING BAYESIAN NETWORKS: ELICITATION, EVALUATION AND LEARNING

Building a Bayesian network involves constructing graph structure G and populating each node X_i in the graph with corresponding conditional probability distributions. Whereas eliciting the graph structure is challenging, the most daunting task is the quantification that often requires specification of thousands of conditional probabilities. One way to acquire the probability distributions is to elicit the probability parameters by interviewing domain experts. Unfortunately, human judgement on probability is prone to systematic errors (biases) that can be invoked by a variety of factors [39]. Elicitation of probabilities, if not performed carefully, can result in poor quality estimates. Behavioral decision theorists have proposed several elicitation approaches that minimize the risk of bias. However, these methods tend to be cumbersome and often infeasible for models that include more than a few variables because of the large number of elicitations required[39]. Decision analytic practice is usually based on methods that require less effort and still protect subjective assessments from common biases. Among them, graphical elicitation is the most intuitive and the easiest to use. However, the current graphical tools for elicitation leaves a lot of room for improvement given what we know about the principles of human computer interaction.

With various methods available for probability elicitation, a closely related question is, then, how effective the elicitation methods are. Evaluation of the elicitation methods can provide guidance to select the appropriate schemes for probability elicitation when building Bayesian networks using knowledge engineering approach. Some studies compared a handful of elicitation methods [37, 47, 59, 73]. But the comparisons performed were more like case studies rather than systematic evaluations. The existing techniques for probability elicitation focus on balancing quality of elicitation with the time required to elicit the enormous number of parameters associated with many practical models where exactly this balance should fall is not obvious. Finding the right balance requires a systematic evaluation and comparison of different elicitation schemes with respect to the quality and efficiency criteria.

Since the probabilities elicited from domain experts are often systematically biased, it is interesting to investigate the reliability or robustness of Bayesian networks to see the effect of imprecision in probabilities on the network performance. In other words, model validation is necessary for a Bayesian network to achieve satisfying performance. For this purpose, a sensitivity analysis technique is often used to investigate the effect of probability parameter changes on the performance of Bayesian networks. The traditional approach takes a few steps to get statistically sound results [56, 40, 52]. First certain level of log-normal noise is added to probability parameters under investigation using Monte Carlo sampling. Second many Bayesian networks are generated with the same graphical structure as the original network but different probability parameters. Third the newly generated networks are used for reasoning in the test scenarios. And then the sensitivity of the network is analyzed by looking at the changes of the posterior probabilities of the variables of interest. However, the valid noise level may be rather limited since it models the uncertainty of expert knowledge and the biased estimate of probabilities should not be too far away from the real probability values. Furthermore, the used measure, average changes of the posterior probabilities, may not be an indicative measure for sensitivity of Bayesian networks. Using indicative measures for sensitivity, Bayesian networks may actually show various sensitivities to the imprecision of the probability parameters [40, 52].

Following the findings that Bayesian networks can have different sensitivities to variations in probability parameters, it is desirable to obtain low sensitive, i.e., highly robust, Bayesian network models among other candidates that well represent the domain knowledge. This is a typical model selection problem in learning Bayesian networks. In Bayesian network learning, model selection is to find a graph structure which best describes the probabilistic dependencies represented by the data. Traditional approaches to model selection are either scored-based or constraint-based. Score-based approach uses scoring metrics to guide a heuristic search in the space of possible graph structures. The existing scoring functions such as BIC [63], BDe [30] and MDL [44], trade the likelihood against the complexity of the constructed models. An alternative approach to model selection is constraint-based approach that utilizes the property of Markov equivalence. In this approach, models are selected based on the Markov property, i.e., the dependence relationships between nodes, which can be detected by statistical tests. The output is a partially directed acyclic graph (PDAG) that represents the Markov equivalence class [71, 55]. Interestingly, it was proved that the Markov equivalent graphs [10] are score equivalent in terms of BIC, BDe, and MDL etc. It is interesting to see whether or not the Markov equivalent graphs are sensitivity equivalent, if not, what relationships are there between the sensitivities of the Markov equivalent graphs, can sensitivity be a model selection criterion?

On the other hand, a Bayesian network often shows different sensitivity to different sets of its probability parameters. That means, some probability parameters may have a larger effect on the network's performance than others. In other words, some probability parameters are more influential on the network performance. The erroneous estimate of these important parameters may greatly deteriorate the network quality. This happens in both knowledge engineering approach and the machine learning approach to building Bayesian networks. Because there are huge number of conditional probability values in a Bayesian network needs to be quantified, the data base is often relatively scarce for an accurate estimate of the numeric parameters in learning Bayesian networks, and results in erroneous values, especially for rare-event probabilities. To get around the problem, domain knowledge is utilized in Bayesian learning approach. The Bayesian learning method views the prior knowledge of a domain expert as equivalent to a pseudo (or imaginary) data set drawn from Dirichlet distributions [27]. The Dirichlet exponent parameters (also called hyperparameters) are used to represent the equivalent sample size of the expert's prior knowledge [14]. Unfortunately, the number of the hyperparameters is as large as the number of the probability parameters in a Bayesian network. Most learning algorithms simply use noninformative hyperparameters, and are subsequently ignorant of the variety of valuable domain knowledge.

1.3 STATEMENT OF THESIS

Given the identified challenges in building Bayesian networks in the previous section, this thesis explores possible solutions to the following research questions:

- What kind of tools can we develop for efficient probability elicitation?
- How can we evaluate elicitation methods for probabilistic models systematically and objectively?
- In evaluating sensitivity of Bayesian networks, what is the valid range of the noise to simulate the small variation in probability estimates? What is an indicative measure for sensitivity of Bayesian networks?
- Are Markov equivalent graphs necessarily equivalent in sensitivity? Is sensitivity a suitable criterion for model selection in learning Bayesian networks?
- How can we refine probability parameters with less effort but achieve reliable network performance?

To answer the first question, I have investigated the existing graphical tools for elicitation of probabilities with an emphasis on user interface design. A set of user interface tools were developed for efficient elicitation. These tools focus on two aspects of probability elicitation: (1) interactive graphical assessment of discrete probability distributions, and (2) navigation through conditional probability tables. Based on what is known about graphical presentation of quantitative data to humans, I offer several useful enhancements to probability wheel and bar graph, including different chart styles and options that can be adapted to user preferences and needs. Realizing that the navigation in very large conditional probability tables (CPTs) may decrease the efficiency in probability elicitation if the navigation tool is not effective, I developed two new graphical views that aid CPT navigation: the CPTree (Conditional

Probability Tree) and the sCPT (shrinkable Conditional Probability Table). I will present the results of a simple usability study that proves the value of these tools [77].

With regard to the second research problem, I invented an objective approach for evaluating probability and structure elicitation methods in probabilistic models. The main idea is to use the model derived from an expert's experience rather than the true model as the standard to evaluate the elicited model. The method draws on ideas from research on learning Bayesian networks: if we assume that the expert's knowledge is manifested essentially as a database of records that have been collected in the course of the expert's experience, and if this database of records were available to us, then the structure and parameters of the expert's beliefs could be reliably constructed using techniques for Bayesian learning from data. This learned model could, in turn, be compared to elicited models to judge the effectiveness of the elicitation process. I will describe a general procedure by which it is possible to capture the data corresponding to the expert's beliefs and present a simple experiment in which this technique is utilized to compare three methods for eliciting discrete probabilities: (1) direct numerical assessment, (2) the probability wheel, and (3) the scaled probability bar. I will show that for our domain, the scaled probability bar is the most effective tool for probability elicitation [76].

Empirical study of sensitivity analysis on a Bayesian network examines the effects of varying the network's probability parameters on the posterior probabilities of the true hypothesis. One appealing approach to modeling the uncertainty of the probability parameters is to add normal noise to the log-odds of the nominal probabilities. However, I will argue that differences in sensitivities found on true hypothesis may only be valid in the range of standard deviations where the log-odds normal distribution is unimodal. I will also show that using average posterior probabilities as criterion to measure the sensitivity may not be the most indicative, especially when the distribution is very asymmetric as is the case at nominal values close to zero or one. It is proposed, instead, to use the partial ordering of the most probable causes of diagnosis, measured by a suitable lower confidence bound. I will also present the preliminary results of our sensitivity analysis experiments with three Bayesian networks built for diagnosis of airplane systems. The results show that some networks are more sensitive to imprecision in probabilities than previously believed [40].

To get insight into the appropriateness of using sensitivity as model selection criterion for learning Bayesian networks, I investigated the relationship between sensitivities of Markov equivalent networks that represent the same joint probability distribution but differ in the network structure. I proved that Markov equivalence does not necessarily implies sensitivity equivalence, and the relationship between the sensitivities of Markov equivalent networks can be mathematically expressed by a simple linear function. The coefficient of the function reduces to the local probability distribution for the equivalent networks that transforms one to another by a series of edge reversals. Based on the finding, I argue that sensitivity is actually not an appropriate criterion for model selection.

Since the performance of a Bayesian network may be sensitive to different probability parameters, the degree of sensitivity to a parameter x indicates the importance of x for the quality of the network. This provides a possibility of achieving reliable performance without getting accurate estimates for every probability parameter in a Bayesian network. Obtaining accurate assessments for those parameters that are important may be sufficient. I present a method for a selective update of the probabilities based on the results of sensitivity analysis when learning a Bayesian network from data. The process begins with a rough estimate of probability distributions with uniform hyperparameters in Bayesian learning. Then it performs the sensitivity analysis on the Bayesian network populated with the rough estimates of probabilities. This way, we can identify the most important, i.e., the most influential, probability parameters with respect to the query nodes. And these important probabilities can be updated to more accurate values with informative hyperparameters extracted from expert knowledge or more data. The process is repeated until refining the probabilities any further does not improve the performance of the network [78].

1.4 OVERVIEW

This chapter briefly introduces the challenges and some solutions in building Bayesian networks with respect to elicitation, evaluation and learning. The rest of the thesis elaborates the research work and gives the detailed discussions.

Chapter 2 introduces the user interface tools that I developed for navigation in conditional probability tables and elicitation of probabilities in Bayesian networks.

Chapter 3 describes the method for objective evaluation of elicitation schemes for probabilistic models.

Chapter 4 investigates the valid noise level to model uncertainty of probability parameters and the indicative measure of sensitivity of Bayesian networks.

Chapter 5 proves the hypothesis that Markov equivalent graphs are sensitivity inequivalent, and argues that sensitivity may not be a suitable measure for model selection in learning Bayesian networks.

Chapter 6 describes the learning algorithms that use sensitivity analysis in Bayesian network learning for selective parameter update.

Chapter 7 summarizes the completed research work, and discusses the future research plan.

In Appendix , I present an efficient algorithm for sensitivity analysis that was based on junction tree inference algorithm and the relevance-based reasoning.

2.0 USER INTERFACE TOOLS FOR NAVIGATION IN CONDITIONAL PROBABILITY TABLES AND ELICITATION OF PROBABILITIES IN BAYESIAN NETWORKS

2.1 INTRODUCTION

Elicitation of numerical parameters is one of the most laborious tasks in building probabilistic models. The foremost problem is the large number of parameters required to fully quantify a model. For example, in *HEPAR* network [53], there are 72 variables. Full quantification of the *HEPAR* network required over 2000 numerical parameters. In most real problem domains, elicitation of numerical parameters is a dominant task in probabilistic modeling (e.g., [31], [33], [23]).

On the other hand, human judgement is prone to systematic errors (biases) that can be invoked by a variety of factors [39]. Elicitation of probabilities, if not performed carefully, can result in poor quality estimates. Behavioral decision theorists have proposed several elicitation approaches that minimize the risk of bias. However, these methods tend to be cumbersome and often infeasible for models that include more than a few variables because of the large number of elicitations required. Decision analytic practice is usually based on methods that require less effort and still protect subjective assessments from common biases.

A major obstacle to effective probability elicitation in Bayesian networks is navigation in large conditional probability tables (CPTs). In a CPT, a conditional probability distribution over a variable is required for each combination of values of its parents. The total size of the conditional probability matrix is exponential in the number of parents. For example, the CPT of a binary variable with n binary parents requires 2^{n+1} parameters. For a sufficiently large n, the 2^{n+1} numbers will not fit on the screen and the user will have to spend a considerable effort in navigating through them. The problem of navigation in conditional probability tables has not really surfaced in the field of decision analysis, as the size of typical decision-analytic models has been limited to a handful of variables. Bayesian networks, however, quickly reach the size of tens or hundreds of variables. It is not uncommon to see a variable with as many as ten parents, which, even if each parent is binary, results in CPTs consisting of thousands of elements. Existing software packages implementing Bayesian networks and influence diagrams have coped with the problem in various ways, few of which seems to follow established principles of good human-computer interface design. Users have to scroll back and forth to locate a particular item in a table or a list or have to manually give the combination of parent states in a combo box. Separate tables, applied in some solutions, require significant mental effort when users shift from one view to another.

While the problem of graphical elicitation of probabilities is easier to cope with, my investigation into the existing implementations has also shown a lot of room for improvement. The only graphical tool for probability elicitation implemented seems to be the probability wheel, which visualizes discrete probability distributions in a manipulable pie-chart graph. However, probability wheel has some problems and may sometimes be not the best tool for graphical elicitation of probabilities. A pie chart is known to make the judgement of part-to-part proportion difficult and is often inferior to a bar graph. Also, the labeling style applied and the overall design of interaction with the user is far from ideal in a typical implementation.

In this chapter, I describe a set of tools that were developed to help improve navigation through large CPTs and improve interactive assessment of discrete conditional probability distributions. I developed two new navigation tools: the CPTree (conditional probability tree) and the sCPT (shrinkable conditional probability table). The CPTree is a tree view of a CPT with a shrinkable structure for any of the conditioning parents. The sCPT is a table view of a CPT that allows to shrink any dimension of the table. Both CPTree and sCPT allow a user to efficiently navigate through CPTs and, in particular, to quickly locate any combination of states of conditioning parents. I enhanced the probability wheel by providing alternative chart styles, bar graphs and pie charts, to support different kinds of proportion judgement. The pie chart and bar graph support locking functions for those probabilities that have been elicited. Two labeling styles are provided: text and percentage. I use center-surround labels for the pie charts. While both tools are viable alternatives for probability elicitation, pie chart supports more accurate assessment of part-to-whole proportion whereas bar graph performs better for part-to-part proportion judgements. Both tools support context-specific independence and allow for elicitation of several distributions at a time, if these are identical.

The remainder of this chapter is organized as follows. Section 2.2 describes existing approaches to navigation in CPTs and existing implementations of graphical probability elicitation tools. Section 2.3 describes the CPTree and sCPT and discusses the enhancements to the graphical elicitation tools. I report some findings from an empirical study based on the developed tools in Section 2.4.

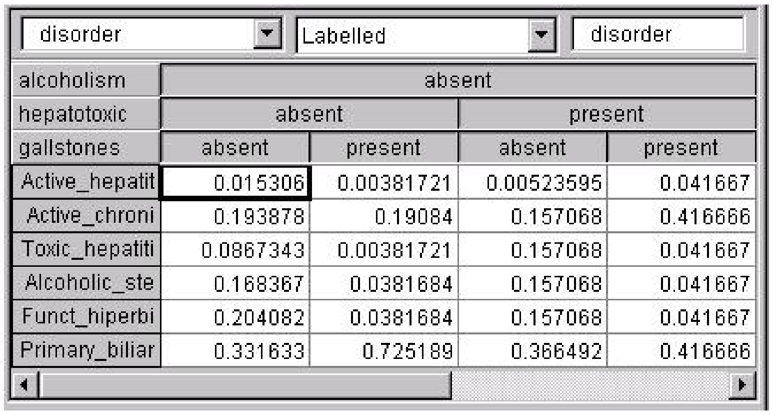
2.2 EXISTING GRAPHICAL TOOLS

Most of the existing probabilistic modeling systems provide graphical interface for navigation in conditional probability tables. Some of them supply a probability wheel as a graphical tool for subjective probability elicitation. In this section, I analyze critically existing graphical elicitation and navigation tools. An annotated list of these systems (including *GeNIe*, *Data*, *Dpl*, *Ergo*, *Hugin*, *Msbn*, and *Netica*) along with links to their web sites, where demonstration versions can be examined, is available on a web page for the INFORMS' Society for Decision Analysis at *http://www.sis.pitt.edu/∼dsl/links.htm.*

2.2.1 Navigation in Conditional Probability Tables

There are several existing ways of dealing with the problem of navigation in conditional probability tables.

In a flat table, the solution adopted in *GeNIe 1.0* and *Hugin* (Figure 2.1), the header cells indicate parent states and the numerical cells display the conditional probability distributions. The parent states are organized in a hierarchical structure that labels the conditional probability distributions. A table is a natural view for multi-dimensional data and, when it fits on the screen, it is fairly easy to explore. However, when a table is larger than the available screen area (this happens very often given the exponential growth of CPTs), users have to scroll back and forth to locate a particular conditional probability distribution. Watts ([79]) observed that users of very large spreadsheets were often lost and ended up creating chapter maps to guide them in navigation.

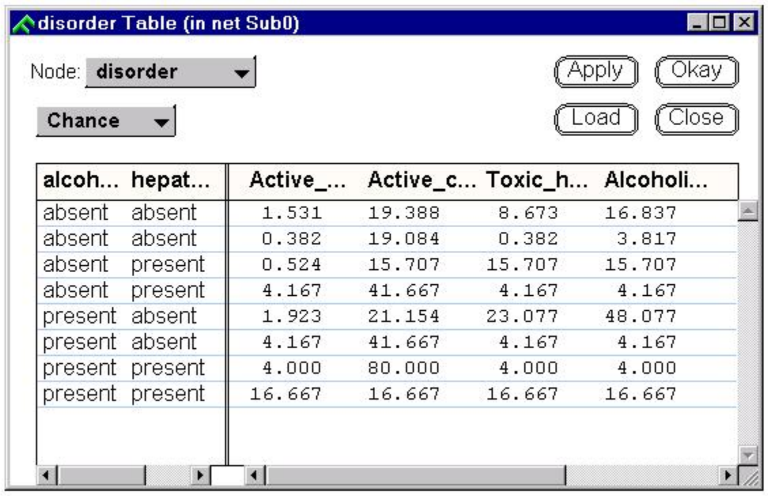


alcoholism	absent			
hepatotoxic	absent		present	
gallstones	absent	present	absent	present
Active_hepatit	0.015306	0.00381721	0.00523595	0.041667
Active_chroni	0.193878	0.19084	0.157068	0.416666
Toxic_hepatiti	0.0867343	0.00381721	0.157068	0.041667
Alcoholic_ste	0.168367	0.0381684	0.157068	0.041667
Funct_hiperbi	0.204082	0.0381684	0.157068	0.041667
Primary_biliar	0.331633	0.725189	0.366492	0.416666

Figure 2.1: *Hugin*'s Probability Definition Table for Node *Disorder* in the *HEPAR* Network.

Another approach is using a list, a solution applied in *Netica* (Figure 2.2). *Netica*'s navigation screen consists of a list of all possible combinations of parent states. The list of conditional probability distributions associated with each parent combination is shown next to the list of parent outcomes. Both lists are viewed by scrolling. If there are more items than those which can be shown in a list view, a scroll bar is provided for users to look for the hidden items. In Figure 2.2, two parents of node *Disorder* are shown in the parent list. The third one is hidden. The list in *Netica* can be viewed as a transposed matrix of the table in *GeNIe* and *Hugin*. But the hierarchical structure is not clear in the list. Users are required to manually traverse the hierarchy to determine its structure. Generally, lists are capable of providing detailed content information but are poor at presenting structural information. A great deal of effort is needed on the part of the user to achieve a mental model of the structure in large hierarchies.

Yet another solution is based on combo boxes, applied in *Msbn* (Figure 2.3). There is one combo box for each parent and it is used to select an outcome for that parent. Only one



alcoh...	hepat...	Active_...	Active_c...	Toxic_h...	Alcoholi...
absent	absent	1.531	19.388	8.673	16.837
absent	absent	0.382	19.084	0.382	3.817
absent	present	0.524	15.707	15.707	15.707
absent	present	4.167	41.667	4.167	4.167
present	absent	1.923	21.154	23.077	48.077
present	absent	4.167	41.667	4.167	4.167
present	present	4.000	80.000	4.000	4.000
present	present	16.667	16.667	16.667	16.667

Figure 2.2: *Netica*'s Probability Definition Table for Node *Disorder* in the *HEPAR* Network.

column of the CPT is visible at a time. In order to select a column, the expert has to assign values to all of the parents manually. When there are many parents, there is a danger that the user will forget to assign some of these combinations and, effectively, leave some of the probabilities unspecified.

Separate tables for parent combinations and conditional probabilities are yet another solution. *Ergo* uses two separate tables, one for a parent list and the other for a probability editor (Figure 2.4). When editing conditional probabilities for a node, the last parent is displayed in the probability editor table, and all other parents are displayed in the parent list. Separate tables show the conditional probability distribution for one combination of values of parents at a time, occupying relatively small screen space. However, it is important to recognize that shifts from one table to another can be cognitively costly [80].

A probability tree is a natural and familiar metaphor for the organization of conditional probability information. *Dpl* provides a probability tree showing all of the possible combinations of parent outcomes (Figure 2.5). In *Dpl*, the tree is always completely expanded and the entire tree appears in the available display space. The program shrinks the tree as needed to fit it on the screen. There is no zooming function for a clear view. The tree view provides a visual hierarchy of the context for specification of conditional probabilities. However, a

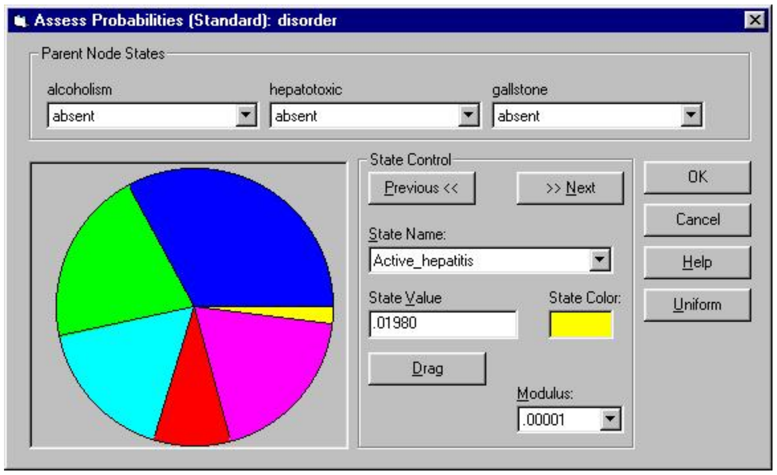


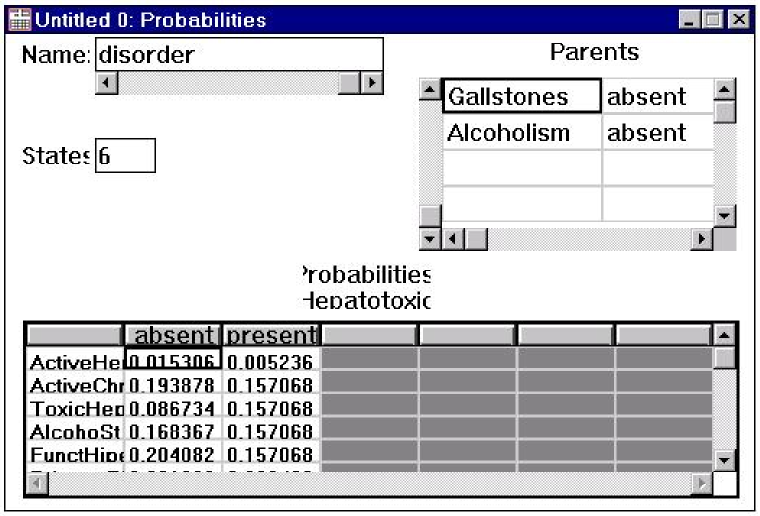
Figure 2.3: *Msbn*'s Probability Definition Table for Node *Disorder* in the *HEPAR* Network.

completely expanded tree in a restricted display space becomes quickly unreadable. It is almost impossible to navigate in the tree view without remembering the order of parents and their outcomes.

2.2.2 Elicitation of Probabilities

Probability wheel [70, 50] is probably the oldest and the most popular graphical tool for probability elicitation. It consists of a wheel with two adjustable sectors (traditionally colored red and blue) and a fixed pointer in the center. When spun, the wheel will finally stop with the pointer either in red or blue sector. The probability that the wheel will stop in the red sector is proportional to the sector size. The relative size of the two sectors can be adjusted until the expert judges that the event under consideration is equally likely as the event of the wheel stopping in the red region. In computer systems (e.g., *Data*, *Dpl* and *Msbn*), it is usually implemented as a pie chart. The pie chart is partitioned into several sectors representing each of the outcomes of the variable. The area of each sector is proportional to the probability of the corresponding outcome. The user can shrink or expand the proportion of each area by dragging its edge to the desired position.

While the probability wheel is a useful tool, it has several disadvantages. Probability


Untitled 0: Probabilities

Name: disorder

States 6

Parents

Gallstones	absent
Alcoholism	absent

'robabilities
Hepatotoxic

	absent	present				
ActiveHe	0.015306	0.005236				
ActiveChr	0.193878	0.157068				
ToxicHep	0.086734	0.157068				
AlcohoSt	0.168367	0.157068				
FunctHipe	0.204082	0.157068				

Figure 2.4: *Ergo*'s Probability Definition Table for Node *Disorder* in the *HEPAR* Network.

elicitation involves complex perceptual processes that include judgements of proportions, comparisons, and changes. Graphical tools help experts to estimate proportions, and to dynamically change the sizes of component parts in the graph until the sizes reflect personal beliefs of the experts. When eliciting subjective probabilities, some experts find it difficult to judge a part-to-whole proportion. They often use a larger value as reference point and compare smaller values with it for a part-to-part judgement. Although empirical studies have demonstrated that pie charts lead to a higher accuracy in part-to-whole judgement of proportion, they have shown inferiority of pie charts to bar graphs in part-to-part comparison and change perception [11, 67, 35, 36]. A pie chart has the additional disadvantage of being too fragmentary when partitioned into many sectors. Preece *et al.* ([58]) recommended that a pie chart should be used only when there are fewer than five sectors.

Lack of user control is another problem with the existing implementations of probability wheel. Since total probability is always equal to one, a specific change in probability of one outcome results in proportional changes in the probability of the remaining outcomes. The proportion of the remaining outcomes usually stays the same. However, this automatic adjustment of probabilities is frustrating when an expert just wants to modify some of

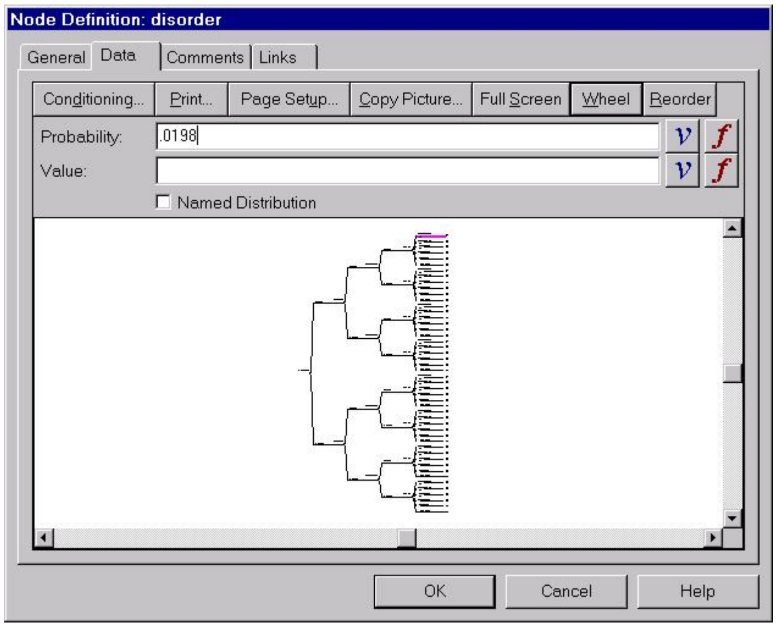


Figure 2.5: *Dpl*'s Probability Definition Table for Node *Disorder* in the *HEPAR* Network.

the numbers and keep other numbers unchanged. This happens, for example, when the expert accepts some probabilities encoded and only wants to graphically modify remaining probabilities. It seems necessary to let the expert be in control of when and to which probabilities the automatic changes apply.

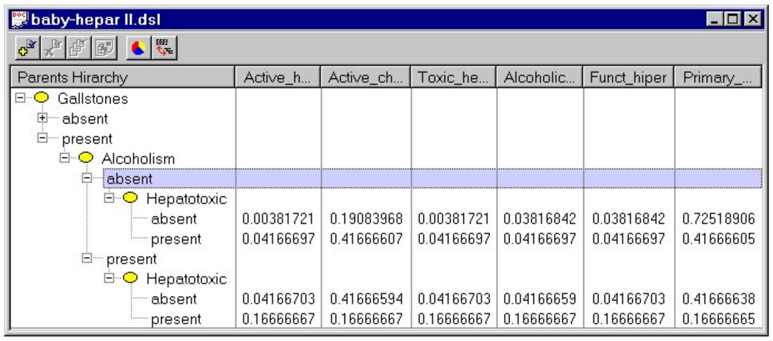
Besides, the legend annotation of a pie chart requires a mapping procedure to recognize which sector represents which outcome of the variable. When a variable has many outcomes, it becomes difficult for the user to search in a long list for those mappings between outcomes and sectors. The coordination of human focal attention and orienting perceptual functions such as peripheral vision supports the process of knowing where to look, and when [60]. Woods [[80]] suggested the use of a center-surround technique, which is an annotation style that labels wedges around sectors of the pie. A direct label is provided for each sector, thus reducing the mental workload of the users.

2.3 GRAPHICAL TOOLS DEVELOPED

2.3.1 Navigation in CPTs

As I discussed in the previous section, the plain form of a CPT is hard to navigate due to the exponential growth of its size. In order to address this problem, I adopted the tree metaphor for hierarchical visual representations and developed two browsing tools: the CPTree (conditional probability tree) and the sCPT (shrinkable conditional probability table).

2.3.1.1 Conditional Probability Tree The CPTree (Figure 2.6) is a tree view of a node's CPT. In a CPTree, every parent variable is represented by two levels of nodes, the name level and the outcome level. The name level is comprised of a single node indicating the variable's name. The outcome level includes nodes for all possible outcomes of the corresponding variable. The name node always appears as the parent of the outcome nodes for the same variable. Each name node is a child of an outcome node of the previous parent variable. The root of the CPTree is the name node of the first parent variable. The path from root to a leaf specifies a combination assignment to values of parents. On the right-hand side of the tree is a table in which each row is associated with a branch in the tree. The table defines the probabilities conditional on the context specified by the branches.



Parents Hirarchy	Active_h...	Active_ch...	Toxic_he...	Alcoholic...	Funct_hiper	Primary_...
Gallstones						
absent						
present						
Alcoholism						
absent						
Hepatotoxic						
absent	0.00381721	0.19083968	0.00381721	0.03816842	0.03816842	0.72518906
present	0.04166697	0.41666607	0.04166697	0.04166697	0.04166697	0.41666605
present						
Hepatotoxic						
absent	0.04166703	0.41666594	0.04166703	0.04166659	0.04166703	0.41666638
present	0.16666667	0.16666667	0.16666667	0.16666667	0.16666667	0.16666665

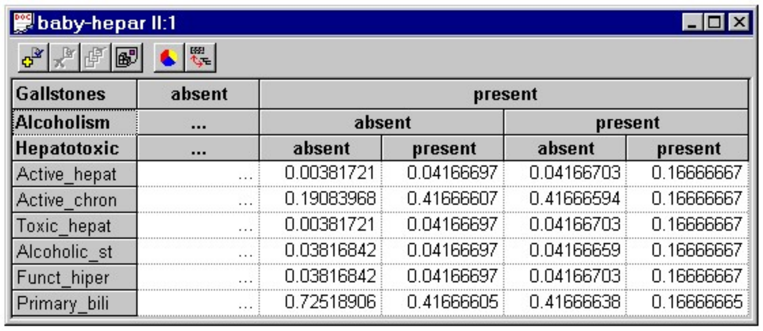
Figure 2.6: CPTree of Node *Disorder* in a Simplified Version of the *HEPAR* Network.

With shrinking and expanding function, an expert can quickly go to the branches of interest while collapsing others in order to optimize screen use. A click on the corresponding toolbar icon will bring up a probability wheel for the probability distribution conditioned on the selected combinations of the parent assignments for the current node represented by the CPTree. A combination of parent assignments is specified by a path from the root to a leaf in the CPTree. If a leaf node is selected, the conditioning context is given by all of the parent assignments along the path. If an internal node in the CPTree is selected, the context is given by a partial combination of parent assignments. Only those parents that are between the root of the tree and the selected node will count. For example, in Figure 2.6, when the tree node *absent* under *Alcoholism* is selected, the selected branch specifies the context of $Gallstones = present \wedge Alcoholism = absent$. The state of *Hepatotoxic* is irrelevant. In other words, the probability distribution over *Disorder* is independent of the state of *Hepatotoxic*. The selected branch defines a context-specific independence (also called asymmetric independence) relationship [65, 2, 28] between the current variable, *Disorder*, and its parent, *Hepatotoxic medications*.

The design of the navigation interface allows the user to dynamically change the order of the parents in the navigation windows. Many times the users of *GeNIe 1.0* found the order of node parents counterintuitive because it did not follow the temporal or causal order. Changing the order of parents as the user desires allows the user to compose the most natural order of conditioning events. Secondly, it facilitates easy encoding of context-specific independence.

Multiple selection of branches is also supported. By selecting multiple branches and then triggering graphical elicitation through the probability wheel, experts can give their assessment for those conditional probabilities that are numerically identical but different in conditions. In Figure 2.6, the conditional probabilities under the context of $Gallstones = present \wedge Alcoholism = absent \wedge Hepatotoxic = present$, and the conditional probabilities under the context of $Gallstones = present \wedge Alcoholism = present \wedge Hepatotoxic = absent$ can be estimated at the same time by selecting both of the corresponding branches. Using this multiple assignment, experts can save a lot of duplicate input, which often happens in flat CPTs of current graphical probabilistic modeling development environments.

2.3.1.2 Shrinkable Conditional Probability Table The sCPT (Figure 2.7) includes virtually all of the functions implemented in the CPTree. Double-clicking on a header item triggers the shrinking or expanding of the columns that it covers. We can view the sCPT as a tree-structured conditional probability table. All the columns in the covered range of a header item constitute its children items. A branch can be traced from the first header row through its covered range. With the aid of probability tools, experts can assign the same probability values to multiple groups under distinct branches.



Gallstones	absent	present			
Alcoholism	...	absent		present	
Hepatotoxic	...	absent	present	absent	present
Active_hepat	...	0.00381721	0.04166697	0.04166703	0.16666667
Active_chron	...	0.19083968	0.41666607	0.41666594	0.16666667
Toxic_hepat	...	0.00381721	0.04166697	0.04166703	0.16666667
Alcoholic_st	...	0.03816842	0.04166697	0.04166659	0.16666667
Funct_hiper	...	0.03816842	0.04166697	0.04166703	0.16666667
Primary_bili	...	0.72518906	0.41666605	0.41666638	0.16666665

Figure 2.7: A Shrinkable CPT of Node *Disorder* with the *Gallstones=absent* Branch Shrunk.

Compared to the CPTree, the sCPT has a higher data density, which is a desired property of graphical displays of quantitative data, defined as the ratio of the amount of data displayed to the area of the graphic [72]. In the CPTree, a considerable amount of screen area is consumed at the expense of displaying the dependence context for conditional probabilities. This results in the difficulty of the CPTree to represent a node with a large number of parents. However, for some users, the CPTree may visualize the structure of conditional dependence more intuitively.

2.3.2 Probability Assessment Tools

I have designed a graphical tool for elicitation of discrete probability distributions that implements two chart styles: pie charts and bar graphs. When the user selects the assessment tool within the navigation window, the tool is presented in a separate pane of the splitter window of the navigation tool.

The pie chart (Figure 2.8) combines easy user interaction with intuitive illustration. To change a probability of an outcome of a variable, the user drags the handle of the corresponding sector in the pie to its new position. During the dragging process, the pie is redrawn, showing the new partition resulting from the probability changes. When one probability is being changed, the remaining probabilities are automatically adjusted proportionally. If the user wants to keep the probabilities of some events intact, she or he can simply click the right mouse button on the sectors corresponding to these events to lock them before beginning the dragging process. A right click on a locked sector unlocks it. A locked sector of the pie is shaded out and drawn slightly outside the pie, visually communicating the idea that this part of pie is cut off and cannot be changed. In Figure 2.8, two outcomes of *Disorder*: *Toxic_hepat* and *Active_chron* are locked and shaded out of the whole pie.

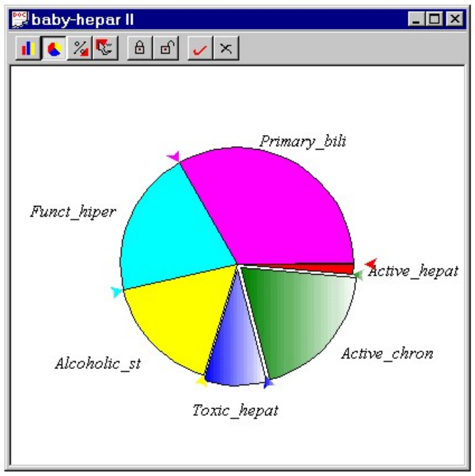


Figure 2.8: Pie-chart-styled Probability Wheel

Pie-chart-styled Probability Wheel for Node *Disorder* in the *HEPAR* Network with two Locked Sectors: *Toxic_hepat* and *Active_chron*.

The bar graph (Figure 2.9) provides a similar functionality. The user can adjust the length of a bar by dragging the handle at its end horizontally to a new position. The unlocked bars are changed proportionally, while the locked bars remain unchanged during the adjusting process. All locked bars are shaded in their vertical color gradients. Figure 2.9 shows a probability elicitation tool styled as a bar graph with probability scale appearing on the bottom of the graph. *Toxic_hepat* and *Active_chron* are locked and shown in their vertical color gradients.

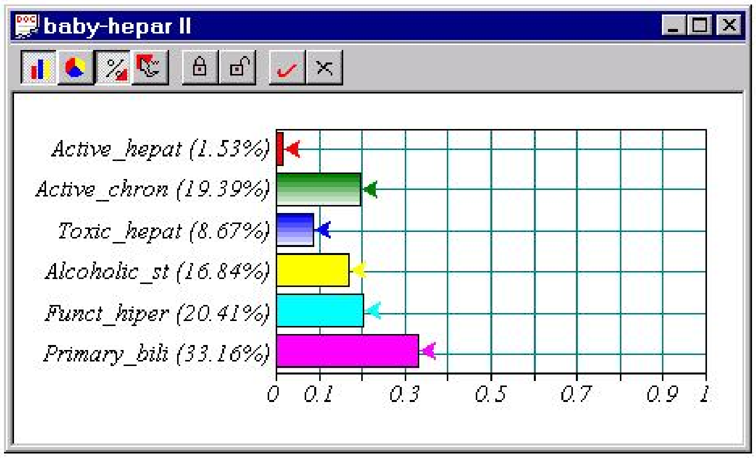


Figure 2.9: Bar-graph-styled Probability Tool

Bar-graph-styled Probability Tool for Node *Disorder* in the *HEPAR* Network with two Locked Sectors: *Toxic_hepat* and *Active_chron*. Note Optional Percentage Labeling.

In addition, two labeling options are provided for both chart styles. One is simple text of the outcome name as shown in Figure 2.8. The other is the name plus its probability as shown in Figure 2.9. Text labeling eliminates the interference of numbers and leads to a qualitative estimation from experts. Percentage labeling allows experts to see the exact numerical parameters corresponding to the manipulated graph components.

In addition, I use center-surround labels for the pie chart. Labels are positioned outside the pie near their corresponding sectors. This supports the user's perceptual process of knowing where to look and when, and reduces the mental workload of mapping labels from legend annotation to the corresponding sectors in the pie. When there are overlaps for two adjacent labels, only the last one is displayed, the next label is hidden.

Both charts are viable alternatives for probability elicitation tools, although they serve different purposes. Pie charts are more natural to show the relative distribution of data among sectors that make up a whole. Generally they support more accurate assessment of part-to-whole proportion. But the bar graphs are supplemented with a scale ranging from zero to one, which also facilitates assessment of part-to-whole proportion. On the other hand, people sometimes do not have a clear idea about what proportion a part takes up in a whole. But they often can give a proportion of one part to another by comparing them. According to the ranking of human perception identified by Cleveland and McGill ([11]), people usually produce faster and more accurate judgements when comparing position and length than when comparing angle or area. In addition, bar graph has an advantage over pie chart in the perception of change. People can easily capture small changes in a bar graph. Thus, a bar graph can be expected to allow for a better performance in probability estimation based on part-to-part proportion.

2.4 IMPLEMENTATION AND EMPIRICAL VERIFICATION

I have implemented the tools described in this chapter in *GeNIe 2.0*, an environment for building graphical decision-theoretic models, under development at the Decision Systems Laboratory, University of Pittsburgh. User interface has received a considerable attention in *GeNIe*. I believe that *GeNIe*'s growing popularity can be in part attributed to our attention to detail and the resulting powerful, yet pleasant interface. One of my objectives is to enhance *GeNIe*'s interface so that it becomes natural and easy to use for both experts and novices. I believe that it is not the speed of inference but rather the quality of the user interface that will increase the popularity of decision-theoretic methods. In my experience, reasoning in most practical models is sufficiently fast. The current bottleneck is in building models. Therefore, techniques that facilitate model building and intuitive interaction with the system are worth pursuing, even if they are cumbersome to implement in software.

All *GeNIe 2.0* windows are fully resizable. Users can always see a larger view of a CPTree or a probability wheel by enlarging an appropriate window. The size of the pie chart or bar graph is adjusted automatically to fit in the newly resized window. A relevant detail of our implementation is that *GeNIe*'s models are always syntactically correct at any stage of model development. A newly added node, in particular, has by default two outcomes, *State0* and *State1*, that are uniformly distributed. Any additional operation preserves this correctness. A negative side-effect of this is that the program does not have a clear way of showing which probabilities have been elicited and which have not.

While there have been other studies testing graphical representation of numerical data (e.g., [26], [68]), none of them focused on elicitation of probabilities. I tested empirically the two graphical probability elicitation methods, pie chart and bar graph, on a task involving elicitation of conditional probability distributions [76]. The results of my test have shown statistically significant differences in both speed and accuracy between each of the two methods and direct elicitation of numerical probabilities. Even though graphical probability elicitation methods were both faster than direct elicitation of numerical probabilities, bar graph was a clear winner in terms of both accuracy and speed (11% more accurate and 41% faster than direct elicitation and 3% more accurate and 35% faster than the pie chart). In next chapter, I will describe the methods we used for evaluation of the elicitation tools and give more details of the experimental design.

Qualitative questionare was presented to the testing subjects on the usability of our interface tools. The questions include easiness of use, time to locate a conditional probability parameter, and the subjects' general preference, etc. The study has shown that our subjects valued highly the availability of alternative tools for navigation and elicitation, and in majority of cases preferred *sCPT* to *CPTree*, and graphical elicitation to direct numerical input of probabilities, though there were some exceptions. As far as preference between the two graphical modes is concerned, it varied between subjects, suggesting that a good tool should provide a variety of methods that can adjust to individual user preferences.

2.5 CONCLUDING REMARKS

The tools described in this chapter enhance greatly user navigation in CPTs during the process of model building and help to improve both the quality and speed of elicitation. Also, the flexible navigation and visualization of probability distributions help to detect unspecified probabilities and inconsistency in responses. Combined, these tools provide a pleasant and powerful visual environment in which experts can give their qualitative estimates of numerical probabilities. Except *CPTree*, all of the graphical tools I developed are now adopted in the formal release of *GeNIe 2.0*.

I did not use 3-D displays, even though extra dimensions are often decorative and attractive. Some experiments [69, 4, 66] evaluated 3-D graphs in a perception task of relative magnitude estimation. The results did not show an advantage of 3-D displays in accuracy and speed. The performance of 3-D displays depends on the graphs and tasks. Compared to their 2-D counterparts on relative magnitude estimation, 3-D pie charts result in lower accuracy, and the 3-D bar graphs require a longer elicitation time.

While the tools that I have designed and implemented may be applicable to other graphical probabilistic structures, such as chain graphs, I focused on Bayesian networks. Obviously, the tools are readily applicable to chance nodes in influence diagrams [38]. To apply the tools to the elicitation of utilities in influence diagrams, a small change in the value scales may suffice, since unlike probabilities, utilities do not require to sum up to 1 and thus normalization is unnecessary. The common practice in utility elicitation is to use integer values ranging from 1 to 100, where 1 indicates the lowest utility and 100 indicates the highest utility. Also, eliciting utilities is more a part-to-part comparison than a part-to-whole comparison, therefore, the bar chart may be a better choice for utility elicitation than the pie chart.

In addition, these tools can be extended to chance nodes described by canonical probabilistic interactions, such as Noisy-OR or Noisy-AND nodes. As a matter of fact, the latest version of GeNIe has already support for the elicitation of Noisy gate parameters.

Addressing suggestions collected from *GeNIe*'s many users gives us an opportunity to learn problems and opportunities of enhancements. The program is under continuous development. One enhancement is to support user-adjustable level of granularity in graphical elicitation. The user will be able to set a given precision level, for example, 0.001, in which case all numbers obtained from the graphical elicitation procedure will be rounded to three places after decimal point. Another enhancement is allowing for the probability tables to be constructed from a mathematical expression involving parent variables rather than elicited directly. Yet another useful enhancement to the bar graph tool is marking it with user-defined probability scales, such as verbal probabilities, that will for some users enhance the elicitation process even further.

One limitation of the current version of the assessment tools is lack of support for elicitation of very small probabilities. Due to the screen resolution restriction and sensitivity of mouse movement, it is hard to capture very small changes of mouse position. Therefore, it is impossible to distinguish between low probabilities such as 0.000001 and 0.00001, even though they are orders of magnitude apart. Such values have to be entered manually. A good solution applied by others is to use log-scale [49]. This requires elicitation of both order of magnitude and the precision value at that granularity level. The order of magnitude is simple integer value and can be easily estimated directly. The precision value can then be elicited using the graphical tools. In other words, the combination of direct elicitation method and the use of graphical tools may be a better solution for eliciting very small probabilities.

3.0 EVALUATING ELICITATION SCHEMES FOR PROBABILISTIC MODELS

3.1 INTRODUCTION

As more and more decision-analytic models are being developed to solve real problems in complex domains, extracting knowledge from experts is arising as a major obstacle in model building [24]. Quite a few methods have been proposed to elicit subjective probabilities from domain experts. These techniques balance quality of elicitation with the time required to elicit the enormous number of parameters associated with many practical models. Structure elicitation is likewise a tedious problem and formal techniques for this task are even less mature. Systematic evaluation and comparison of different model elicitation methods are thus becoming of growing concern.

In Bayesian probabilistic models, encoded probabilities is often considered reflecting the degree of personal beliefs of the experts, though sometimes the probabilities can be data-driven and not subjective. The sole purpose of probability elicitation is to extract an accurate description of the expert's personal beliefs. In order to judge whether the elicitation procedure has produced an accurate model, therefore, the elicitor must know intimate details about the expert's knowledge. Unfortunately, these details that the elicitor is seeking from the start are hidden from explicit expressions; so it has not been possible to evaluate elicitation schemes directly. Less direct methods are the only possibility.

In this chapter I present an objective approach for evaluation of elicitation methods that avoids the assumptions and pitfalls of existing approaches. The technique is much closer to the ideal "direct" comparison between the elicited network and the expert's beliefs. The main idea is to simulate the training/learning process of an expert by allowing the trainee to interact with a virtual domain. Underlying the domain is a Bayesian network that is used to stochastically update the state of the world in response to the subject's interaction. Then by recording every state of the world that is experienced by the trainee, we can effectively gain direct access to the trainee's knowledge. It is quite an established fact that people are able to learn observed frequencies with an amazing precision if exposed to them for a sufficient time [25]. Therefore, after training, the trainee obtains some level of knowledge of the virtual world and, consequently, becomes an "expert" at a certain proficiency level. This knowledge, in the form of a database of records, D_{exp}, can be converted to an "expected" model of the expert, $\widehat{M}_{exp}$, by applying Bayesian learning algorithms to D_{exp}. Finally, this expected expert model can be directly compared to the model elicited from the "expert" to judge the accuracy of elicitation.

The approach captures a subject's state of knowledge of the probabilistic events in the toy world. The subject's experience with the toy world, rather than the actual model underlying the world, forms the basis of his or her knowledge. For this reason, the learned model should be the standard used to evaluate the elicitation schemes, rather than the original toy model. This technique allows us to avoid the expensive process of training subjects to fully-proficient expertise. For example, the expert's experience may have led him to explore some states of the world very infrequently. In this case, even if the elicitation procedure is perfect, the elicited probabilities of these states may be significantly different from the underlying model. Using the expert's experience rather than the original model gets around this problem completely because we know precisely how many times the expert has visited any given state of the world.

I use these techniques along with a toy cat-mouse game to evaluate the accuracy of three methods for eliciting discrete probabilities from a fixed structure: (1) direct numerical elicitation, (2) the probability wheel [70], and (3) the scaled probability bar [77]. I use mean squared errors between the learned and the elicited probabilities to evaluate the accuracy of each of the three methods. I show that for our domain the scaled probability bar is the most effective and least time-consuming.

Furthermore, the effectiveness of elicitation techniques is likely to task-dependent [70] or even expert-dependent [49], and there is no guidance as to how to select an appropriate method for various domains or experts. Our solution, presented here, will solve both of these problems by providing a general method whereby elicitation techniques can be compared for different domains and for different experts. Within a reasonable amount of time, we can determine which elicitation scheme works best for a given expert. Also, our experiment may shed some light on the well-known phenomenon of overestimation of small probability events.

In the following sections, I give a brief review of the existing evaluation techniques for probability elicitation methods. Then I present the relevant learning equations that allow us to capture a subject's beliefs in the form of learned network parameters. I describe the cat-mouse game that was used to train our subjects and collect data for learning. I present the experimental design and results followed by a discussion of my findings.

3.2 EVALUATION SCHEMES OF PROBABILITY ELICITATION METHODS

The difficulty in evaluating elicitation methods is that the true model is needed in order to be compared to the elicited model. Since the former is encapsulated in the expert's mind, it is not readily available for comparison. Previous comparisons of elicitation schemes followed essentially three lines of reasoning: (1) expert's preference, (2) benchmark model, and (3) performance measure.

The first approach, *expert's preference*, is based on the assumption that when an elicitation method is preferred by the expert, it will yield better quality estimates. While this assumption is plausible, to our knowledge it has not been tested in practice. There are a variety of factors that can influence the preference for a method, such as its simplicity, intuitiveness, or familiarity and these factors are not necessarily correlated with accuracy.

The second approach, *benchmark model*, compares the results of elicitation using various methods against an existing benchmark (gold standard) model $\widehat{M}$ of a domain (or a correct answer that is assumed to be widely known). Accuracy is measured in terms of deviation of the elicited model from $\widehat{M}$. For example, in Lichtenstein *et al.* [47] study of people's perception of frequencies of lethal events, there was a readily available collection of actuarial data on those events. Similarly, in Price's [59] study on effects of a relative-frequency elicitation question on likelihood judgment accuracy, general knowledge was used. An important assumption underlying the benchmark model method is that the model $\widehat{M}$ is shared by all experts. While in some domains this assumption sounds plausible, human experts notoriously disagree with each other [51, 12], and an experimenter is never sure whether the model elicited is derived from a gold standard model or some other model in the expert's mind. A debiasing training of experts with an established knowledge base may help to establish a benchmark model among them. For example, Hora *et al.* [37] trained their subjects in a formal probability elicitation process directed toward assessing the risks from nuclear power generating stations and compared two elicitation methods for continuous probability distributions. Their subjects were scientists and engineers who quite likely possessed extensive background knowledge about the risks. Effectively, it is hard in this approach to make an argument that the elicited model is close to the experts' actual knowledge, as the latter is simply unknown.

The third approach, *performance measure*, takes a pragmatic stand and compares the predictive performance of models derived using various methods. This reflects, in practice, how well calibrated the expert's knowledge is [46]. An example of this approach is the study performed by van der Gaag *et al.* [73], who used prediction accuracy to evaluate their probability elicitation method in the construction of a complex influence diagram for cancer treatment. While it is plausible that the quality of the resulting model is correlated with the accuracy of the elicitation method, this approach does not disambiguate the quality of the expert's knowledge from the quality of the elicitation scheme. A model that performs well can do so because it was based on superior expert knowledge, even if the elicitation scheme was poor. Conversely, a model that performs poorly can do so because the expert's knowledge is inferior, even if the elicitation scheme is perfect.

The next section introduces an evaluation method that I believe does not suffer from the problems identified in the existing evaluation schemes.

3.3 DATAMINING EXPERT BELIEFS

To evaluate the accuracy of an elicitation method is to make a judgment about how well the elicited model reflects the expert's real degree of personal belief. The closer the elicited model reflects the expert's real beliefs, the more accurate we say the method of elicitation is. But how can we measure an expert's real degree of personal belief? What can be used as a standard to evaluate the accuracy of a subjective probability? What we need is a method to capture the knowledge/beliefs that are held by our expert, then we need a method to construct a model entailed by that knowledge.

On the other hand, if we have a set of records in the form of a database, there are many machine-learning algorithms that are available to learn various types of models from that database. In this section I will present the key equations for learning probabilistic network models from data. A detailed description will be given in the next chapter.

3.3.1 Capturing the Expert's Knowledge

Complicating this effort is the fact that a person becomes an expert from a novice in a process of acquiring knowledge from a wide array of sources. Sources of knowledge range from reading books, talking to other experts, and most importantly for us, to observing a series of instances in the real world. In the method that I am proposing, we *create* an expert in a particular toy domain. In the process, we confine the source of knowledge available to that expert to be strictly of the latter type; namely, a series of observations of the real world. Being assured that our expert accumulates only this knowledge allows a particularly simple analysis of what our expert's beliefs about the domain should be. Throughout the chapter I will refer to this type of knowledge as *observational knowledge*.

If we assume that we have an expert whose entire knowledge of a domain is observational, then the expert's knowledge can be viewed as originating entirely from a database, D_{exp}, of records filled with instances of the domain our expert has committed to memory. If we further assume that we have recorded all relevant instances of the domain that our expert has actually observed into a database D, then our database D will be identical to D_{exp} under the assumption that the subject has paid attention to the occurrence of each event during his or her observation process. Thus, in any experiment designed to measure D_{exp}, it will be important to incentivate the subject in some way to pay attention to all events in the world.

3.3.2 Learning Bayesian Networks From Data

Assuming that we can assess D_{exp} correctly, we must now construct a probabilistic model that is most consistent with that data. Much work has been done on this problem in recent years [71, 14]. I will present just the key results of the Bayesian learning approach [29].

In the Bayesian approach, the data set D is considered fixed. To find a good network structure which encodes the physical joint probability distributions for multivariate $\mathbf{X}$ is to select a network structure S that has highest posterior probability $p(S|D)$. Assuming all possible structures are equally likely, $p(S|D)$ is proportional to the marginal likelihood of the data given structure, $p(D|S)$.

Let r_i be the number of possible values $x_i^1, \ldots, x_i^{r_i}$ for variable X_i (i.e., the *domain* of the variable X_i). And let q_i be the number of $\boldsymbol{Pa_i}$'s possible instantiation of X_i's parents in their joint combinatorial states (i.e., $\boldsymbol{Pa_i}$'s *configuration*). As a bit of notation, we define θ_{ijk} to be the probability parameter that $X_i = x_i^k$ given that $\boldsymbol{Pa_i} = Pa_i^j$, where $1 \leq k \leq r_i$, and $1 \leq j \leq q_i$. Under the assumption of complete data set D, Dirichlet prior parameters α_{ijk}, and parameter independence, the most likely structure can be selected using the following scoring metric,

$$p(D|S) = \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{\Gamma(\alpha_{ij})}{\Gamma(\alpha_{ij} + N_{ij})} \cdot \prod_{k=1}^{r_i} \frac{\Gamma(\alpha_{ijk} + N_{ijk})}{\Gamma(\alpha_{ijk})}, \tag{3.1}$$

and the expected value of the network parameters given a structure can be expressed as

$$\hat{\theta}_{ijk} = \frac{\alpha_{ijk} + N_{ijk}}{\alpha_{ij} + N_{ij}}. \tag{3.2}$$

In Equation 3.1 and 3.2, $\Gamma(\cdot)$ is the *Gamma-function* which satisfies $\Gamma(x+1) = x\Gamma(x)$ and $\Gamma(1) = 1$. N_{ijk} are the number of times in D that the variable X_i took on value x_i^k when its parents $\boldsymbol{Pa_i}$, took on configuration Pa_i^j, and $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$. Parameters α_{ijk}

are equivalent data samples that the experts have seen of the events $X_i = x_i^k$ conditioned on $\boldsymbol{Pa_i} = Pa_i^j$, and $\alpha_{ij} = \sum_{k=1}^{r_i} \alpha_{ijk}$. For a domain where the expert has little or no previous experience, we assume that all α_{ijk} are equal and small. Under this assumption, when no data are present for a particular (i, j) configuration of the world (i.e., $N_{ij} = 0$), then the N_{ijk} terms drop out of Equation 3.2 and the small equal priors produce a uniform distribution. However, even if a small amount of data is involved, then the priors have little influence on the parameters learned. But a larger α parameter weighs more subject's expertise in the estimates of the probability parameters.

For example, assume we are estimating the probability that a given coin will come up heads on an arbitrary toss, and assume that for our subject $\alpha_{heads} = \alpha_{tails} = 0.001$. Such a low prior indicates that our subject has had very little experience with coins, but still assumes initially that the coin is equally likely to be weighted towards heads or tails. After one flip of the coin (say a "heads" outcome), our subject's estimate of $P(heads) = \frac{1+0.001}{1+0.002} \approx 1$, so our subject's initial belief in uniformity has quickly been affected by the data. On the other hand, if our subject's initial beliefs were $\alpha_{heads} = \alpha_{tails} = 10$, then after one flip, his or her new assessment would be $P(heads) = 11/21 \approx 0.5$, much closer to his initial estimate. So the larger the α parameters are, the more weight our subject's expertise will play into his estimate of parameters.

We use the learned network as a standard to compare with the network elicited from the domain experts, when both are given the same data. The underlying assumption is that the experts make the same modeling assumptions as the Bayesian network learning method. These include, for example as we described earlier in this section, the Dirichlet prior parameters that simulate the pseudo data of the experts' domain knowledge. As we discussed in the above paragraphs, the larger Dirichlet prior parameters weigh more subject's expertise in the estimates of the probability parameters, or represent a more obstinate expert who do not change his or her estimate quickly. Since we do not know what value of the Dirichlet prior parameters are the best, we use several different assignments: $\alpha = 5$, $\alpha = 10$, and $\alpha = 10^{-4}$ and with each assignment, run the learning algorithm to get a Bayesian network model. All of the learned models are used to compare with the elicited model and we analyze the result by looking at the majority vote. This way we may get around the problem of the suitable value for the Dirichlet prior parameters.

3.4 EVALUATING ELICITATION SCHEMES WITH A TOY VIRTUAL WORLD

I designed a game in which a subject can move a cat to capture a mouse. I recorded the state changes of the cat-mouse game during the game playing process. What each subject experiences is unique and depends on the subject's actions. The recorded data allows for the learning of the probabilistic model of the toy world as seen by the subject. This learned model, in turn, gives a standard by which to measure the accuracy of the model elicited from the subject.

3.4.1 The Cat and Mouse Game: A Toy Virtual World

The toy world includes three characters: a cat and two mice. The objective of the game is for the cat to capture a mouse. There are twelve possible positions indicated by the grid cells in a horizontal line (see Figure 3.1). The cat can move one cell at a time between the current cell and an adjacent cell. One and only one mouse is present at any given time, and it can only bounce back-and-forth between two positions on each side of the screen. The two special positions for the mice are called *left-pos* and *right-pos* respectively. When the cat enters the cell/position where the mouse is located, it catches the mouse and the game is over.

The two mice are characterized by a color: *yellow* or *grey.* The cat can be in one of the four states: *normal, angry, frustrated*, and *alert.* Four icons are used to represent the states of the cat. Tables 3.1 and 3.2 illustrate the icons I used in the game. (The experimental subjects only saw the figures as the representation of the cat's states and mouse color. The verbal expressions are used to encode the cat's states and mouse color in the Bayesian network for the cat-mouse world due to the restraint of the modeling environment. These labels, "normal," "angry," etc., were not provided to the subjects during game play but

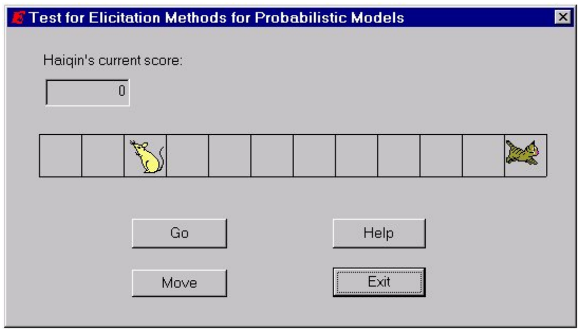


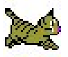
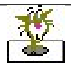
Figure 3.1: A screen snapshot of the cat-mouse game

were used, together with the pictures, to identify the states of the cat during the elicitation process.)

Table 3.1: Yellow mouse and grey mouse

yellow	*grey*

Table 3.2: Four states of the cat

normal	*angry*	*frustrated*	*alert*
			

Two buttons, labeled *move* and *go* respectively, are provided for the subject to manipulate the position of the cat. After the subject clicks a button, the cat moves to either the left or the right. Its moving direction is uncertain and depends on the current state of the world (i.e., which mouse is present, the position of the mouse, the state of the cat, and which button the subject has clicked). There is a short delay (half a second in our experiment) between button clicks during which the buttons are disabled. This prevents the subject from clicking the buttons too frequently and paying little attention to probabilistic relationships among the variables. It allows the subject to have enough time to observe how the moving direction of the cat is influenced by the state of the world and the subject's own actions. (The delay length of the disabled state of the buttons was selected based on our experiments with pilot subjects. I first tried 1 second and 2 seconds as the delay, but the pilot subjects soon complained the delay was too long and made the game boring. So I selected the maximum delay (half a second) with which the subjects still felt comfortable.)

After this delay, the toy world is updated to a new state. One mouse may disappear and another may show up instead. The mouse may appear in a different position. The cat may change its state. The two buttons for the subject's action become enabled.

In the beginning, the yellow mouse is put in the *left-pos* position. The cat is put in the farthest position away from the mouse. After the cat has caught a mouse, the game ends and a new round of the game begins. A new game always begins with the same initial positions for both the mouse and the cat. But the states of the rest of the world are uncertain.

Scoring rules are adopted to encourage the subject's involvement in the game. Whenever the cat captures a mouse, the subject's score increases as an incentive. Also, the game emits a celebratory sound as a reward for the subject.

3.4.2 The Bayesian Network for the Cat-mouse World

The cat-mouse world is based on a simple Bayesian network (Figure 3.2) consisting of five variables, *Action*, *Mouse Color*, *Mouse Position*, *Cat State* and *Cat Moving Direction*.

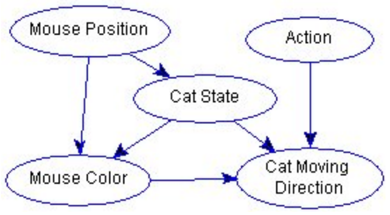


Figure 3.2: The Bayesian network of the cat-mouse world

Variable *Action* with two outcomes, *move* and *go*, models the observed subject's action. *Mouse Color* which could be *yellow* and *grey*, defines which of the two mice is present.

Mouse Position indicates the current position of the present mouse: *left-pos* and *right-pos*. *Cat State* represents four possible states of the cat: *normal*, *angry*, *frustrated*, and *alert*. The last variable *Cat Moving Direction* reflects the moving direction of the cat in the current step. Two directions are defined, *left*, and *right*.

The five variables influence each other probabilistically. The states of the variables change at each step according to the probabilities encoded in the network. Their probability distributions, either prior or conditional, were assigned randomly when the network was built to avoid biases to a particular probability distribution. One exception is the probability distribution of the *Action* node. The value of the *Action* node is always instantiated to the state that corresponds to the subject's action, and hence, the prior probability distribution becomes irrelevant. I chose the two nearly identical action words, *move* and *go*, to avoid any semantic difference which could have a potential influence on the subjects' preference.

3.4.3 The State Change of the World by Sampling

After the subject has clicked a button to take an action, the state of the world and the cat's moving direction are updated. The new states are selected by generating a stochastic sample on the cat-mouse network following the partial parent order of the graph. I use probabilistic logic sampling [32] to generate node states on the basis of their prior probabilities of occurrence. By choosing more likely states more often, I simulate the state changes of the toy world. The subjects are exposed to changes in the world that are an effect of their actions and the underlying joint probability distribution.

3.4.4 Collecting Data for Expert's Knowledge

Every time the state of the toy world changes, it is recorded automatically. In the data set, a case consists of the outcomes of all five variables encoded in the cat-mouse Bayesian network. The database of a subject's experience contains all states of the world that the subject has seen and it is the subject's observational knowledge about the toy virtual world. This knowledge comes completely from the subject's game-playing experience. Therefore, the records constitute a perfect data set for learning the subject's knowledge about the cat-mouse domain. There are various machine learning algorithms which can be applied to learn a Bayesian network for both structure and numerical parameters from data set. In the experiment for evaluating probability elicitation methods, the numerical parameters were learned from data using Equation 3.2 based on the original structure of the cat-mouse Bayesian network model.

3.5 EXPERIMENTAL DESIGN

I demonstrated the method in an experimental study that investigated the effectiveness of three elicitation methods: asking for numerical parameters directly, translating graphical proportions by using the probability wheel, and using the scaled probability bar. I used the graphical modeling system *GeNIe* [21] and build a module of cat-mouse game in *GeNIe* as well.

Subjects

The subjects were 28 graduate students enrolled in an introductory decision analysis course at the University of Pittsburgh, who received partial course credit for their participation.

Design and procedure

The subjects were first asked to read the instructions from a help window that introduced the game characters and the game rules. They were asked to pay attention to the probabilistic influences from the state of the toy world and their action choice to the direction of the cat's movement. The subjects were told that knowledge of these probabilistic relationships would help to improve their performance. To motivate the subjects to perform well, extra credit was offered for higher scores in the cat-mouse game and lower errors of estimates of the probabilities in elicitation.

Each trial included two stages. The subjects first played the cat-mouse game for 30 minutes. The data about their experienced states of the toy virtual world were automatically recorded. The data sets in the experiment typically contained between 400 and 800 records.

The second stage involved probability elicitation by each of the three elicitation methods. The subjects were shown the Bayesian network structure in Figure 3.2 and were asked to estimate the conditional probability table (CPT) for the node *Cat Moving Direction* by

1. typing the numerical parameters directly in conditional probability tables,
2. giving graphical proportions in the probability wheel, and
3. giving graphical proportions in the scaled probability bar.

I did not elicit the structure of the Bayesian network from the testing subjects for comparison because my purpose was to show how to utilize the evaluation method in elicitation task. In addition, it seems the only affordable way in the experiment for structure elicitation was direct asking.

I applied here a within-subject design in which each subject used the three elicitation methods. To offset the possible carry-over effects, I counterbalanced the order of method usage across the subjects.

The CPT elements θ_{ijk} elicited were compared to $\hat{\theta}_{ijk}$, the CPT elements learned by applying Equation 3.2 to the subjects' acquired data. The mean-squared error (MSE) of the parameters was calculated as

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(\theta_{ijk} - \hat{\theta}_{ijk})^2 \ .$$

In order to evaluate the speed of the elicitation methods, I also recorded the time taken for each elicitation procedure.

Since the domain experts had very little knowledge about the probability distributions of the cat-mouse domain prior to playing the game, I assigned small uniform values to the α parameters for the learning algorithm. For all possible values of i, j, k, I used the assignment $\alpha_{ijk} = 5$ and $\alpha_{ijk} = 10$ respectively. Obviously there is no absolute guidance to the selection of the values for the α parameters. These two values are chosen because they are relatively small compared with the typical data record size of 400 - 800, and the small number of probability parameters(approximately 25) that need to be elicited for quantifying the toy world Bayesian network. In order to test purely data-based learning, I also used $\alpha_{ijk} = 10^{-4}$.

Results

Table 3.3 shows the means and standard deviations of the mean squared errors (MSE) of the three elicitation methods when compared to the probabilities learned with different α parameters. A time comparison is also shown as the last two lines in the table. Figure 3.3 plots the elicitation time and MSE ($\alpha = 5$) for each of the three elicitation methods.

Table 3.3: Experiment Results For Elicitation Methods

Means (μ) and standard deviations (σ) for MSEs and time for each of the three elicitation methods

		wheel	**bar**	**direct**
$\boldsymbol{\alpha = 5}$	μ	0.0786	0.0758	0.0850
	σ	0.0384	0.0383	0.0448
$\boldsymbol{\alpha = 10}$	μ	0.0685	0.0663	0.0744
	σ	0.0376	0.0371	0.0431
$\boldsymbol{\alpha = 10^{-4}}$	μ	0.1217	0.1182	0.1283
	σ	0.0462	0.0468	0.0520
$\boldsymbol{time(minutes)}$	μ	6.6	4.9	6.9
	σ	4.0663	2.1141	2.3242

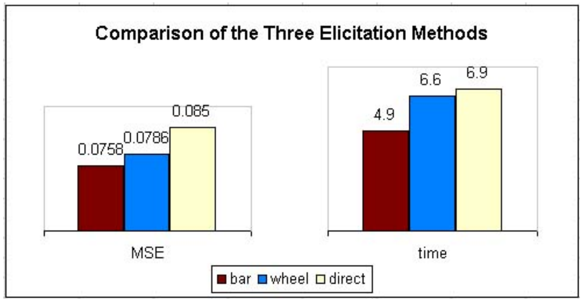


Figure 3.3: MSE($\alpha = 5$) and elicitation time for each of the three methods tested

For each pair of the elicitation methods, I conducted one-tailed, paired sample t test for comparison of accuracy corresponding to the learned results with different α's. One-tailed t test is chosen because we want to compare and see if one method has a better performance than the other in each pair, and if so, whether or not the difference is statistically significant. Three similar t tests were also done for time comparison. The p values resulted from the t tests are shown in Table 3.4.

Table 3.4: p values of one-tailed t tests for each pair of the elicitation methods

	bar *vs.* wheel	bar *vs.* direct	wheel *vs.* direct
$\alpha = 5$	0.19	0.03	0.07
$\alpha = 10$	0.22	0.03	0.07
$\alpha = 10^{-4}$	0.21	0.05	0.12
time	0.007	0.0005	0.37

The t tests showed that scaled probability bar performed significantly better than direct numerical elicitation, $p \leq 0.05$ for all three values of the learning parameter α. Probability wheel was marginally better than direct numerical elicitation, $p < 0.1$ for $\alpha = 5$ and $\alpha = 10$. The p value (0.12) was a little higher when $\alpha = 10^{-4}$. However, probability wheel was almost as accurate as scaled probability bar. Even though the latter had a slightly lower MSEs, the difference was not statistically significant ($p \approx 0.20$ under all values of α).

From the t tests conducted for the comparison of elicitation time, we can see that generally using scaled probability bar took the shortest time ($p \leq 0.007$). However, using probability wheel did not improve the time compared with direct numerical assessment ($p = 0.37$).

However, there were some outliers. Some subjects took a shorter time in direct elicitation than using the graphical tools. One interesting observation was that some people tried to move the bars to match a fine-grained number, e.g., 0.85, and found a hard time to do it since the mouse tracking changes rapidly and making such an input by graphical tools harder than just typing keyboard. These subjects might find the direct elicitation method easier to use and result in a faster estimate using direct elicitation method. This suggests that different users may have different preferences. A good system should provide more than one elicitation method to satisfy different users' need for a better performance.

Discussion

The experimental results showed that the learning approach to evaluate elicitation methods for probabilities is quite robust. From the results of the paired sample t tests, we can draw a conclusion about the accuracy of the three elicitation methods. Both the scaled probability bar and the probability wheel performed better than direct numerical elicitation, though the latter difference was not statistically significant. Scaled probability bar may be more accurate than probability wheel. However, the difference was again not quite statistically significant at $p = 0.05$ level. Considering time taken in elicitation processes, we can order the three methods according to their speed: probability bar, probability wheel, and direct numerical elicitation.

An interesting effect is evident in Table 3.4. When the value of the prior parameters was 5 or 10, the MSE for all techniques is lower than when the α's are set to 10^{-4}. In fact, when the α parameters were set to very small values, it was observed that the probabilities elicited from the experts were closer to the *true* model than they were to the *expected* models calculated with the α parameters. I believe that the reason for this discrepancy is the following. The subjects will naturally have a small but substantial prior belief of uniformity in the parameters, which may act like an anchor in the elicitation. For example, if a subject were given a loaded coin with the instruction to estimate the probability of the coin coming up heads, he or she is likely to require at least a few (5 or 10) flips of the coin before concluding that the coin is weighted one way or the other.

When the α parameters are set to 5 and 10, the elicited models are closer to the expected models than they are to the original model. Furthermore, the results were observed to be statistically significant; whereas with $\alpha = 10^{-4}$ the results were not significant. Another way of looking at this result is that if the user explored one configuration of a node's parents only a few times, then the small-α parameter model would produce very extreme, non-smooth probability distributions under certain parent configuration. For example if the user explored one configuration just 1 time, then the low-α model would produce a probability distribution with the one visited state having probability ≈ 1; whereas, a sensible user would not predict such an extreme distribution, but would rather assume that the probability was still roughly uniformly distributed.

This observation may be related to the well-known finding that people tend to overestimate very low probabilities [47]. In fact, what may be happening in the case of low probability events is that the person's assumption of weak prior uniformity is smoothing the distribution, producing "erroneous" estimates. This fact may suggest a means of correcting for low-probability event estimates by first subtracting out the small uniform distribution from the assessed distribution.

One objection that could be raised to the technique is that a thirty-minute training session is not sufficient for the subjects to achieve expert status. This would be a key objection if we were comparing the elicited models to the *original* model underlying the toy-world; however, the main point in using the trainees' actual acquired knowledge is to deflect this criticism: we are comparing the elicited model precisely to the knowledge that we know our trainee has observed. In principle, this technique should work regardless of the expertise of the trainee. Nonetheless, I acknowledge that there may be some transition during the process of achieving true expertise that alters the trainees' elicitation behavior. I assume that these effects will affect the elicitation techniques in a uniform way, so that the relative assessment of elicitation techniques is not affected.

It may be that the effectiveness of different elicitation techniques varies from expert to expert. In that case, the evaluation technique can provide a relatively quick and effective way to judge which elicitation procedure is most effective for a given expert. The expert can be quickly trained on a toy model, and then the experimental procedure can be used to decide which elicitation technique is most effective for that particular expert.

3.6 CONCLUDING REMARKS

I presented a method that allows for an objective evaluation of the elicitation methods for probability distributions and the structure of probabilistic models. The method is based on machine learning the expert's beliefs when data of the expert's learning knowledge is available. I illustrated the evaluation approach with a toy virtual world and evaluated three elicitation methods for probabilities: direct numerical elicitation, the probability wheel, and the scaled probability bar. Based on the results of the experiment, I concluded that the probability wheel and the scaled probability bar both performed better than direct numerical elicitation, which supports the proposition that graphical tools are useful in eliciting experts' beliefs. The scaled probability bar was the most efficient in terms of being most accurate and taking the least time. My conclusion supports the proposition that graphical tools are useful in eliciting experts' beliefs. The technique can also be used to assess the best technique for a given expert, by training the expert in the toy-world and discovering his or her most effective elicitation technique.

The evaluation method I presented in this chapter can be applied to other elicitation schemes, both for probability elicitation and for structure elicitation. I envisage the use of the method as selecting a proper elicitation scheme for experts in real applications to build Bayesian network models through knowledge acquisition from experts. The first step is to discover the best elicitation method for a particular domain expert through experiment with the toy world. Then the expert can use the best suitable method for him to elicit his knowledge for building probabilistic models.

A side effect of the experiments was to shed light on the well-known phenomenon that people tend to overestimate small probabilities, and to possibly suggest a means to correct for this effect.

4.0 EVALUATING SENSITIVITY OF BAYESIAN NETWORKS

4.1 INTRODUCTION

To validate a Bayesian network model, it involves testing the reasoning accuracy based on the model, evaluating the model's performance robustness, and evaluating the model's tolerance to noises etc. The performance robustness and noise tolerance are closely related and often investigated using sensitivity analysis. In Bayesian networks, sensitivity analysis is studying the effect of small changes of the numeric parameters on a Bayesian network's performance.

The common belief, to a great degree based on a series of experiments in [56], is that Bayesian networks are, on the average, insensitive to inaccuracies in the numeric value of their probabilities. Henrion et al. [34] further elaborated one of the experiments and explored the possible explanations of the low sensitivity. In [34], the conclusions were drawn based on the average of the probabilities of the true diagnosis with simulated scenario cases run by imparting random noise on the nominal probabilities of known networks at increasing levels of uncertainty. The reported results differentiated between true-positive diagnosis cases and true-negatives, and between the effect of noise on the priors of conditional probabilities, leak probabilities (the probabilities in noisy-gate), and prior probabilities.

In this chapter, I argue that differences in sensitivities found in [34] between true-positive and true-negative results may not be valid because the log odds-normal distribution, which was used to generate random noise on probabilities, used an invalid range of standard deviations where the differences were observed. The presence of true-positive and true-negative biases in Bayesian network diagnosis results from misevaluation of the network by experts, and should be corrected once those biases are detected. Differences in network sensitivity due to noise on the different types of probabilities is a quantifiable random effect that depends on the distribution used to model the added noise and possibly, on the topology of the network.

I also show that comparing the average results of the simulated posterior probabilities to the nominal posterior probabilities may not be the most indicative measure of network sensitivity because information about the effect of the noise distribution variance is lost, especially when the distribution is very asymmetric as is the case at nominal values close to zero or one. It is in the variation of these posterior probabilities that imprecision in parameters is reflected. Although the difference in computed posteriors derived from noisy versus nominal probabilities is indicative of the sensitivity of the network, the partial ordering of the posterior probabilities is argued to be a more critical indicator of the outcome of the diagnosis. It is proposed then to assess the sensitivity of the network based on the effect that the uncertainty in probabilities has on the partial ordering of the probable causes, measured using a suitable lower confidence bound.

A series of experiments were designed to investigate the sensitivity of three Bayesian networks built for diagnosis of airplane systems, to inaccuracy in different type of probabilities: prior probabilities, conditional probabilities, and leak probabilities. I varied the probability parameters in the networks by introducing log-odds normal noise for the following range of standard deviations: 0.1, 0.25, 0.5, 0.8, and 1, respectively. The criterion I used to measure the sensitivity of the networks is a set of lower confidence bounds measured by percentiles: e.g., 50%, 80%, 90%, 95%, and 99%. The results showed that generally, increasing noise level to the probabilities produced higher sensitivities in the tested networks. The results also suggested that prior probabilities turned out to be more influential parameters to diagnosis in the tested networks, compared with conditional probabilities and leak probabilities. In contrast to the common belief that Bayesian networks are generally insensitive to inaccurate probabilities, the results showed that some networks can show significant sensitivity to inaccuracy in probabilities even with a small variance in the noise distribution. The results agree with recent findings of high sensitivities reported by [15, 52] in an empirical study using a Bayesian network from medical diagnosis/prognosis and treatment planning.

The chapter is organized as follows: Section 4.2 elaborates my arguments related to the empirical approach to sensitivity analysis. Section A.2 describes the sensitivity experiments conducted on three large production networks built for diagnosis of airplane systems. In Section A.5, I give a brief conclusion about the results of my experiments.

4.2 LOG-ODDS NOISE AND MEASURES OF SENSITIVITY IN BAYESIAN NETWORK

The two main points are: a) the log-odds normal distribution, although it has appealing properties for modelling the noise of probabilities, it may not be valid for assessing network sensitivity when the values of standard deviations are greater than one, and b) the use of averages for comparing the posterior probabilities, derived from noisy probabilities, to the nominal posteriors may hide the effect of the variance of the noise distribution, especially for probability values near zero or one. I will deal with each point separately.

4.2.1 Validity Of Log-odds Normal Distribution

One simple way to simulate the small changes of a probability is to sample the probability value around the nominal value, or equivalently, add random noise directly to the probability. This approach, however, has some problems. First, a large additive error is likely to produce a probability greater than 1 or less than 0, so need to be truncated. Same thing with the sampling method, the sampling distribution may not be a normal distribution with mean at the nominal value because of the truncation effect, especially when the nominal value is close to 0 or 1. Second, the probability change by plus/minus a certain value, e.g., 0.1, may not be significant to a nominal probability 0.9 but much more serious to a nominal probability 0.001.

A more appealing approach to avoid these problems should have a symmetric effect on the two bounds of probability: 0 and 1. It also needs to use a relative change of the nominal probability value when simulating the small changes of the probability.

The log-odds normal transformation provides exactly this behavior. It first transforms a probability p into log-odds form $log\ \frac{p}{1-p}$, then it adds normal noise $\epsilon \sim N(0, \sigma)$ to the log-odds, and finally transforms the quantity back into probabilities $p' = \frac{1}{1+(p^{-1}-1)\ 10^{-\epsilon}}$. It is a suitable model for noise imposed on probabilities. One reason is that the sampled probability still remains in the [0,1] range. Another reason is that the transformation recognizes differences imparted by noise to probabilities near 0 or 1 versus those in the middle of the range near 0.5 [34].

However, this distribution may not be valid for standard deviations greater than one for the purpose of assessing network sensitivity to probability noise. Equation 4.1 illustrates the probability density of the log-odds normal distribution:

$$Y = log\ \frac{p}{1-p} + \epsilon\ , \tag{4.1}$$

where $\epsilon \sim N(0, \sigma)$, transitions from unimodal to bimodal for values of $\sigma > 1$. A simplified equation for the distribution of the nominal probabilities with added noise, p', in terms of the nominal p and noise ϵ is

$$p' = \frac{1}{1 + (p^{-1} - 1)\ 10^{-\epsilon}}\ . \tag{4.2}$$

The relation between the noisy odds and the nominal odds is then:

$$\frac{p'}{1-p'} = \frac{p}{1-p}\ 10^{\epsilon}\ , \tag{4.3}$$

or $odds' = odd * 10^{\epsilon}$, where $odds' = \frac{p'}{1-p'}$ and $odds = \frac{p}{1-p}$.

This indicates that error introduced by the log-odds normal noise ϵ reflects the scale of the change of odds by a factor of 10^{ϵ}, which is so substantial that p' density function becomes bimodal with peaks at 0 and 1 when $\sigma > 1$ as shown in Figure 4.1 for probability $p = 0.8$. In other words, when $\sigma > 1$, the noise is so extreme and the imparted probability is so skewed that it is no longer close to the nominal probability, rather, it distributes densely at 0 and 1. The log-odds normal distribution guarantees only that the median probability is equal to the nominal probability. The mean probability may be very different.

Table 4.1 shows values of p' computed from Equation 4.2 for various values of p and ϵ. Note that the values of ϵ correspond to values of σ in the standard normal distribution. For values of $\epsilon > 1$, the difference between the noisy and nominal probabilities increases rapidly

Table 4.1: Values of the noisy p' computed from Equation 4.2 for different values of the nominal p and noise ϵ.

Values	**Values of p and percentage of $(p'-p)/p$**									
of ϵ	0.0001	%	0.01	%	0.1	%	0.25	%	0.5	%
0.1	0.00013	26	0.013	26	0.12	23	0.30	18	0.56	11
0.3	0.00020	100	0.020	98	0.18	81	0.40	60	0.67	33
0.5	0.00032	216	0.031	210	0.26	160	0.51	105	0.76	52
0.7	0.00050	401	0.048	382	0.36	258	0.63	150	0.83	67
1	0.00010	899	0.092	817	0.53	426	0.77	208	0.91	82
3	0.09092	90817	0.91	8999	0.99	891	1.00	299	1.00	100
5	0.90910	908999	1.00	9890	1.00	900	1.00	300	1.00	100

for values of the nominal probability that are close to zero. The effect is also large but less pronounced for values of nominal probabilities in the mid-range towards p = 0.5.

The log-odds normal distribution is an adequate model of noise added to probabilities for values of $\sigma < 1$, where the distribution is unimodal. For values of $\sigma > 1$ the distribution becomes bimodal as shown in Figure 4.1. Using the distribution in that range to describe the noise on the priors is equivalent to considering an expert who assesses a prior probability, known to be near zero, and erring in judgment by such margin that the true prior probability may, in fact, be close to one! This is what the log-odds normal distribution implies for large values of σ.

4.2.2 Measures For Assessing Bayesian Network Sensitivity

In using Bayesian networks for diagnosis the partial ordering of probable causes resulting from the update of the posterior probabilities given a set of findings constitutes the diagnosis.

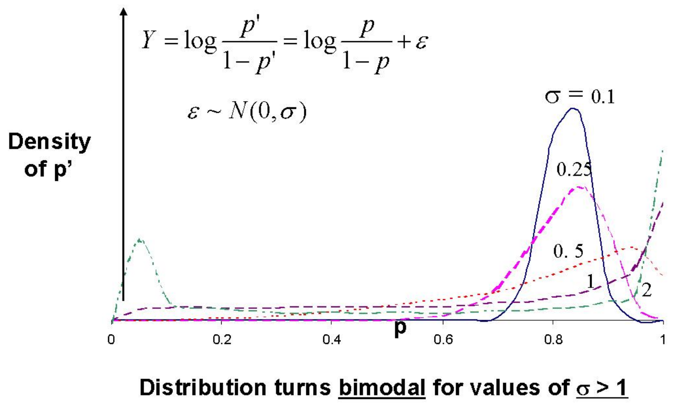


Figure 4.1: The log-odds normal distribution centered around the nominal prior of 0.8, for various values of σ

While the most probable cause is often given the highest consideration, typically, in multiple fault-diagnostic systems it is a particular set of the top causes (e.g., the top five) and their partial ordering that is most informative. Since very seldom the diagnosis singles out a particular cause, the partial ordering provides guidance for subsequent actions. The effect that noise has on the posterior partial ordering of the causes is, therefore, a significant measure of the network sensitivity.

Table 4.2 shows the top five suspect parts selected from a Bayesian network diagnosis system representing a particular test case scenario of an airplane fault. It compares posterior probability from the nominal network with the average posteriors from one hundred noisy networks. The noise distribution used was the log-odds normal with $\sigma = 0.5$. Note that the deviation of the average posterior from the nominal does not appear to be substantial.

The top of Table 4.3 shows the average change in rank order for the five suspect parts of Table 2. This average reflects the average absolute-value change in rank for each part from its nominal rank that is due to noise added to the network probabilities. The average change in rank shows that noise with $\sigma = 0.5$ is expected on the average to affect a change

Table 4.2: The nominal posteriors of the top five suspect parts from an airplane diagnosis compared to the average from one hundred noisy posteriors (log-odds normal, $\sigma = 0.5$).

	Part 1	**Part 2**	**Part 3**	**Part 4**	**Part 5**
Nominal posterior	0.40	0.29	0.11	0.07	0.07
Average posterior	0.35	0.28	0.15	0.07	0.07
Standard Deviation	0.24	0.23	0.15	0.07	0.09

Table 4.3: Lower confidence bounds and average changes of the ranks for the five most probable causes.

	Part 1	**Part 2**	**Part 3**	**Part 4**	**Part 5**
Average rank change	1.02	1.05	1.26	1.21	1.15
Standard deviation	1.19	0.93	0.90	0.83	1.06
50th percentile	1	1	1	1	1
80th percentile	2	2	2	2	2
90th percentile	3	2	2	2	2.9
95th percentile	3	3	3	2	3
99th percentile	4	3.99	3	3	4

in the rank of the top five suspect parts by approximately one ranking order. By itself, this is not a bad result considering the size of the variance of the distribution used. However, it is somewhat misleading.

Typically, for airplane diagnosis, the reliability estimates of most airplane parts is of order greater than 10^5 hours for the mean time between part failures. The corresponding prior probabilities are therefore approximately of order smaller than 10^{-5}. At such low probabilities the log-odds normal distribution is very asymmetric and the average rank does not adequately represent the effect that the noise imparts on the network. Shown at the bottom of Table 3 are lower-bound confidence estimates for confidence levels from 50% to 99%. The data show that for noise with $\sigma = 0.5$, there is a 50% chance that the ranking order of the parts could change by at least one position. For the most probable suspect part (i.e., Part 1), there is a 20% chance that it could drop by more than two ranks, a 10% chance that it may drop by more than three orders in rank, and a 1% chance that it may drop by more than four. For networks with high sensitivity to noise, the nominal diagnosis could advise the airplane maintainer to unleash a series of irrelevant actions that could result in unnecessary and costly delays and cancellations.

This analysis, we believe, is more representative of the sensitivity of the network due to noise in the network probabilities. The remainder of the paper will present data compiled from several airplane diagnosis networks under various test scenarios, and will distinguish the network sensitivity to noise contributions from prior probabilities, conditional probabilities, and leak probabilities.

4.3 SENSITIVITY EXPERIMENT

4.3.1 Measure Of Diagnostic Performance

As indicated in Section 4.2.2, average posterior probabilities may not be an adequate measure to assess sensitivity of Bayesian networks with respect to diagnosis, especially when the probability distribution is extremely skewed by adding in the log-odds normal noise. Instead, lower confidence bounds on rank changes of the diagnosis recommended by a partial-ordered list of suspect parts, better reflect the effect that random noise has on the network.

In the experiments, I use lower confidence bounds for 0.50, 0.80, 0.90, 0.95, 0.99 percentiles of the diagnosis ranks over test cases to quantify diagnostic performance. Average and standard deviation of rank changes are also calculated for comparison.

4.3.2 Networks And Test Cases

I used three large networks built for diagnosing three major airplane systems. A number of test case scenarios were defined for each network. These scenarios represented real-life cases encountered during routine airplane maintenance procedures. Each test case constitutes a set of findings, used as inputs to the networks, which do not necessarily isolate the failed parts with certainty, but rather generate a ranked list of the most likely suspect parts. The ranked list of parts constitutes the diagnosis given a particular test case scenario. For illustration purposes and without loss of generality I denote the three networks as Net 1, Net 2, and Net 3. The airplane parts are also denoted by numbers associated with their posterior ranking order, i.e., Part 1, Part 2, etc.

4.3.3 Experimental Design

I tested with three networks built for diagnosing airplane part failure. For each network, I first classified the nodes into different sets according to their probability types: prior, conditional and leak. To generate a noisy network, I added noise to each set of nodes independently for a given level of noise and scenarios. Each scenario was run one hundred times with the same noise distribution for each set of nodes. A noisy network was generated in each run. The total number of networks used in our experiment were 34503, consisted of 3 types of probability * 5 levels of noise * 100 runs * (3+5+15) scenarios, plus 3 original networks without noise.

The test began with a diagnosis on the nominal network given the findings defined in the scenario. For this network, the nominal partial ordering of the recommended failed parts was generated. The rank of each probable failed-part was recorded according to the partial ordering. Under the same situation, (i.e., the same set of nodes, the same noise distribution, and the same scenario), the noisy networks were used to compute the noisy rank changes of the diagnosed failed-parts from the rank changes computed with the nominal network. The effect of noise was assessed by computing statistics on the rank changes, such as average and standard deviation of rank changes, and 0.50, 0.80, 0.90, 0.95, 0.99 percentiles of lower confidence bounds.

4.3.4 Results

Figure 4.2 plots the average rank changes over one hundred cases across different scenarios of the most probable failed parts in Net 3 affected by five levels of prior noise. As expected, performance degrades as the noise increases. Note that the rank of the most probable failed-part drops, on the average, about one position when noise is distributed with $\sigma(or std) = 0.1$, and it drops about two positions when noise is distributed with $\sigma = 1.0$.

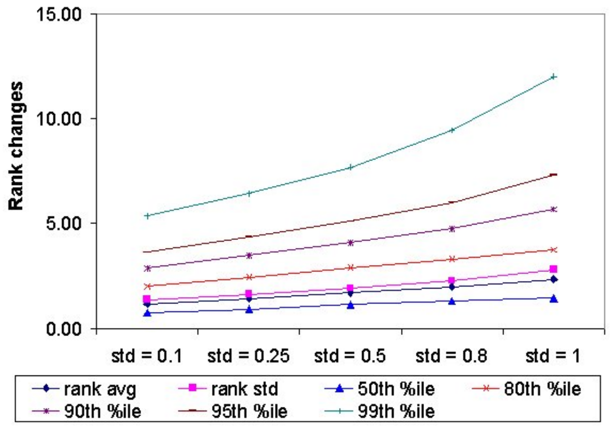


Figure 4.2: Rank changes of the most probable failed parts in Net 3 based on 100 run cases across different scenarios and prior noise.

Since with $\sigma = 1.0$ the most probable failed part will change on the average almost three rank positions, it may look as if the diagnosis performance is robust and insensitive to the imprecise prior probabilities. However, looking at the lower confidence bounds, Figure 2 indicates that there is a 90% chance that the most probable failed part will stay within the top five rank positions for noise with $\sigma < 0.5$. Conversely, with $\sigma >= 0.5$, there is a 90% chance that the most probable failed part will disappear from the top five recommended parts given by the diagnosis, which could possibly result in incorrect diagnosis by the network.

Figure 4.3 illustrates the rank changes of the top five most probable failed parts in Net 3 when the prior noise is distributed with $\sigma = 1.0$. From the chart, we see that 0.50 percentile lower bound for the five parts are smaller than rank average, further indicating the asymmetry of the noise distribution.

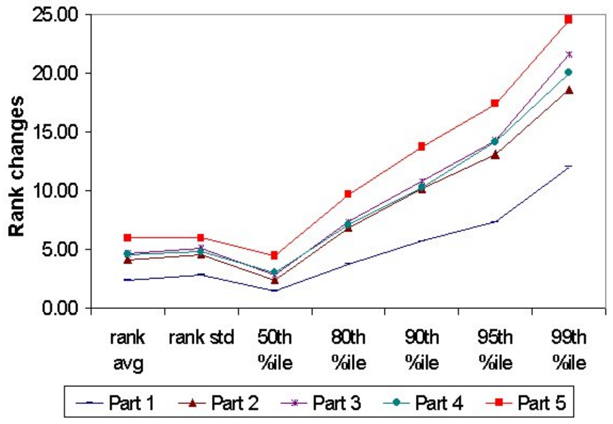


Figure 4.3: Rank changes of the top five most probable failed parts in Net 3 based on 100 run cases across different scenarios and prior noise $\epsilon \sim N(0, 1.0)$.

Also note the high standard deviations of the rank. This illustrates that the sensitivity of the noisy networks varies greatly with different scenarios. The noisy network may be pretty robust for some of the observations, but may be quite sensitive to others. Therefore, different scenarios play an important role in testing sensitivity of Bayesian networks.

The effect of noise on conditional probabilities and on leak probabilities is much smaller than that on prior probabilities for all of the three networks in the experiments. As shown in Figure 4.4, the average rank changes are smaller than 1 even when the conditional noise is distributed with $\sigma = 1.0$. The 0.99 percentile lower bounds are all smaller than 4. Therefore, in 99 percent of the time, the top five most probable failed-parts would stay in the top positions in the partial ordering given by the diagnosis.

As was the case with noise added to prior probabilities, the same trend is observed

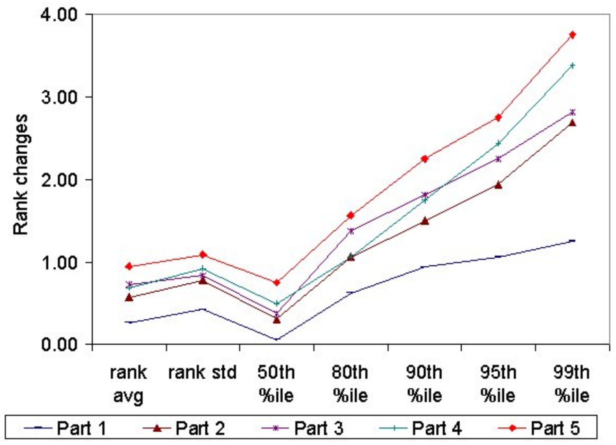


Figure 4.4: Rank changes of the top five most probable failed parts in Net 3 based on 100 run cases across different scenarios with CPT noise $\epsilon \sim N(0, 1.0)$.

with conditional probabilities when noise level increasing. When the noise added to the conditional probability tables becomes higher, the network becomes more sensitive, as a result, the diagnosis capability of the networks degrades.

Figure 4.5 shows the rank changes of the most probable failed part in Net 1 based on one hundred run cases across different scenarios and different prior noise distributions. The rank changes in Net 1 are much smaller than the rank changes found in Net 3, which indicates that different networks may have a different degree of sensitivity to imprecise probabilities. However, the rank changes in Net 2 were close to those of Net 1.

4.4 CONCLUSION

I argue in this chapter that the log-odds normal distribution is valid as a model for sensitivity analysis only in the range of standard deviations where the distribution is unimodal. I also shows that using average posterior probabilities as criterion to measure the sensitivity may not be the most indicative, especially when the distribution is very asymmetric as is

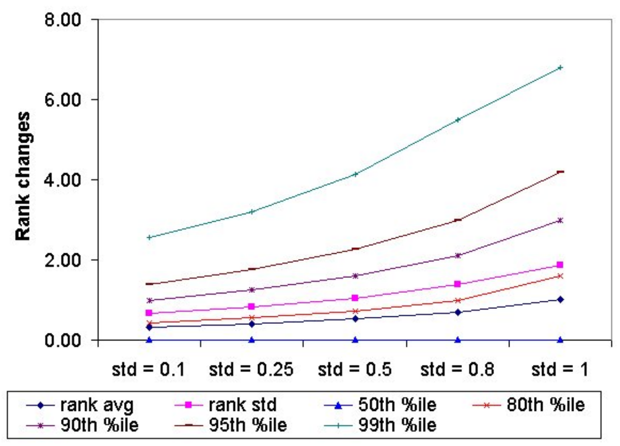


Figure 4.5: Rank changes of the most probable failed part in Net 1 based on 100 runs cases across different scenarios and prior noise.

the case at nominal values close to zero or one. It is proposed, instead, to use the partial ordering of the most probable causes of diagnosis, measured by a suitable lower confidence bound on the change in the rank order. Preliminary results of the sensitivity analysis experiments were shown with three Bayesian networks built for diagnosis of airplane systems. The results showed that some networks are more sensitive to imprecision in probabilities than previously believed, and the networks have different sensitivity to different sets of variables. In the three networks for airplane diagnosis used for the test, the prior probabilities for the parentless nodes are the most sensitive, compared to the conditional probabilities and the leak probabilities for the noisy gate.

5.0 SENSITIVITY FOR MODEL SELECTION IN LEARNING BAYESIAN NETWORKS?

5.1 INTRODUCTION

In Chapter 4, we see that Bayesian networks can have different sensitivities to variations in probability parameters. Similar results were found that some networks can be very sensitive to small changes of their probability parameters [15, 52]. Indeed, parameter estimates obtained from an expert or from a relatively scarce data can have quite high variance. In addition, the networks can be also sensitive to a small change of graph structure such as adding, deleting or reversing an edge. As a consequence, the results of inference obtained by using a particular Bayesian network are not guaranteed to be stable with respect to small changes in the model.

Therefore, it may be desirable to obtain low sensitive, i.e., highly robust, Bayesian network models among other candidates that well represent the domain knowledge. This is a typical model selection problem in learning Bayesian networks. In learning Bayesian networks, Model selection refers to finding a graph structure which best describes the dependence relationships between the domain variables represented by the data. One approach to model selection is using scoring metrics to guide a heuristic search in the space of possible graph structures [14, 29]. The existing scoring functions such as BIC [63], BDe [30] and MDL [44], attempt to maximize likelihood of a model while minimizing its representation complexity (e.g., the number of parameters). An alternative approach is a constraint-based learning [71, 8], that searches for network structures satisfying a set of independence assumptions (constraints) obtained via statistical conditional-independence tests or from some prior domain knowledge. Recently, the two approaches were shown being equivalent under certain conditions, such as using same variable ordering and applying cross-entropy methods for testing conditional independence [17].

But the two approaches do not differentiate between *Markov-equivalent graphs* [10], i.e. graphs that yield same set of independence assumptions but differ in the directionality of certain edges. Interestingly, as I will show later in this chapter, Markov-equivalent Bayesian networks can have different *sensitivities* to the parameter changes. Therefore, two models with same scores (i.e., both fitting data well and both having relatively low representation complexity), can have different sensitivity to small changes in certain parameters. In this case, a less sensitive network may be preferred in model selection.

However, depending on the measure of sensitivity for Bayesian networks, sensitivity may not be an appropriate criterion for model selection. As I will show in this chapter, using maximum value of parameter sensitivity as a measure of sensitivity for Bayesian networks, the relationship between the sensitivities of equivalent networks reduces to a single, equivalent, local probability distribution after one edge reversal. Therefore, selecting a low sensitive model is essentially same as selecting a model with low probability values in the local distributions for the nodes involving in the edge reversal. As shown by Renooij and van der Gaag [61], a uniform probability distribution has the lowest sensitivity bounds. Selecting a low sensitive model using such a sensitivity measure then prefers uniform distributions, which is often interpreted as a total random process with the highest uncertainty. This violates the principles of Bayesian network modelling, which encodes the expert knowledge at the best effort and is supposed to prefer certain level of certainties in the models. But selecting the low sensitive model leads to the contrary choice, if the sensitivity measure is based on the maximum value of parameter sensitivities.

The chapter starts with a brief review of the typical approaches to model selection in learning Bayesian networks. Then it describes Markov equivalent network structures with transformational characterization of equivalent structures. After introducing a definition of sensitivity for Bayesian networks, the chapter gives a proof that Markov equivalent networks have different sensitivities, and their sensitivities can be expressed by a simple linear function. The establishment of this relationship leads us to a conclusion that sensitivity, measured as the maximum value of parameter sensitivities, may not be an appropriate criterion for model selection.

5.2 LEARNING BAYESIAN NETWORK STRUCTURE

Learning the structure of a Bayesian network is an *unsupervised learning* problem which can be informally stated as following: given a set of observations $D = \{\mathbf{d^1}, \ldots, \mathbf{d^N}\}$ (training data), find a network structure G that best matches D. Learning the network structure is referred to as model selection, which is often viewed as an optimization problem with respect to a particular *scoring function* defined on a Bayesian network. Then learning Bayesian network structure is a greedy search process which always selects the structure that has the highest score.

5.2.1 Bayesian approach

Following the Bayesian approach, to find a good network structure which encodes the physical joint probability distributions Θ for multivariate $\mathbf{X}$ is to select a network structure that has highest posterior probability given data set D. This posterior probability, $p(G|D)$, can be computed by Bayes' rule:

$$p(G|D) = p(G)p(D|G)/p(D).$$

In this equation, $p(D)$ is a normalization constant that does not depend on network structure. For the sake of convenience, we can assume that all structures are equally likely. Then, to determine $p(G|D)$, one must compute the marginal likelihood $p(D|G)$ for each of the possible structures. For a complete data set with assumption of *parameter independence*[29], and Dirichlet prior distribution, the likelihood can be computed as in the following equation:

$$p(D|G) = \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{\Gamma(\alpha_{ij})}{\Gamma(\alpha_{ij} + N_{ij})} \cdot \prod_{k=1}^{r_i} \frac{\Gamma(\alpha_{ijk} + N_{ijk})}{\Gamma(\alpha_{ijk})} \ , \tag{5.1}$$

where $\alpha_{ij} = \sum_{k=1}^{r_i} \alpha_{ijk}$, $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$, and $\Gamma(\cdot)$ is the *Gamma-function* which satisfies $\Gamma(x+1) = x\Gamma(x)$ and $\Gamma(1) = 1$.

Thus, the above equation provides a scoring metric that is proportional to the posterior probability of G given data. It is often referred to as BDe metric ("B" for Bayesian, "D" for Dirichlet, and "e" for likelihood equivalence). Note that a higher prior probability P(G) to simpler models has the effect of penalizing complex models. Generally, when likelihood equivalence does not hold, BD metric is used which combines $P(G)$ and $P(D|G)$ as a product.

Besides the*Bayesian score*, other commonly used scoring functions include the *Minimum Description Length (MDL)* criterion [62, 44], and the equivalent to it, *Bayesian Information Criterion (BIC)* [63] (often called BIC/MDL). Other closely related information-theoretic scoring functions include *Akaike Information Criterion (AIC)* and *Minimum Message Length (MML)*. These scores provides a tradeoff between the accuracy and the complexity of a model by minimizing the sum of the encoding length of the model (i.e. its representation complexity) and the encoding length of the data given the model, which is measured by the negative log-likelihood and thus reflects the model error. For example, the MDL criterion [62, 44] is

$$MDL(G|D) = -logP(D|G) + \frac{logN}{2}|\Theta| \ , \tag{5.2}$$

where $|\Theta|$ is the dimension of the model, e.g., the number of independent parameters in the model, and N is the number of data cases. The first term is the negative log-likelihood of the graph structure G given data D (which equals the number of bits needed to describe D when using structure G), while the second term is the model complexity given as the number of bits required to encode the network parameters. Thus, minimizing the MDL criterion provides the shortest description of the training data (i.e. learning can be viewed as data-compression), which at the same time favors models that predict data better (have higher log-likelihood $logP(D|G)$) and have lower representation complexity ($\frac{logN}{2}|\Theta|$).

5.2.2 Constraint-based Approach

An alternative approach to learning graph structure is constraint-based approach [71, 8]. In constraint-based learning, training data are viewed as reflection of dependence relationships between variables. Statistical tests, such as chi-squared or mutual information, are applied to identify the dependencies directly. Recently, efficient algorithms were developed for conditional independence test using a information-theory based approach [9]. The independence relations inferred from data are used to constrain the search space for possible graph structures. Consequently, the Bayesian network structure learned this way is a partially directed acyclic graph (PDAG) which represents a class of Markov-equivalent networks. Since the directed edges often have causal semantics, constraint-based learning is closely related to learning causal relationships. See [55] for an in-depth discussion on this topic.

5.2.3 Equivalence Characteristics

Interestingly, the above scoring metrics that are popularly used in learning Bayesian networks can not differentiate Markov equivalent graphs [10]. In other words, Markov equivalent graphs are score equivalent under the criteria of AIC, BIC, BDe, ML, MDL etc. Recently, the constraint-based approach is shown being equivalent to the scoring-based approach under certain conditions, such as using same variable ordering and applying cross-entropy methods for testing conditional independence [17].

5.3 MARKOV EQUIVALENCE

A Bayesian network for a set of variables $\boldsymbol{X} = \{X_1, X_2, ..., X_n\}$ represents a joint probability distribution over these variables. Structurally a Bayesian network is an acyclic directed graph (*DAG* in short). The structure of a Bayesian network defines the independence relationships between the variables given by the *Markov condition*. That is, any node in the graph is conditionally independent of its nondescendants given its parents.

Two DAGs are Markov equivalent if and only if they contain the same set of variables and they represent the same conditional independence relationships on those variables. In other words, two network structures G and G' are equivalent if the set of distributions that can be represented using G is identical to the set of distributions that can be represented using G'. We use $G \approx G'$ to denote the equivalence relationship between G and G'.

Formally, two DAGs G and G' are equivalent if for every Bayesian network $B = \{G, \Theta\}$, there exists a Bayesian network $B' = \{G', \Theta'\}$ such that B and B' define the same probability distributions, and vice versa.

When two Bayesian networks have equivalent structures and they represent the same joint probability distribution over the same domain of variables, we say that the two networks are Markov equivalent.

Definition 1 (Markov Equivalence)**.** *Two Bayesian networks $B = \{G, \Theta\}$ and $B' = \{G', \Theta'\}$ are Markov equivalent if $G \approx G'$, and Θ and Θ' represent the same joint probability distributions $p(X_1, X_2, ..., X_n)$.*

Equivalent graphs share the same *skeleton*, i.e., the undirected graph resulted from ignoring the directionality of edges in the graph. Also, equivalent graphs share the same *v-structures*. A v-structure in DAG G is an ordered triple of nodes (X, Y, Z) such that G contains the arcs $X \rightarrow Y$ and $Z \rightarrow Y$ but X and Z are not adjacent in G. As derived by Verma and Pearl [75], sharing the same skeleton and the same v-structures is both necessary and sufficient condition for two DAGs to be equivalent.

One interesting characterization of equivalent graphs is that a DAG G can be transformed to its equivalent graph G' by a sequence of edge reversals [10]. Only the covered edges are necessary to involve in the reversal for the transformation. An edge $X \rightarrow Y$ is covered in G if node X and node Y have identical parents in graph G, with the exception that X is not a parent of itself. Formally, a covered edge is defined as below.

Definition 2 (Covered Edge)**.** *[10] An edge $X \rightarrow Y$ is covered in G if $\boldsymbol{Pa_Y} = \boldsymbol{Pa_X} \bigcup X$*

Clearly, any property that holds over all DAGs in an equivalent class must hold over every pair of DAGs in that class which differ by the orientation of a single covered edge. Chickering [10] proved the converse by showing that for any pair of equivalent DAGs G and G', we can transform G into G' by a series of covered edge reversals, where each reversed edge has different orientation in G and G'.

Theorem 3. *[10] Let G and G' be any pair of DAGs such that $G \approx G'$, and let m be the number of edges that have opposite orientation in G and G'. There exists a sequence of m distinct edge reversals in G with the following properties:*

1. *Each edge reversed in G is a covered edge;*

2. After each reversal, G is a DAG and $G \approx G'$;

3. After all reversals, $G = G'$.

For example, we know that the DAG $G^1 = X \to Y \to Z$ is equivalent to the DAG $G^2 = X \leftarrow Y \leftarrow Z$. To transform from G^1 to G^2, we can first reverse the covered edge $X \to Y$. What we get after the reversal is an equivalent DAG $X \leftarrow Y \to Z$. Secondly we reverse the covered edge $Y \to Z$. The result of the two subsequent edge reversals is the equivalent graph G^2.

Based on Theorem 3, Chickering [10] derived a simple local characterization of equivalent network structures. He showed that a property holds for all pairs of equivalent networks that differ by a single covered edge orientation if and only if that property holds for all networks in the equivalence class. This is very useful to prove a given property is invariant (or variant) over all equivalent structures. Simply showing that the property is invariant (or variant) to any reversal of a single covered edge is sufficient.

Using this technique, Chickering [10] proved that equivalent graphs are score equivalent in terms of the scoring metrics such as likelihood, BIC, MDL, and BDe. I will use the same technique to prove that the Markov equivalent networks are not necessarily sensitivity equivalent.

5.4 MARKOV EQUIVALENCE DOES NOT IMPLY SENSITIVITY EQUIVALENCE

5.4.1 Network Sensitivity Measures

Let $\mathbf{X} = \{X_1, X_2, \ldots, X_n\}$ be a set of random variables modeled in a Bayesian network B. Let r_i be the number of possible values $x_i^1, \ldots, x_i^{r_i}$ for variable X_i, and let q_i be the number of $\boldsymbol{Pa_i}$'s jth configuration. As a bit of notation, we define θ_{ijk} to be the probability parameter that $X_i = x_i^k$ given that $\boldsymbol{Pa_i} = Pa_i^j$, where $1 \leq k \leq r_i$, and $1 \leq j \leq q_i$. We use the vector

$$\boldsymbol{\theta_{ij}} = \{\theta_{ijk}\} = (\theta_{ij1}, \ldots, \theta_{ijk}, \ldots, \theta_{ijr_i})$$

to denote the conditional probability distribution of X_i under its jth parent configuration. Since the states of any variable X_i in a Bayesian network are mutually exclusive and collectively exhaustive, we have by the laws of probability that

$$\sum_{k=1}^{r_i} \theta_{ijk} = 1$$

for each i, j.

Therefore, if a parameter θ_{ijk} changes its value to θ'_{ijk}, the remaining parameters $\theta_{ijk'}$ (where $k' \neq k$ with the same i, j) have to co-vary their values accordingly to comply with the laws of probability. A popular method for the co-variation of the relevant parameters is *proportional scaling*, which assigns the values of relevant probabilities a proportion of the probability mass left that corresponds to their original ratios. It was proved that proportional scaling results in the closest probability distribution to the original distribution [7].

Formally, proportional scaling can be defined as

$$\theta'_{ijk'} = \begin{cases} \theta'_{ijk} & if \quad k' = k \\ \theta_{ijk'} \frac{1-\theta'_{ijk}}{1-\theta_{ijk}} & otherwise \end{cases} \tag{5.3}$$

Based on the proportional scaling assumption, as defined in Equation 5.3, Castillo *et al.* [5] identified some important properties of Bayesian networks for efficient sensitivity analysis. The following theorems review these properties. Readers who are interested in the proofs of these theorems can refer to the papers by Castillo *et al.* [5] or Kjærulff and van der Gaag [42].

Theorem 4. *[5] Let B be a Bayesian network, x be a probability parameter θ_{ijk}, y be a query on posterior probability of the target variable X_t with interest in $X_t = x_t$, and e be evidence entered into B. The posterior probability $y = p(x_t|e)(x)$ is a fraction of two linear functions of x.*

$$p(x_t|e)(x) = \frac{p(x_t, e)(x)}{p(e)(x)} = \frac{\alpha x + \beta}{\gamma x + \delta} \, .$$

As presented by Laskey [45], the partial derivative $\partial y/\partial x$ measures the sensitivity of a single query y to the probability parameter x given evidence e. The sensitivity value is a first-order approximation to the changing behavior of the posterior probability y due to small changes in x under the evidence scenario e. This definition of sensitivity is very useful in model validation for building Bayesian networks, where the probability parameters are tuned to make the beliefs more in line with expert's estimate.

The partial derivative $\partial y/\partial x$ is the most popular measure of the sensitivity for a probability parameter x with regard to query y given evidence scenario e. It is widely used in recent research on sensitivity analysis of Bayesian networks [45, 5, 42, 7], and in the corresponding software packages too. Examples include SamIam which is developed in the University of California at Los Angels (available at http://reasoning.cs.ucla.edu/samiam/), and GeNIe, which is developed in University of Pittsburgh (available at http://www.sis.pitt.edu/ genie/). In this thesis, I refer this sensitivity definition to *parameter sensitivity.*

Definition 5 (Parameter Sensitivity)**.** *Let B, x, y and e be as before. y's sensitivity with regarding to the parameter x given e is a partial derivative:*

$$S(x)_{y|e} = \frac{\partial p(x_t|e)}{\partial x} = \frac{\alpha\delta - \beta\gamma}{(\gamma x + \delta)^2}\ . \tag{5.4}$$

Note that 9i8not all of the variables in a network are of interest for queries. Usually only a subset of the variables are the query target. For example, in a medical diagnostic network, queries are more often posted to disease nodes rather than symptom nodes. When there are multiple query targets $\boldsymbol{T} = \{T_1, T_2, ..., T_n\}$ whose posterior probabilities are of interest, the parameter x may have n different sensitivity values corresponding to $\boldsymbol{T}$ given an evidence e. Similarly, for a set of evidence cases $\boldsymbol{e} = \{e_1, e_2, ..., e_m\}$, x's sensitivity value varies in a range determined by the different evidence scenarios. The maximum value is the worst-case sensitivity for x with regard to $\boldsymbol{T}$ and $\boldsymbol{e}$. Without loss of generality, I use the maximum value of the parameter sensitivities for x with regard to $\boldsymbol{T}$ and $\boldsymbol{e}$ to represent x's parameter sensitivity.

With the definition of parameter sensitivity, we can now define the sensitivity of Bayesian networks. One way to measure the sensitivity of a Bayesian network is to take the maximum value of its parameter sensitivities, which represents the worst case of the performance deterioration with the changes on its probability parameters. Another measure is to take the average value of its parameter sensitivities, which represents the average robustness of the network's performance.

Definition 6 (Maximum Network Sensitivity)**.** *Let B be a Bayesian network and x be a probability parameter. With the evidence cases $\boldsymbol{e} = \{e_1, e2, ..., e_m\}$, and the posterior probability queries posted on target nodes $\boldsymbol{T} = \{T_1, T_2, ..., T_n\}$, B's maximum sensitivity is :*

$$S(B)_{max} = \max_{x, t_i, e_j} S(x)_{p(T_i = t_i | e_j)} = \frac{\partial p(T_i = t_i | e_j)}{\partial x} = \frac{\alpha\delta - \beta\gamma}{(\gamma x + \delta)^2} \; . \tag{5.5}$$

The computation of the parameter sensitivities for a Bayesian network can be done efficiently using the algorithms [42, 7]. Appendix A presents a even faster algorithm I developed based on Kjærulff and van der Gaag's algorithm [42] with the relevance reasoning turned on.

5.4.2 Sensitivity Inequivalence

Theorem 7. *Let $B = (G, \Theta)$ and $B' = (G', \Theta')$ are two Markov equivalent Bayesian networks, and G differs from G' by only one edge orientation. Suppose that the edge is $X_i \rightarrow X_j$ in G. Let $\boldsymbol{Pa_i^G}$ be the parents of X_i in G, then $\boldsymbol{Pa_i^G} \bigcup X_j$ be the parents of X_i in G'. Denote X_i's possible states $x_i^1, ..., x_i^k, ..., x_i^{r_i}$, and X_j's possible states $x_j^1, ..., x_j^w, ..., x_j^{rj}$, and $\boldsymbol{Pa_i}^G$'s possible configurations $Pa_i^1, ..., Pa_i^l, ..., Pa_i^{q_i}$. In addition, denote $p(x_j^w)$ to be the marginal probability that $X_j = x_j^w$.*

Now let θ_{ilk}^B be the probability parameter in B that $X_i = x_i^k$ given $\boldsymbol{Pa_i^G} = Pa_i^l$, and let $\theta_{il^w k}^{B'}$ be the probability parameter in B' that $X_i = x_i^k$ given $\boldsymbol{Pa_i^G} = Pa_i^l$ and $X_j = x_j^w$. Let $p(x_t, e)(x) = \alpha x + \beta$ be the marginal probability of query $X_t = x_t$ with evidence e in term of the parameter $x = \theta_{ilk}^B$. And let α', β' be the coefficients of the corresponding sensitivity function in term of $\theta_{il^w k}^{B'}$. Then

$$\begin{cases} \alpha' = \alpha p(x_j^w) \\ \beta' = \alpha \sum_{u \neq w} \theta_{il^u k}^{B'} p(x_j^u) + \beta \end{cases}$$

Proof. Since in G', X_i's parents are $\boldsymbol{Pa_i^G} \bigcup X_j$, and $\theta^{B'}_{il^wk} = p(x_i^k|x_j^w, Pa_i^l)$. Therefore, $\theta^{B'}_{il^wk}p(x_j^w) = p(x_i^k, x_j^w|Pa_i^l)$. By marginalizing out X_j, we have

$$\sum_{w=1}^{r_j} \theta^{B'}_{il^wk}p(x_j^w) = p(x_i^k|Pa_i^l) = \theta^B_{ilk} \ . \tag{5.6}$$

Take the Equation 5.6 into the sensitivity function $p(x_t, e)(x) = \alpha x + \beta$ for $x = \theta^B_{ilk}$, the probability $p(x_t, e)$ can be expressed as below:

$$\begin{aligned} p(x_t, e)(x) &= \alpha\theta^B_{ilk} + \beta \\ &= \alpha \textstyle\sum_{w=1}^{r_j} \theta^{B'}_{il^wk}p(x_j^w) + \beta \\ &= \alpha p(x_j^w)\theta^{B'}_{il^wk} + \alpha \textstyle\sum_{u\neq w} \theta^{B'}_{il^uk}p(x_j^u) + \beta \end{aligned}$$

According to the definition of sensitivity function, in $p(x_t, e)(x) = \alpha\theta^B_{ilk} + \beta$, $\alpha\theta^B_{ilk}$ is the only term that contains θ^B_{ilk} and β does not contains θ^B_{ilk}. And Substituting θ^B_{ilk} with $\sum_{w=1}^{r_j} \theta^{B'}_{il^wk}p(x_j^w)$ only introduces term $\alpha p(x_j^w)\theta^{B'}_{il^wk}$ that contains $\theta^{B'}_{il^wk}$. Therefore, in the above expression, $\alpha p(x_j^w)\theta^{B'}_{il^wk}$ is the only term that contains $\theta^{B'}_{il^wk}$ and $\alpha \sum_{u\neq w} \theta^{B'}_{il^uk}p(x_j^u) + \beta$ does not contain $\theta^{B'}_{il^wk}$. The sensitivity function can be rewritten as a function of $x = \theta^{B'}_{il^wk}$:

$$p(x_t, e)(x) = \alpha' x + \beta' \ ,$$

where

$$\begin{cases} \alpha' = \alpha p(x_j^w) \\ \beta' = \alpha \sum_{u\neq w} \theta^{B'}_{il^uk}p(x_j^u) + \beta \ . \end{cases}$$

□

Similarly, the sensitivity coefficients γ' and δ' have the relationship with γ and δ as stated in the following lemma.

Lemma 8. *Let θ^B_{ilk} and $\theta^{B'}_{il^wk}$ be as before. Let γ, δ be the coefficients of the sensitivity function for θ^B_{ilk}, and let γ', δ' be the coefficients of the sensitivity function for $\theta^{B'}_{il^wk}$,then,*

$$\begin{cases} \gamma' = \gamma p(x_j^w) \\ \delta' = \gamma \sum_{u\neq w} \theta^{B'}_{il^uk}p(x_j^u) + \delta. \end{cases}$$

From Theorem 7 and Lemma 8, we can get the relationship of the parameter sensitivities in B and B' for the probability parameters of the same variable that involved in the covered edge reversal.

Theorem 9. *Let θ^{B}_{ilk} and $\theta^{B'}_{il^wk}$ be as before. Let $S(\theta^{B}_{ilk})_{y|e}$ and $S'(\theta^{B'}_{il^wk})_{y|e}$ be the parameter sensitivities for θ^{B}_{ilk} and $\theta^{B'}_{il^wk}$ respectively. Then*

$$S'(\theta^{B'}_{il^wk})_{y|e} = S(\theta^{B}_{ilk})_{y|e}p(x^w_j)$$

Proof. From the definition of parameter sensitivity in Definition 5, we have the sensitivity of the parameter $\theta^{B'}_{il^wk}$ in B' expressed as a fraction:

$$S'(\theta^{B'}_{il^wk})_{y|e} = \frac{\partial p(x_t|e)}{\partial \theta^{B'}_{il^wk}} = \frac{\alpha'\delta' - \beta'\gamma'}{(\gamma'\theta^{B'}_{il^wk} + \delta')^2}$$

Take the sensitivity coefficients α', β', γ' and δ' as derived in the Theorem 7 and Lemma 8 into the above definition formula, we have the numerator of this expression

$$\begin{aligned}
\alpha'\delta' - \beta'\gamma' &= \alpha p(x^w_j)(\gamma \textstyle\sum_{u\neq w} \theta^{B'}_{il^uk}p(x^u_j) + \delta) - (\alpha \textstyle\sum_{u\neq w} \theta^{B'}_{il^uk}p(x^u_j) + \beta)\gamma p(x^w_j) \\
&= \alpha p(x^w_j)\gamma \textstyle\sum_{u\neq w} \theta^{B'}_{il^uk}p(x^u_j) + \alpha p(x^w_j)\delta - \alpha p(x^w_j)\gamma \textstyle\sum_{u\neq w} \theta^{B'}_{il^uk}p(x^u_j) - \beta\gamma p(x^w_j) \\
&= (\alpha\delta - \beta\gamma)p(x^w_j)
\end{aligned}$$

The denominator of the expression can be derived as

$$\begin{aligned}
(\gamma'\theta^{B'}_{il^wk} + \delta')^2 &= (\gamma p(x^w_j)\theta^{B'}_{il^wk} + \gamma \textstyle\sum_{u\neq w} \theta^{B'}_{il^uk}p(x^u_j) + \delta)^2 \\
&= (\gamma \textstyle\sum_{w=1}^{r_j} \theta^{B'}_{il^wk}p(x^w_j) + \delta)^2 \\
&= (\gamma\theta_{ilk} + \delta)^2
\end{aligned}$$

Therefore,

$$S'(\theta^{B'}_{il^wk})_{y|e} = \frac{\alpha'\delta' - \beta'\gamma'}{(\gamma'\theta^{B'}_{il^wk} + \delta')^2} = \frac{(\alpha\delta - \beta\gamma)p(x^w_j)}{(\gamma\theta_{ilk} + \delta)^2} = S(\theta_{ilk})_{y|e}p(x^w_j)$$

□

Consider the maximum network sensitivity, unless the maximum value of the parameter sensitivity is larger than the sensitivity values of the two nodes involved in the covered edge reversal, the network sensitivity changes with the edge reversal.

Example: Let A and B be boolean random variables that take the values $\{a, \tilde{a}\}$ and $\{b, \tilde{b}\}$, respectively. Their joint distribution can be represented by two equivalent Bayesian networks:

1. $A \rightarrow B$ (meta-parameters $\theta = (p(a)$,$p(b|a)$,$p(b|\tilde{a})))$,
2. $A \leftarrow B$ (meta-parameters $\theta = (p(a|b)$,$p(a|\tilde{b}))$, $p(b))$.

Note that we only need to use meta-parameters in the networks instead of all the parameters defined in the probability distributions. For example, $p(a)$ is used as a meta-parameter for node A in network $A \rightarrow B$ and $p(\tilde{a})$ is ignored. This is because $p(\tilde{a}) = 1 - p(a)$, and therefore, the absolute values of their parameter sensitivities are same.

Obviously, the node A has no parents in the network $A \rightarrow B$, but has B as its single parent in the network $A \leftarrow B$. Its parameter $p(a)$ can be decomposed as:

$$p(a) = p(a|b)p(b) + p(a|\tilde{b})p(\tilde{b})$$

First let us consider the case of $p(x_t, e) = x^B_{ilk}$. For a query $p(a)$ posed to the network $A \rightarrow B$, the sensitivity for the parameter $p(a)$ equals to 1. Therefore, $\alpha = 1$ and $\beta = \gamma = \delta = 0$; For the same query posed to the network $A \leftarrow B$, the sensitivity function for the parameter $p(a|b)$ reduced to a linear function. The coefficients of this linear function is

$$\begin{cases} \alpha' = p(b) = \alpha p(b) \\ \beta' = p(a|\tilde{b})p(\tilde{b}) = p(a|\tilde{b})p(\tilde{b}) + \beta \ , \end{cases}$$

Thus the parameter sensitivity $S'(p(a|b))_{p(a)} = S(p(a))_{p(a)}p(b)$. The result verifies the Theorems 7 and 9. Same result can be obtained for the other decomposing factor $p(a|\tilde{b})$.

Second, let us consider the case of $p(x_t, e) = x^{B'}_{il^w k}$, for example, the query $p(a|b)$. Note that the decomposition of $p(a)$ can be rewritten as the following form,

$$p(a|b) = \frac{p(a) - p(a|\tilde{b})p(\tilde{b})}{p(b)} \ .$$

For the query $p(a|b)$ posed to the network $A \rightarrow B$, the sensitivity function for $p(a)$ is a linear functioin with the coefficients as:

$$\begin{cases} \alpha = \frac{1}{p(b)} \\ \beta = -\frac{p(a|\tilde{b})p(\tilde{b})}{p(b)} \end{cases} .$$

The corresponding sensitivity function in the network $A \leftarrow B$ for the decomposing factor $p(a|b)$ is the constant 1. The sensitivity coefficients for parameter $p(a|b)$ can be expressed as a function of the sensitivity coefficients for parameter $p(a)$ as below:

$$\begin{cases} \alpha' = 1 = \frac{1}{p(b)}p(b) = \alpha p(b) \\ \beta' = 0 = \frac{1}{p(b)}p(a|\tilde{b})p(\tilde{b}) - \frac{p(a|\tilde{b})p(\tilde{b})}{p(b)} = \alpha p(a|\tilde{b})p(\tilde{b}) + \beta \end{cases} .$$

And thus the parameter sensitivity $S'(p(a|b))_{p(a|b)} = S(p(a))_{p(a|b)}p(b)$. Again, the result verifies the Theorems 7 and 9. And the same result can be obtained for the other decomposing factor $p(a|\tilde{b})$.

Third, let us consider the case of $p(x_t, e) = p(x_j^w)$, for example, the query $p(b)$. For the network $A \rightarrow B$, $p(b)$ can be expressed in the meta-parameters as $p(b) = p(a)p(b|a) + (1 - p(a))p(b|\tilde{a})$. Then the vector of parameter sensitivities is

$$\frac{\partial q(\theta)}{\partial \theta} = (p(b|a) - p(b|\tilde{a}), p(a), 1 - p(a)) \quad (5.7)$$

For the equivalent network $A \leftarrow B$, $p(b)$ is a meta-parameter encoded in the network. Therefore, the change of query $p(b)$ can be directly determined by the meta-parameter without any computation using the rest of parameters. So the $p(b)$'s parameter sensitivity with regard to the query equals to 1 and all other parameters' sensitivity equal to 0. The vector of parameter sensitivities is

$$\frac{\partial q(\theta)}{\partial \theta} = (0, 0, 1)$$

That is, $S'(p(a|b))_{p(b)} = S'(p(a|\tilde{b}))_{p(b)} = 0$ but $S(p(a))_{p(b)} = p(b|a) - p(b|\tilde{a}) \neq 0$ unless $p(b|a) = p(b|\tilde{a})$. The result, again, verifies the Theorems 7 and 9.

5.5 SENSITIVITY IS NOT AN APPROPRIATE CRITERION FOR MODEL SELECTION

As shown in the previous section, using maximum value of parameter sensitivity as a measure of sensitivity for Bayesian networks, the relationship between the sensitivities of equivalent networks reduces to a single, equivalent, local probability distribution after one edge reversal. The covered edge reversal transforms a DAG into one of its equivalent graphs. Populating the two graphs with the corresponding probability distributions, the equivalent networks only differ in the local probability distributions for the two nodes X_i and X_j that involve in the edge reversal. When node X_i adds node X_j as additional parent in the equivalent graph G', node X_j must deletes node X_i from its original parent lists in the equivalent graph G. Therefore, the conditional probability distributions for node X_i before the edge reversal is decomposed under the condition of X_j's states. The decomposition effect reflects the sensitivity difference and actually gives a simple relationship between the sensitivities of the equivalent networks.

Now that the sensitivities of equivalent networks can be different, and their relationship can be expressed linearly to the local probability distributions, using sensitivity to guide the search for model selection is therefore reduced to the comparison of local probability distributions. However, the fact is that equivalent networks have the same joint probability distribution over the domain. For two equivalent networks that have only one edge direction different , the local probability distribution is equivalent. The only difference is the factor of decomposition. But the decomposition is actually symmetrical. In the above example, the probability distribution of node X_i in G is decomposed in G' with the edge reversal, at the mean time, the probability distributions of node X_j in G' is decomposed in G. In this sense, we say that the local probability distribution is equivalent. Therefore, the sensitivity which is only different in a factor expressed by the probability distribution is not appropriate to guide the search for Bayesian networks.

Furthermore, based on the linear relationship of the sensitivities of the equivalent networks, selecting a low sensitive model from the equivalence class is essentially same as selecting a model with low probability values in the local distributions for the nodes involving in the edge reversal. As shown by Renooij and van der Gaag [61], a uniform probability distribution has the lowest sensitivity bounds. Selecting a low sensitive model using such a sensitivity measure then prefers uniform distributions, which is often interpreted as a total random process with the highest uncertainty. This violates the principles of Bayesian network modelling, which encodes the expert knowledge at the best effort and is supposed to prefer certain level of certainties in the models. But selecting the low sensitive model leads to the contrary choice, if the sensitivity measure is based on the maximum value of parameter sensitivities.

Therefore, we can draw the conclusion that model selection using (maximum) sensitivity criterion based on parameter sensitivity is not appropriate for learning Bayesian networks.

5.5.1 Discussion

Note that the assumption of proportional scaling was used in Coupe's theory. It is the foundation of the theories that I derived. In case that the fundamental assumption does not hold, for example, the co-vary of probability parameters in the same distribution takes another form in the Bayesian network modelling, then the parameter sensitivity and the network sensitivity may take a different form and the different results can be derived. Also, the network sensitivity definition can take other form regardless of the parameter sensitivity definition. Or other measures are used for network sensitivity. Consequently, the conclusion of the chapter may be different.

6.0 USING SENSITIVITY ANALYSIS FOR SELECTIVE LEARNING OF PROBABILITY PARAMETERS

6.1 INTRODUCTION

The results of Chapter 4 indicate that a Bayesian network often shows different sensitivity to different sets of its probability parameters. That means, some probability parameters may have a larger effect on the network's performance than others. In other words, some probability parameters are more influential on the network performance. The erroneous estimate of these important parameters may greatly deteriorate the network quality. This happens in both knowledge engineering approach and the machine learning approach to building Bayesian networks.

Traditionally, Bayesian networks have been used as a knowledge-engineering tool for representing expert knowledge and reasoning under uncertainty. However, such knowledge-engineering approach can be quite expensive and time-consuming. Over the last decade, the research focus is shifting more towards learning Bayesian networks from data, especially with increasing volumes of data available in biomedical, internet, and e-business applications. In the past few years, significant progress has been made in developing techniques for learning Bayesian networks [29].

Among the various learning approaches, the Bayesian learning, which utilizes the domain knowledge as well as the data, is the most appealing and becomes the standard method for estimating probability distributions [14]. In this learning method, the prior knowledge of a domain expert are treated as an equivalent pseudo (or imaginary) data set which observes Dirichlet distributions [27]. The Dirichlet exponent parameters, referred to as hyperparameters, are used to represent the equivalent sample size of the expert's prior knowledge. Since the number of the hyperparameters is same as the number of the probability parameters needed in a Bayesian network, providing informative hyperparameters for such learning algorithms is highly demanding. If uniform hyperparameters are used in learning, the only possibility to learn the accurate probability parameters seems to be having a sufficiently large data set.

However, the data set can be relatively scarce for learning the large number of probability parameters. A real-domain Bayesian network has typically hundreds or thousands of the probability parameters. This requires a large data sets in order to learn accurate parameters, especially for probability distributions describing rare events. Unfortunately, in real-life settings, the data set is often too scarce, and the probability values computed from the data set by the learning methods without informative hyperparameters may therefore be erroneous, especially for the rare events. One of such instances happens in the Bayesian network models built for airplane diagnosis. In these networks, the Line Replaceable Units (LRUs) are represented as parentless nodes with prior probabilities describing their failure rates. The LRUs are manufactured highly reliable to achieve very high standard for flight safety. They are required to support over 10^5 flight hours before any failure occurs. Therefore, the prior probabilities of the LRU failures are extremely small, usually less than 10^{-5}. In such cases, there may be very few data describing failure situation even though data set for learning is large.

On the other hand, not all parameters are equally important because their effect on the network's performance can be different [40, 74]. Sensitivity analysis of Bayesian networks studies the network's reliability in the presence of noise in its parameters. Therefore, applying sensitivity analysis on a network can identify the most important parameters for further refinement. It may be sufficient for learning a good-quality network only providing the informative hyperparameters to the most important parameters without allocating effort to acquire prior knowledge for all of the probability distributions.

In this chapter, I will present a method that uses sensitivity analysis for selective learning of the probability parameters for Bayesian networks [78]. This method first runs sensitivity analysis on the Bayesian network learned with uniform hyperparameters to identify the most important probability parameters. Then it updates this set of probabilities to their accurate values by acquiring their informative hyperparameters. The process is repeated until further elaboration of probabilities does not improve the performance of the network or other stopping rules satisfied.

The following sections start with a review of methods for learning probability parameters. It then argues that parameter sensitivity is a more appropriate measure for parameter importance than mutual information. The following section describes a procedure for selective learning of probability parameters for Bayesian networks. The experimental results will be presented to illustrate the convergence of the learned probability distributions to the true probability distributions. Performance comparison will be given between the learning methods with uniform hyperparameters, with informative hyperparameters for the most important parameters, and the exhaustive update for all the parameters with informative hyperparameters. Finally I will give a discussion on applying the selective learning method in automatic model validation and active learning.

6.2 LEARNING PROBABILITY PARAMETERS

In classical statistical approach, the probability parameters are viewed as *physical property*, though unknown, of the world. They are assumed *objective* constants that can be estimated purely from a data set of training examples using *maximum-likelihood (ML)* estimates. The log-likelihood $logP(D|\Theta)$ can be decomposed according to the graph structure G using the chain-rule representation of joint probability in the equation 6.1 and expressed as below:

$$logP(D|\Theta) = \sum_{i,j,k} N_{ijk} log\theta_{ijk}, \tag{6.1}$$

where N_{ijk} are *sufficient statistics* representing the number of data instances matching the instantiations $X_i = x_i^k$ and $\mathbf{Pa_i} = \mathbf{pa_i}^j$. It is easy to show that this expression is maximized by the frequencies (maximum-likelihood estimates) $\hat{\theta}_{ijk} = \frac{N_{ijk}}{N_{ij}}$, where N_{ij} is the number of samples matching the assignment $\mathbf{Pa_i} = \mathbf{pa_i}^j$, and $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$.

Bayesian approach takes a different view at the probability parameters. In Bayesian statistics, the probabilities represent degree of *subjective* belief. The parameters are unknown variables governed by probability distributions. We assume some prior belief (e.g., based on background knowledge or historical information) in Θ that is represented by the *prior distribution* $P(\Theta)$. When a new data set D becomes available, this belief is updated according to Bayes' rule $P(\Theta|D) = \frac{P(D|\Theta)P(\Theta)}{P(D)}$. Thus, the Bayesian approach takes advantage of prior knowledge about the parameters, which is especially useful when data are scarce. Imagine all possible values of Θ from which this data set could have been generated. The *maximum a posteriori (MAP)* estimate of Θ is the expectation of Θ with respect to our posterior beliefs about its value:

$$E_{p(\Theta|D)}(\Theta) = \int \Theta \; p(\Theta|D) \; d\Theta \; .$$

A common approach to modelling the prior belief over *multinomial* variables $\mathbf{X}$ with the parameters Θ uses *Dirichlet distribution*, a *conjugate* distribution to multinomial, which has a nice property that the posterior $P(\Theta|D)$ belongs to the same *conjugate family* as the prior $P(\Theta)$ [27]. For a variable $X_i \in \mathbf{X}$, its probability distribution $\theta_{ij} \in \Theta$ which observes *Dirichlet distribution* is defined as follows:

$$Dir(\theta_{ij}|\alpha_{ij1}, \ldots, \alpha_{ijr_i}) \equiv \frac{\Gamma(\alpha_{ij})}{\prod_{k=1}^{r_i} \Gamma(\alpha_{ijk})} \prod_{k=1}^{r_i} \theta_{ijk}^{\alpha_{ijk}-1} \; ,$$

where $\alpha_{ij} = \sum_{k=1}^{r_i} \alpha_{ijk}$ and $\Gamma(\cdot)$ is the *Gamma-function* which satisfies $\Gamma(x+1) = x\Gamma(x)$ and $\Gamma(1) = 1$. The exponent parameters α_{ijk} are often called *hyperparameters*, in order to distinguish them from the θ_{ijk} parameters of the corresponding multinomial distribution. A common interpretation for α_{ijk} parameter is the number of times that an expert has previously observed the instantiation of $X_i = x_i^k$ and $\mathbf{Pa_i} = \mathbf{pa_i}^j$. For that, the α-parameters are also called *equivalent sample size* (i.e. the size of a data set that is an equivalent of the expert's knowledge). Thus, greater hyperparameters reflect higher confidence in our prior. Given a set of observations D on a multinomial variable X_i with the parameters $\theta_{ij} = \{\theta_{ijk} | 1 \leq k \leq r_i\}$, it is easy to see that the posterior $P(\theta_{ij}|D)$ is also Dirichlet:

$$\begin{aligned} P(\theta_{ij}|D) &\propto P(D|\theta_{ij})P(\theta_{ij}) \\ &\propto \prod_{k=1}^{r_i} \theta_{ijk}^{N_{ijk}} \cdot \prod_{k=1}^{r_i} \theta_{ijk}^{\alpha_{ijk}-1} \\ &\propto \prod_{k=1}^{r_i} \theta_{ijk}^{N_{ijk}+\alpha_{ijk}-1} \; . \end{aligned}$$

Therefore (taking into account normalization constant),

$$P(\theta_{ij}|D) = Dir(\theta_{ij}|\alpha_{ij1} + N_{ij1}, ..., \alpha_{ijr_i} + N_{ijr_i}) \ .$$

Given a network structure G, a complete data set D without missing values, a set of Dirichlet prior parameters α_{ijk}, and the assumption of parameter independence, which says that θ_{ij} is independent of $\theta_{ij'}$ for all $j \neq j'$, it can be shown that the expected value of the parameters of the network with respect to the posterior distribution $p(\theta_{ij}|D, G, \alpha_{ij})$ can be estimated by Equation 6.2:

$$\hat{\theta}_{ijk} = \frac{\alpha_{ijk} + N_{ijk}}{\alpha_{ij} + N_{ij}} \ , \tag{6.2}$$

where $\alpha_{ij} = \sum_{k=1}^{r_i} \alpha_{ijk}$, and $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$.

As is apparent from Equation 6.2, the Dirichlet exponents α_{ijk} completely specify a user's current knowledge about the domain for purposes of learning probability parameters of the Bayesian network. Unfortunately, the specification of α_{ijk} for all possible $(x_i^k, \mathbf{pa_i^j})$ configurations corresponding to all values of i, j, and k is demanding, to say the least. Most existing learning algorithms simply adopt an uninformative assignment. For example, Cooper and Herskovits [14] suggest a global uniform distribution with $\alpha_{ijk} = 1$ for all values of i, j, and k; Buntine [3] suggests an local uniform assignment $\alpha_{ijk} = \alpha/(r_i \cdot q_i)$, where r_i is the number of X_i's possible values and q_i is the total number of $\mathbf{Pa_i}$'s configuration. With additional assumption of *likelihood equivalence* [1] and introducing *complete network* structures [2], Heckerman *et al.*[30] derived an exponent constraint on the α_{ijk} parameters. As a consequence, informative prior for the α_{ijk} parameters can be constructed by building a complete network S_c and assessing an equivalent sample size α for S_c.

$$\alpha_{ijk} = \alpha \cdot p(X_i = x_i^k, \mathbf{Pa_i} = \mathbf{pa_i^j}) \ ,$$

[1]Likelihood equivalence says that, for any database D, the probability of D is the same given hypotheses corresponding to any two equivalent network structures. Two network structures are said to be equivalent when they encode the same independence relationships between nodes, only that the directions of edges can be different.

[2]A complete network is a network that has no missing edges. It encodes no assertions of conditional independence. In a domain with n variables, there are $n!$ complete network structures.

where $p(X_i = x_i^k, \mathbf{Pa_i} = \mathbf{pa_i^j})$ is the joint probability in the complete network S_c. Whereas assessing an equivalent sample size α is simple, building a complete network and fully quantifying it can be formidable. Most of the current learning algorithms simply ignore the variety of background knowledge of the domain experts by using uninformative prior for the α_{ijk} hyperparameters.

As discussed above, both *maximum-likelihood (ML)* estimate and *maximum a posteriori (MAP)* estimate can be used to quantify a Bayesian network. However, a typically huge number of probability parameters in a Bayesian network may require very large data set in order to get accurate estimates, especially for probability distributions involving rare events. In many real-world applications, data are relatively scarce and result into extreme (close to 0 or 1) values for the probability parameters, especially in *ML* estimates. The *MAP* estimate can avoid this problem by choosing appropriate hyperparameters that represent domain background knowledge. But by using uniform distributions for the hyperparameters, the learning algorithms actually ignore the background knowledge. Therefore, the *MAP* estimates may also deviate from the true probability values. As discovered in experiment in Chapter 3, the values of α_{ijk} hyperparameters indeed affect the accuracy of learned probability values.

Since fully eliciting all of the hyperparameters necessary for learning a Bayesian network is just as expensive as fully quantifying the network in knowledge engineering approach, Selective learning the most important parameters in a network may provide a feasible approach to efficient fusion of expert knowledge and data. This technique is useful when data is limited and experts are available. In the following sections, I first argue that parameter sensitivity is a more appropriate measure for the importance of a parameter than mutual information with regard to the query target. Then I give more detail on the mixed approach to quantifying Bayesian networks using both learning and knowledge elicitation methods.

6.2.1 Mutual Information vs. Sensitivity as Importance Measure

One possible measure for importance is mutual information, which is closely related to the definition of *entropy* and *conditional entropy*. The entropy of a discrete random variable X

with probability mass function $p(x) = p(X = x)$ is defined as

$$H(X) = -\sum_x p(x) \log p(x) \ .$$

The conditional entropy of a pair discrete random variables Y given X is defined as

$$\begin{aligned} H(Y|X) &= \textstyle\sum_x p(x) H(Y|X = x) \\ &= -\textstyle\sum_x p(x) \sum_y p(y|x) \log p(y|x) \\ &= -\textstyle\sum_x \sum_y p(x, y) \log p(y|x) \ . \end{aligned} \tag{6.3}$$

It is easy to show that $H(Y|X) = 0$ when Y is a function of X, i.e., $Y = g(X)$. Since for all x with $p(x) > 0$, the value of Y is determined as $y = g(x)$ with probability $p(y|x) = 1$.

Mutual information measures the amount of information one random variable contains about another. For two random variables X and Y with a joint probability function $p(x, y)$, the *mutual information* $I(X; Y)$ is the relative entropy between the joint distribution $p(x, y)$ and the product distribution $p(x)p(y)$, and can be expressed by entropy and conditional entropy,

$$I(X; Y) = \sum_x \sum_y p(x, y) \log \frac{p(x, y)}{p(x)p(y)} = H(Y) - H(Y|X) = H(X) - H(X|Y) \ .$$

When Y is a function of X, $I(X; Y)$ degrades to the the entropy of the responsive variable $H(Y)$, in other words, it becomes *self information*. In this case, the information gain of knowing X is zero to Y.

Therefore, following the sensitivity function in Theorem 4, when measuring the importance of a probability parameter θ_{ijk} to the query q in form of "what is $q = p(X_t = x_t|\boldsymbol{E} = e)$?", the mutual information between θ_{ijk} and q is equal to the entropy of the q, which is determined purely by q's distribution without any association with θ_{ijk}, for any i, j, k. θ_{ijk}'s changes do not affect the mutual information $I(q, \theta_{ijk})$. Using mutual information as an importance measure is not appropriate in such situation. The derivative $\partial q / \partial \theta_{ijk}$ is more indicative as it is the first order approximation of q's changes as the effect of θ_{ijk}'s variation in a small range.

6.3 SELECTIVE LEARNING THE PROBABILITY PARAMETERS FOR BAYESIAN NETWORKS

Using parameter sensitivity as defined in Equation 5.4, we can identify which parameters in a Bayesian network are the most important with respect to the queries. Intuitively, when the different parameters undergo the same amount of variation, those with higher parameter sensitivity causes bigger changes in the query, and thus, affect the network's performance stronger. Efficient algorithms for sensitivity analysis in Bayesian networks, such as the one we presented in Algorithm 3, made it possible to quickly identify the important probability parameters. As a result, we can focus our effort to refine the prior hyperparameters corresponding to the important parameters in Bayesian learning.

Algorithm 1 Selective Parameter Refinement

```
Input: a Bayesian network $B = (G, \Theta)$, sensitivity threshold $\delta$, sensitivity decreasing rate $\varepsilon$
Output: a Bayesian network $B = (G, \Theta')$
$cose\_effective = true$;
while $cost\_effective$ do
  (1) Calculate sensitivity $S(\theta)$ for all meta parameters $\theta$ in $B$;
  (2) Identify a set of the most sensitive parameters $paraIndex = \{\theta : S(\theta) > \delta\}$;
  if all $paraIndex$ have been refined before, or cost of knowledge elicitation is unaffordable then
    $cost\_effective = false$;
  else
    (3) Extract priors $\alpha$ for all $paraIndex$ from experts or new data;
    (4) Recompute parameters for all $paraIndex$ with $\alpha$ and data counts;
    (5) Update $B$ with the new parameters $\Theta'$, s.t. $B = (G, \Theta')$;
    (6) Update $\delta = \delta - \varepsilon$ ;
  end if
end while
Output $B$
```

As summarized in Algorithm 1, the parameters in a Bayesian network can be selectively refined as the following procedure. It begins with a Bayesian network learned from data with uniform distributions for the prior hyperparameters. Then importance of each meta parameter in the network is computed using sensitivity analysis algorithm. Given a threshold of the importance value, those parameters with a higher value than the threshold are identified and put into the *important parameter set*. For the important parameters not refined before, informative prior parameters are collected from domain experts or learned from new data. With the updated informative hyperparameters for the important probability parameters, MAP estimates are recomputed for these probabilities and the network is repopulated with the newly learned probability parameters. Iteratively perform sensitivity analysis and re-parameterize the network with informative priors until the stopping rules are satisfied. In this iteratively repeated procedure, the domain experts can focus their attention on the probabilities to which the network's behavior shows high sensitivity. Those uninfluential parameters can be left with crude estimates.

Generally, the stopping rules should take into account the following criteria. a) Satisfactory behavior of the network is achieved. b) higher accuracy can no longer be attained due to lack of knowledge. c) Obtaining more accurate estimates for parameters is not affordable any more, e.g., due to the limited time available from the domain experts or the high cost of the interviewing process. And, d) all important parameters are refined already. Note that this procedure is iterative, and the threshold for classifying parameters into class of importance can be adjusted to accommodate the availability of resource.

6.4 EXPERIMENT AND RESULTS

The goal of the experiment is to illustrate the convergence of probability distributions with selective refinement to the true probability distributions. Three networks were used for the test: CPCS (179 nodes version), HEPAR network, and ALARM network. The learning data sets were generated by probabilistic logic sampling[32]. I did not use the two airplane diagnostic networks PNEUMATIC and OIL because the simulated data for learning were extremely scarce. Even though the sample size was as large as 10000, there was not a single data point generated for failure case. It turned out that all of the 10000 data points describe the normal behavior of airplanes without any LRU failures. This is because these two networks have extremely low prior probabilities (typically less than 10^{-5}) for the parentless nodes, and the probabilistic logic sampling generates data based on the prior probabilities of the event occurrence. Therefore, the rare events may not show up in the data sets at all.

For each of the tested networks, I sampled the learning data set with size of 1000, 3000, 5000, and 10000, each for 10 times, totally there were 40 data sets for learning. For each individual data set, I applied the Bayesian learning methods with the identical structure of the original networks. The probability parameters were learned by three learning procedures respectively. The first learning procedure took uniform hyperparameters. The second learning procedure took the informative hyperparameters for the important parameters. And the third learning procedure took informative hyperparameters for all of the parameters. In the following paragraphs, I will refer to the three learning methods as *uniform learning*, *selective learning*, and *exhaustive learning* respectively.

A parameter was selected as important when the node it belongs to has a maximum sensitivity value greater than 10^6. The sensitivity analysis was performed based on 50 randomly generated evidence scenarios for each of the networks. Same as the experiment in Chapter , the query targets for sensitivity analysis are the diagnostic targets that are preset into the GeNIe network format.

The informative hyperparameters take the value of the product of the sample size and the corresponding probability parameters from the original benchmark networks. The assumption is that the experts have experienced the domain events at least the same size as the learning data, and the domain events they experienced have the same probabilities as encoded in the original networks, i.e., the original benchmark networks represent the domain expert knowledge faithfully.

I used *Kullback Leibler (KL) distance* [43], (also known as *relative entropy*) between the learned distributions to the true distributions encoded in the original benchmark network to measure the accuracy of the learned probability values. KL distance between the two probability distributions $p(x)$ and $q(x)$ is defined as

$$D(p||q) = \sum_x p(x) log \frac{p(x)}{q(x)} \ .$$

For each run of the learning methods, The KL distances between the learned probability distributions and the original probability distributions were calculated. Their average and standard deviation were also calculated across the 40 runs for each of the learning procedure. Table 6.1 and 6.2 show the experimental results on accuracy and time respectively.

Table 6.1: Learning Performance: Accuracy Comparison (KL-Distances Between The Learned Probability Distributions And The Original Probability Distributions)

	uniform learning		**selective learning**		**exhaustive learning**	
	μ	σ	μ	σ	μ	σ
ALARM	40.4584	0.1244	14.4195	1.7937	0.0000	0.0000
HEPAR II	35.0126	0.1878	14.3133	1.4885	0.0000	0.0000
CPCS	165.7404	0.3445	14.6955	0.5203	0.0000	0.0000

As expected, the probability distributions learned by the uniform learning method have the largest divergence from the original probability distributions. The selective learning method reduced the KL-divergence by at least two to three folds. The exhaustive learning method produced zero distance between the learned probability and the original probability distributions, this is no surprise because the hyperparameters are assigned the true values based on the original probability distributions and the learning sample sizes. And the large sample size (1000, 3000, 5000,and 10000) used in the experiment represented a very strong opinion of the domain expert.

Figure 6.1 displays the difference of the performance in terms of the accuracy of the learned probability distributions. Obviously, the more informative hyperparameters are provided in learning, the more accurate probability parameters can be obtained by the Bayesian learning methods. Under the constraint of availability of knowledge elicitation or the sufficient data set for learning, the selective update of the probability parameters improves the quality of the learned Bayesian networks.

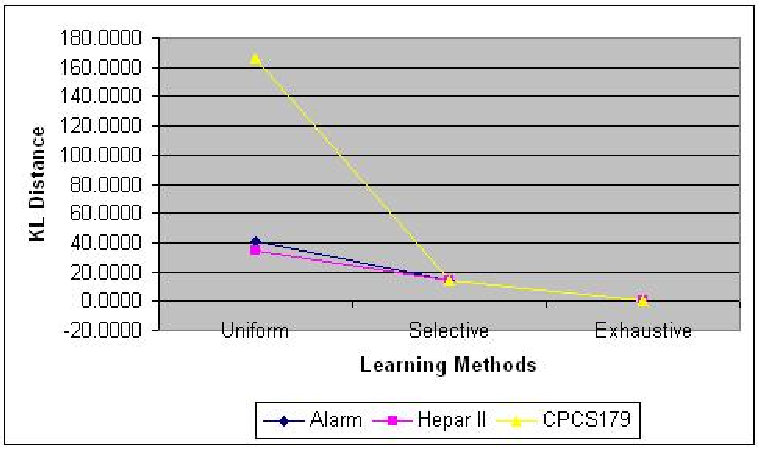


Figure 6.1: Learning Result Comparison: KL-Distances Between The Learned Probability Distributions And The Original Probability Distributions

Table 6.2: Learning Performance: Time Comparison (In Seconds)

	uniform learning		**selective learning**		**exhaustive learning**	
	μ	σ	μ	σ	μ	σ
ALARM	0.0007	0.0000	0.0020	0.0001	0.0015	0.0000
HEPAR II	0.0020	0.0001	0.0065	0.0004	0.0042	0.0000
CPCS	0.0145	0.0005	0.1138	0.0189	0.0311	0.0020

Table 6.2 lists average and standard deviation of the time used by each of the learning procedures. From the table, we see that the uniform learning method is the most fastest one, the exhaustive learning method takes about double the time as unform learning method, and the selective learning method takes triple the time or longer. The exhaustive learning is slower because of additional time needed for the access of the probability parameters in the original Bayesian networks and the computation of informative hyperparameters. The selective learning is even slower because it needs extra time for sensitivity analysis. Overall, the learning procedures run pretty fast for the tested networks, each run can be finished within a second.

6.5 DISCUSSION

In real practice in building Bayesian networks, the cost of eliciting domain knowledge is the most expensive, and most of the time, the bottleneck of the whole procedure for building Bayesian networks. From the experimental results, we see that the selective learning method improves the accuracy of probability parameters with a small cost of sensitivity analysis. When learning Bayesian networks from data, if domain knowledge is accessible, the selective learning method can be applied to gain high quality of network without going through the full knowledge acquisition, which is often too expensive and time-consuming.

The same technique can also be applied to automatic model validation. If we have a set of test scenarios and use it as learning data records, we can use the existing probability parameters in the network as prior knowledge, and selectively update the important parameters by learning from the test scenarios, i.e., only apply minimum change to the network model to make it perform well on the testing data. This way the network model is validated automatically.

Similarly, the selective learning can also be used in active learning of Bayesian networks, where sensitivity analysis can suggest which data is the most informative for future data collection or data processing.

6.5.1 Automating Model Validation Using Selective Learning

After a Bayesian network is built, validation is necessary to see if the network performs on test data as expected. Traditionally the validation is a procedure of manually testing the network using some real scenarios and tuning the network probability parameters so that the network agrees on the expert judgement, for instance, the posterior probabilities of certain variables of interest fall in the expected range of values, or the rank of probable faults is reasonable. Although tools are available for suggesting a valid range of probability parameters in order to meet the expert judgement, the validation process is manual and ad hoc [6]. The same parameter can be changed subsequently many times in various test cases, which in turn, erases the values that work just fine for the previous tested cases.

To automate the validation process, the selective learning method can be applied. Given a Bayesian network model and a set of records of real scenarios, we can modify some of the parameters of the model by selective learning from the data such that the model, with minimal changes as necessary, will perform better on the data but, at the same time, that it will not over fit the data. The selective learning is more appropriate for automatic validation than the exhaustive learning, because the built model has certain level of validity, and minimal changes only applied when necessary, implies highest reliability. The hyperparameters can be readily obtained from the existing model, as we did in the experiment. They are used as weight so that the prior over the model still has some importance.

The problem is that the data records for validation test often contain only values for observable nodes and, possibly, values for fault nodes. The values for intermediate nodes are often missing so that the data records are highly sparse. EM(Expectation and Maximization) algorithm [20] may be applied in such case, but the overall performance of learning in such extreme case may need further study.

6.5.2 Active Learning

The algorithm for refining the probabilities can also be used for active information gathering. The active learning enables selecting new data for those important probabilities for more accurate estimate. When collecting data is very expensive, or when data volume is extremely large and requires too long time to process, it may be unaffordable to get all of the probability parameters at the same accuracy level. Under these situations, selective learning can be adapted to active learning, where sensitivity can be used as a heuristic to guide the future data collection by querying for certain types of samples for learning.

As discussed above, the selective learning method can be applied not only in the learning algorithms for quantifying Bayesian networks, but also in automatic model validation and active learning etc. These will be two of my research focuses that I would like to pursue in the near future.

7.0 SUMMARY

The contribution of this thesis can be concluded as follows:

- The investigation and development of user interface tools for navigation in CPTs and elicitation of probabilities. The tools enhance greatly user navigation in CPTs during the process of model building and help to improve both the quality and speed of elicitation. Also, the flexible navigation and visualization of probability distributions help to detect unspecified probabilities and inconsistency in responses. Combined, these tools provide a pleasant and powerful visual environment in which experts can give their qualitative estimates of numerical probabilities. Except *CPTree*, all of the graphical tools are now adopted in the formal release of *GeNIe 2.0*.
- The investigation and development of the objective evaluation of the elicitation methods for probability distributions and the structure of probabilistic models. I invented a method based on machine learning the expert's beliefs when data of the expert's learning knowledge is available. The evaluation approach is justified based on experimental results. It can be used to evaluate the elicitation methods for probabilities across different users. It can also be used to assess the best technique for a given expert.
- The discovery that log-odds normal distribution is valid as a model for sensitivity analysis only in the range of standard deviations where the distribution is unimodal. I also show that using average posterior probabilities as criterion to measure the sensitivity may not be the most indicative, especially when the distribution is very asymmetric as is the case at nominal values close to zero or one. It is proposed, instead, to use the partial ordering of the most probable causes of diagnosis, measured by a suitable lower confidence bound on the change in the rank order. Preliminary results of the sensitivity analysis

experiments were shown with three Bayesian networks built for diagnosis of airplane systems. The results showed that some networks are more sensitive to imprecision in probabilities than previously believed.

- The finding of relationship between the sensitivities of the equivalent networks, and using sensitivity may not be appropriate for learning Bayesian network structures. I gave a proof that Markov equivalent networks have different sensitivities but their sensitivities have a simple relationship that can be expressed using the local probability distributions corresponding to only one edge reversal. The establishment of this relationship leads us to a conclusion that sensitivity is not an appropriate criterion for model selection.
- The development of method that uses sensitivity analysis for selective learning of the probability parameters for Bayesian networks [78]. This method first runs sensitivity analysis on the Bayesian network learned with uniform hyperparameters to identify the most important probability parameters. Then it updates this set of probabilities to their accurate values by acquiring their informative hyperparameters. The process is repeated until further elaboration of probabilities does not improve the performance of the network or other stopping rules satisfied. The empirical test results show the usefulness of this technique in building Bayesian networks in the domain that both subjective expert and data are available.

For the future work, it is interesting to investigate applying the selective learning technique in automatic model validation. If we have a set of test scenarios and use it as learning data records, we can use the existing probability parameters in the network as prior knowledge, and selectively update the important parameters by learning from the test scenarios, i.e., only apply minimum change to the network model to make it perform well on the testing data. This way the network model is validated automatically. Similarly, the selective learning can also be used in active learning of Bayesian networks, where sensitivity analysis can suggest which data is the most informative for future data collection or data processing.

APPENDIX

EFFICIENT SENSITIVITY ANALYSIS USING RELEVANCE REASONING

A.1 INTRODUCTION

Traditional approaches to sensitivity analysis are often time-consuming [45, 40]. For example, the experimental approach used in the test in chapter 4 added log-normal noise using Monte Carlo sampling to probability parameters under investigation, and generated a new Bayesian network 100 times for each evidence scenario at each noise level to get statistically sound results. This means that there were 100 belief updates in the entire network. Other approaches may have a large space complexity. The differential approach [18] is such an example. It first compiles a Bayesian network into a multivariate polynomial and then computes the partial derivatives of this multi-linear function with respect to each variable. But the exponential number of terms in the polynomial causes large space requirements and makes the method computationally infeasible for large Bayesian networks.

Another approach to sensitivity analysis on Bayesian networks focuses on the relation between the probability parameters of the network and the posterior marginals of targets, i.e., the variables of interest. Such a relation can be expressed by a simple mathematical function. Castillo et al. [5] showed that a posterior marginal probability is a quotient of two functions that are linear in the parameter. Based on this property, sensitivity analysis in Bayesian networks can be reduced to the task of computing the coefficients of this function to determine the effect of variation in probability parameters. Castillo et al. showed that, for each coefficient to be established, a single network evaluation is sufficient.

Kjærulff and van der Gaag [42] presented a more efficient algorithm for performing sensitivity analysis. Their algorithm requires only two outward propagations in a junction tree for establishing the coefficients in the functions for all possible parameters. In addition, an inward propagation is required for processing evidence. This method makes sensitivity analysis more computationally efficient than previous approaches. However, the propagation steps in the junction tree for the entire network, especially one with very large cliques, can still be computationally expensive. Even worse, the method may be infeasible for sufficiently large networks.

Thus, applying methods that can reduce the size of junction tree or the size of cliques can improve the efficiency of sensitivity analysis. In this chapter, I propose an algorithm for sensitivity analysis that is based on relevance-based decomposition [48]. The essence of relevance-based decomposition is computing the marginal posterior distributions over a large network by decomposing the network into partially overlapping sub-networks. Belief updating in the identified sub-networks can be significantly more efficient than belief updating in the entire network. We show that because message propagation in the corresponding smaller junction trees can be performed much faster, relevance-based decomposition typically speeds up sensitivity analysis. Furthermore, it facilitates sensitivity analysis of very large networks, for which junction tree methods are infeasible when applied to the entire network.

In the following sections, I will present Kjærulff and van der Gaag's method for sensitivity analysis based on the junction tree representation. I describe the relevance reasoning techniques that is used for faster inference and efficient sensitivity analysis and propose an algorithm for sensitivity analysis over multiple target variables based on relevance-based decomposition. Finally, I present empirical results that show that the algorithm compares very favorably to the sensitivity analysis algorithm based on the junction tree representation.

A.2 SENSITIVITY ANALYSIS IN JUNCTION TREE

In the junction tree representation of a Bayesian network, the expressions for $p(x_t, e)$ and $p(e)$ in terms of x can be derived from the potential of a clique containing both the variable

X_i and its parents $\boldsymbol{Pa_i}$ to which the parameter x pertains, we refer this clique to the *host clique* of X_i. One inward propagation towards a clique containing the query target variable X_t and a subsequent outward propagation with an instantiation $X_t = x_t$ are sufficient to determine the values of α and β [42].

Theorem 10. *[42] Let B, J, x, y and e be as before, and let K be a host clique of X_i in J. Now let x change its value from original θ_{ijk} to θ'_{ijk}, and denote the corresponding parameter vectors $\boldsymbol{\theta_{ij}}$ and $\boldsymbol{\theta'_{ij}}$ respectively. Suppose that, in J, an inward propagation has been performed towards a clique containing X_t, and suppose that a subsequent outward propagation has been performed with the value $X_t = x_t$. Now let $\phi_K = p(K, x_t, e)$ be the resulting clique potential for clique K. Then*

$$\begin{aligned} y &= p(x_t, e)(x) = \textstyle\sum_K \phi_K \\ y' &= p'(x_t, e)(x) = \textstyle\sum_K \phi_K \frac{\boldsymbol{\theta'_{ij}}}{\boldsymbol{\theta_{ij}}} \ . \end{aligned} \tag{.1}$$

Therefore, $p(x_t, e)(x) = \alpha x + \beta$ with

$$\alpha = \frac{y - y'}{\theta_{ijk} - \theta'_{ijk}} \text{ and } \beta = \frac{y'\theta_{ijk} - y\theta'_{ijk}}{\theta_{ijk} - \theta'_{ijk}} \ . \tag{.2}$$

Similarly, we can compute γ and δ with only one outward propagation after an inward propagation with evidence e.

Lemma 11. *[42] Let all symbols be as before. Suppose that the evidence e has been processed in J by an inward and a subsequent outward propagation. Let $\phi^*_K = p(K, e)$ be the resulting clique potential for clique K. Then*

$$\begin{aligned} z &= p(e)(x) = \textstyle\sum_K \phi^*_K \\ z' &= p'(e)(x) = \textstyle\sum_K \phi^*_K \frac{\boldsymbol{\theta'_{ij}}}{\boldsymbol{\theta_{ij}}} \ . \end{aligned} \tag{.3}$$

Then $p(e)(x) = \gamma x + \delta$, where

$$\gamma = \frac{z - z'}{\theta_{ijk} - \theta'_{ijk}} \text{ and } \delta = \frac{z'\theta_{ijk} - z\theta'_{ijk}}{\theta_{ijk} - \theta'_{ijk}} \ . \tag{.4}$$

Note that even though for the sake of convenience, we use single probability parameter x in the description of the theorems, the computation method in Theorem 10 and Lemma 11 applies to the sensitivity analysis of the same query with respect to all parameters without any other message propagation. Simply choose a host clique K of the owner variable of the probability parameter under investigation in the junction tree J. Then in Equation .1, replace ϕ_K and θ parameters with the right ϕ_K and the right probability distributions $\boldsymbol{\theta_{ij}}$ and $\boldsymbol{\theta'_{ij}}$. Using Theorem 10 and Lemma 11, we can immediately obtain the values of the corresponding coefficients with regard to the probability parameters.

Algorithm 2 Kjærulff and van der Gaag's Algorithm for Sensitivity Analysis[42]

```
(1) Enter evidence $e$ into the junction tree $J$ and perform an inward propagation towards a clique $K$ containing the variable of interest $X_t$;
(2) Perform an outward propagation from $K$;
(3) Compute $z$ and $z'$ using Equation .3;
(4) Compute the coefficients $\gamma$ and $\delta$ using Equation .4 for all relevant parameters, locally per clique;
(5) Perform another outward propagation from $K$ with additional evidence $X_t = x_t$;
(6) Compute $y$ and $y'$ using Equation .1;
(7) Compute the coefficients $\alpha$ and $\beta$ using Equation .2 for all relevant parameters, locally per clique;
```

It follows from Theorem 4 that all parameter sensitivities of a Bayesian network can be quantified by the coefficients of the sensitivity functions. To compute parameter sensitivity $S(x \rightarrow y|e)$, one only needs to bring the values of $\alpha, \beta, \gamma, \delta$, and $x = \theta_{ijk}$ in Equation 5.4 and solve the equation.

A.3 EFFICIENT SENSITIVITY ANALYSIS USING RELEVANCE REASONING

Algorithm 2 is computationally more efficient than other approaches to sensitivity analysis that require many message propagations and belief updates. However, propagation in a junction tree for an entire network, especially a network with large cliques, can still be computationally expensive and, for sufficiently large networks, infeasible. A critical factor in junction tree propagation is the size of the junction tree and especially the size of its cliques. This depends directly on the topology of the graph underlying the Bayesian network or, more precisely, to its connectivity. Generally, the exact inference algorithms, including message propagation, is subject to growth in complexity that is exponential in the size of the graph. And the worst case remains NP-hard [13].

Thus, applying methods that can reduce the size of junction trees or the size of cliques can improve the efficiency of sensitivity analysis and even make it computationally feasible. Given this observation, I adopted relevance-based decomposition technique [48] in junction tree framework for sensitivity analysis. The approach allows for propagation only on the relevant part of the Bayesian network with regard to the query targets given observed evidence. In this section, I review the basic concepts of relevance reasoning that will be useful in the algorithm, and present a multi-target sensitivity analysis algorithm based on relevance reasoning technique.

The concept of relevance is relative to the model, to the focus of reasoning, and to the context in which reasoning takes place [22]. In other words, relevance is identified in the graph G with regard to the query target nodes $\boldsymbol{T}$ (i.e., reasoning focus), and the evidence nodes $\boldsymbol{E}$ (i.e., reasoning context). The major relevance reasoning methods include *d-separation*, deleting *barren nodes*, and *network decomposition*.

A.3.1 D-separation

The fundamental tool of relevance reasoning in graphical models is a basic property known as the *d-separation* condition [19, 54] which ties the concept of conditional independence to the structure of the graph. In the directed acyclic graph, a node A can have three kinds of possible connections with other nodes.

1. *Serial connection* from B to C via A, e.g, $B \rightarrow A \rightarrow C$, evidence from B to C is blocked only when we have hard evidence about A.
2. *Diverging connection* where B and C have the common parent A, e.g., $B \leftarrow A \rightarrow C$,

evidence from B to C is blocked only when we have hard evidence about A.

3. *Converging connection* where A has parents B and C, e.g., $B \to A \leftarrow C$, any evidence about A results in evidence transmitted between B and C.

Definition 12 (D-separation)**.** *Two nodes X and Y in a Bayesian network are d-separated if, for all paths between X and Y, there is an intermediate node A for which either:*

1. the connection is serial or diverging and the state of A is known for certain; or

2. the connection is converging and neither A (nor any of its descendants) have received any evidence at all.

Informally, an evidence node blocks the propagation of information from its ancestors to its descendants. At the same time, it also makes all its ancestors interdependent. Removing those nodes that are probabilistically independent from the target nodes given the evidence nodes reduces the graph size and, therefore, allows for a faster message propagation in the corresponding smaller junction tree.

A.3.2 Barren Nodes

Removal of *barren nodes* [64] is another factor that contributes to reducing of the graph size for inference. Nodes are barren if they are neither evidence nor target and have no descendants, or all of their descendants are barren. Barren nodes may depend on the evidence, but they do not contribute to the change in probability of the target nodes. That is, barren nodes are computationally irrelevant with respect to the target nodes. Formally, we define the *barren node rule* as below:

Definition 13 (Barren Node Rule)**.** *Let $\boldsymbol{\psi}$ be a set of potentials in junction tree J for domain $\boldsymbol{X}$. Let $\boldsymbol{T} \subset \boldsymbol{X}$ be the target nodes and $\boldsymbol{E} \subset \boldsymbol{X}$ be evidence nodes, and let $\boldsymbol{\psi}^{\downarrow \boldsymbol{T}} = \sum_{\boldsymbol{X} \setminus \boldsymbol{T}} \boldsymbol{\psi}$ denote the result of marginalizing out all variables $\boldsymbol{X}$ except the members of $\boldsymbol{T}$. For a node $X_i \notin \boldsymbol{T}$ and $X_i \notin \boldsymbol{E}$, if the only potential in $\boldsymbol{\psi}$ with X_i is of the form $p(X_i|X_j)$, then X_i is barren and can be marginalized by discarding $p(X_i|X_j)$, i.e.,*

$$p(\boldsymbol{T}|\boldsymbol{E}) = \boldsymbol{\psi}^{\downarrow \boldsymbol{T}} = \sum_{\boldsymbol{X} \setminus \{X_i\} \setminus \boldsymbol{T}} \boldsymbol{\psi} \setminus \{p(X_i|X_j)\}$$

From the barren node rule, it is straightforward to see that barren nodes have no impact on computing posterior probabilities of target nodes given observed evidence. Therefore, barren nodes have zero sensitivity values.

Theorem 14. *Let X_i be a barren node in J with regard to a query y on target $\boldsymbol{T}$ given evidence scenario e for $\boldsymbol{E}$. And let $x \in \{\theta_{ijk}\}$ be a probability parameter owned by X_i. Then, $S(x \rightarrow y|e) = 0$.*

A.3.3 An Algorithm for Sensitivity Analysis Based on Relevance Reasoning

Decomposition is the essence of relevance reasoning that I used to speed up computation in Bayesian networks. Typically, decomposition is to divide a network B into several partially overlapping subnetworks, where each subnetwork is computationally relevant to the group of the target nodes, and all sets combined cover the entire network. This allows for performing the computation in the identified subnetworks with smaller junction trees and possibly smaller clique sizes.

The detailed decomposition algorithm can be found in Lin and Druzdzel's paper [48]. Here I outline the algorithm in the context of sensitivity analysis as below. For each of the target nodes T_i, I use relevance reasoning techniques described in the previous sections, mainly, d-separation and removing barren nodes, to identify a subnetwork that is relevant for computing T_i given evidence e. Then the Algorithm 2 is employed to compute the sensitivities for each of the probability parameters. The sensitivity value is updated when necessary to keep its maximum value as the measure of the parameter's sensitivity with regard to the target set $\mathbf{T}$.

The computational irrelevance holds true in most of the Bayesian network inference, especially with partial observation as evidence. When there are multiple target nodes, decomposition can speed up reasoning in many of the practical cases. In sensitivity analysis, computation is typically expensive because the large number of probability parameters in a Bayesian network requires hundreds and thousands of the coefficients to be established in the sensitivity functions. Usually the query to a network is on more than one target node, especially in case of diagnosis and trouble-shooting with multiple suspects of possible diseases and failures. In this case, for each target $T_i \in \boldsymbol{T}$, the nodes are pruned in B that are computationally irrelevant to updating $p(T_i|\boldsymbol{E})$. The size of the subnetwork for T_i can be much smaller than B when not all nodes in B are computationally relevant to T_i.

In the algorithm presented below, I select the cliques where the target nodes reside as root in junction tree for message propagation. The algorithm computes parameter sensitivity values by applying Algorithm 2 in the subnetworks. Because the algorithm deals with multiple targets, I choose for each parameter the maximum sensitivity value over all targets.

Algorithm 3 A Multi-Target Sensitivity Analysis Algorithm Using Relevance-Based Decomposition

```
**Input**: a Bayesian network $B = (G, \Theta)$, query $y$ on a set of target nodes $\boldsymbol{T}$, and evidence
scenario $e$ on evidence nodes $\boldsymbol{E}$
**Output**: Sensitivity value $S(x \to y|e)$ for each $x \in \Theta$

(1) Initialize $S(x \to y|e) = 0$ for all $x \in \Theta$;
**for** each target $T_i \in \boldsymbol{T}$, **do**
    (2) Prune the irrelevant nodes in $B$ with regard to $T_i$ given $e$, denote the resulted
    subnetwork $B_s$;
    (3) Establish the junction tree $J$ for $B_s$;
    (4) Set the host clique of $T_i$ as root of $J$;
    (5) Apply Algorithm 2 on $J$ to compute the sensitivity coefficients;
    (6) Compute $S(x \to t_i|e)$ for the relevant parameters $x$ using Equation 5.4;
    **if** $S(x \to t_i|e) > S(x \to y|e)$ **then**
        (7) Update $S(x \to y|e) = S(x \to t_i|e)$;
    **end if**
**end for**
```

Table A1: Summary Of The Tested Networks

	#nodes	**#parameters**	**#targets**	**#evidence**	**ave#findings**
ALARM	37	752	7	16	11.94
HEPAR II	72	2139	9	61	42.08
OIL	38	312	13	7	4.86
PNEUMATIC	56	2786	16	17	11.64
CPCS	179	17413	8	74	49.8

A.4 EMPIRICAL RESULTS

I implemented sensitivity analysis algorithm using relevance reasoning as described in Algorithm 3. To show the performance difference due to relevance reasoning technique, I tested Algorithm 3 against Algorithm 2 in computation of parameter sensitivities for comparison. The implementation of the junction tree algorithm was identical for the two algorithms except that Algorithm 3 has the relevance-based decomposition turned on.

The test bed consisted five real-domain Bayesian networks: ALARM, HEPAR II, OIL, PNEUMATIC and CPCS. The ALARM [1] network is a model for diagnosing potential anesthesia problems in operating rooms. ALARM has totally 37 nodes. Among them, 7 nodes represent 8 diagnostic problems and 16 represent medical findings such as patient symptoms, signs and laboratory test results. HEPAR II [53] network is a model for diagnosing liver disorders in a clinical setting. It contains 11 different liver diseases (represented by 9 nodes) and 61 medical findings. OIL and PNEUMATIC [41] are two Bayesian network models built in The Boeing Company for diagnosis of 737 airplane oil and pneumatic systems respectively. In these two models, the diagnostic targets are the relevant Line Replaceable Units (LRUs). When a LRU fails, it triggers certain events visible to pilots in the cockpit which are called Flight Deck Effects (FDEs), or other perceived anomalies such as abnormal sounds, smells, or visible cues (e.g, smoke in the cabin). In the OIL and PNEUMATIC models, the FDEs are the main observables used for troubleshooting. The OIL model has 13 LRUs, 7 FDEs and 18 intermediate nodes. In the PNEUMATIC model, there are totally 56 nodes, among which, 16 are LRUs and 17 are FDEs. In addition, we tested the algorithms on the CPCS (Computer based Patient Case Study) network [57] that was built for a subset of the domain of internal medicine. The version of CPCS that we used in our test consists totally 179 nodes [1]. In this version of CPCS, 8 nodes describe internal diseases and 74 nodes describe possible predispositors and symptoms.

For each of the networks, I used diagnostic targets (i.e., diseases or LRUs) as the target nodes for relevance reasoning. For each observable or finding node (i.e., a symptom, a FDE), I randomly chose its possible evidential and ignorant state, i.e., either an observation of its possible states, or simply unknown state when there is no observation on this node at all. Therefore, each evidence case is an n-dimentional array that contains such simulated states for all n observable nodes. I generated a total of 50 test cases for each of the networks in our experiment.

The networks in the test contain only discrete nodes. Table A1 summarizes some attributes of these networks that may influence the run time of sensitivity analysis algorithms. In the table, the "#parameters" column gives the total number of probability parameters in the network, the "ave#findings" column gives the average number of the simulated evidential states of the observable nodes.

I ran both sensitivity analysis algorithms to compute all the probability parameter sensitivities with the generated test cases input as evidence. I recorded the run time of the algorithms on each test case. The host computer was a Dell Latitude laptop with 1.2MHz CPU and 512MB memory running Windows 2000. The test results for individual cases on HEPAR II model and ALARM are presented in Figure A1 and Figure A2. Algorithm 3 (denoted in the figure by "w/ RR") runs typically orders of magnitude faster than Algorithm 2 (denoted in the figure by "w/o RR"). Table A2 lists the mean time and standard deviation as summary on all test runs for each of the tested networks. Again, our algorithm with relevance-based decomposition has considerably better performance on all of the tested

[1]The full CPCS network we have does not have meaningful diagnostic definition available. It consists more than 400 nodes and is considered typically large network in real-domain applications.

Table A2: Summary Of The Simulation Results For The Tested Networks

		w/ RR	**w/o RR**	**t-test**
ALARM	μ	0.0031	0.0251	4.9323
	σ	0.0006	0.0050	$*10^{-36}$
HEPAR II	μ	0.0020	0.0659	2.1969
	σ	0.0010	0.0406	$*10^{-15}$
OIL	μ	0.0004	0.0172	1.9220
	σ	0.0006	0.0039	$*10^{-35}$
PNEUMATIC	μ	0.0117	1.0267	6.56
	σ	0.0181	1.7251	$*10^{-5}$
CPCS	μ	0.0026	29.4158	2.12
	σ	0.0018	29.1870	$*10^{-9}$

networks.

Apparently, the total run time of Algorithm 2 on the PNEUMATIC model and the CPCS network is much longer than the run time of other models in our test (see Figure A3). Note that in Figure A3 and Figure A4, we used log scale of the run time to show the large differences in run time in one picture. Figure A4 shows the averaged run time per parameter in the tested networks for both algorithms. From the figure, we see that, for Algorithm 2, the averaged run time of PNEUMATIC and CPCS are still longer than the average run time of other networks. This confirms that the major reason for performance of sensitivity analysis algorithm in junction tree reasoning framework is not only the number of parameters but also the structure topology. The reason for a slow run on CPCS is obvious because of its very large scale. The cause for a slow run on PNEUMATIC may be the network topology which induces large clique sizes.

But the performance of Algorithm 3 is steadily faster than its counterpart on all of the networks. This result proves that pruning irrelevant part of the networks and propagating messages only in the relevant subnetworks with regard to the query targets provides signif-

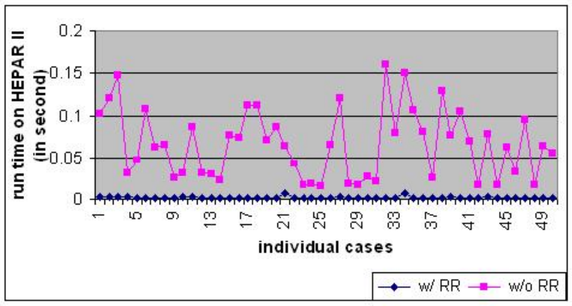


Figure A1: Run Time Of The Sensitivity Analysis Algorithms On Individual Test Cases Of Hepar II

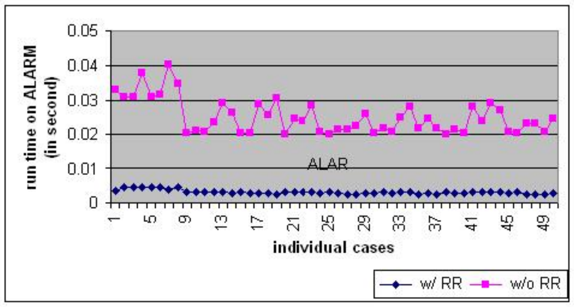


Figure A2: Run Time Of The Sensitivity Analysis Algorithms On Individual Test Cases Of Alarm

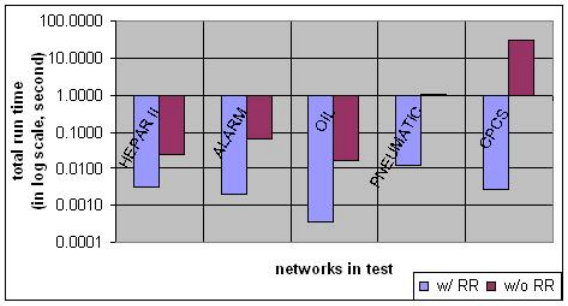


Figure A3: Total Run Time Of The Sensitivity Analysis Algorithms With/without Relevance Reasoning

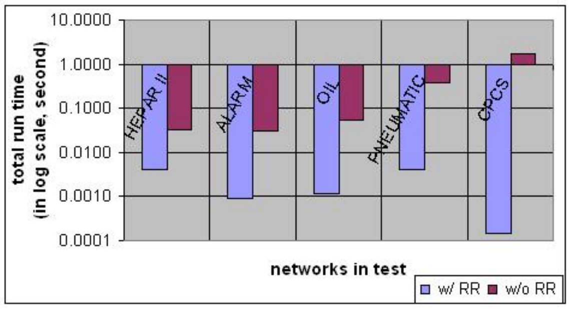


Figure A4: Run Time Per Parameter Of The Sensitivity Analysis Algorithms With/without Relevance Reasoning

icant improvement on sensitivity analysis algorithms in junction tree framework. Table A2 shows the results of paired, one-tailed t-tests testing the hypotheses that the relevance-based decomposition-based algorithm is as fast as the original algorithm. The hypothsis can be rejected at a very low significance level (on the order of 10^{-36}, 10^{-15}, 10^{-35}, 10^{-5} and 10^{-9}).

A.5 DISCUSSION

I developed an efficient sensitivity analysis algorithm using relevance reasoning for Bayesian networks. The algorithm is based on Kjærulff and van der Gaag's algorithm using clustering inference in junction tree framework. I added relevance-based decomposition to the inference algorithm to speed up sensitivity analysis. I performed an empirical test on five real Bayesian networks in comparison of the performance of the two algorithms. The test results show that the relevance reasoning technique is in general very useful for faster sensitivity analysis.

I noted that Coupe and van der Gaag [16] present a method to preprocess a Bayesian network and identify a sensitivity set based on the d-separation criterion. Using their method can save the computational cost on the calculation of sensitivity coefficients for the irrelevant probability parameters given specific evidence and targets. But the junction trees were built for the entire Bayesian networks, as if all of the probability parameters are relevant. Propagation in such junction trees does not take the advantage of the identification of the sensitivity set. Therefore, the computation for the sensitivity coefficients of relevant parameters is slower than the computation based on the decomposed, smaller junction trees.

BIBLIOGRAPHY

[1] I.A. Beinlich, H.J. Suermondt, R.M. Chavez, and G.F. Cooper. 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 Medical Care*, pages 247–256, London, 1989.

[2] Craig Boutilier, Nir Friedman, Moises Goldszmidt, and Daphne Koller. Context-specific independence in Bayesian networks. In *Proceedings of the Twelfth Annual Conference on Uncertainty in Artificial Intelligence (UAI–96)*, pages 115–123, San Francisco, CA, 1996. Morgan Kaufmann Publishers.

[3] W. Buntine. Theory refinement on Bayesian networks. In *Proceedings of the Seventh Annual Conference on Uncertainty in Artificial Intelligence (UAI–91)*, pages 52–60, San Mateo, California, 1991. Morgan Kaufmann Publishers.

[4] C. Melody Carswell, Sylvia Frankenberger, and Donald Bernhard. Graphing in depth: perspectives on the use of three-dimensional graphs to represent lower-dimensional data. *Behavior & Information Technology*, pages 459–474, 1991.

[5] Enrique F. Castillo, José Manuel Gutiérrez, and Ali S. Hadi. Sensitivity analysis in discrete Bayesian networks. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 27(4):412–423, 1997.

[6] Hei Chan and Adnan Darwiche. When do numbers really matter? In *Uncertainty in Artificial Intelligence: Proceedings of the Seventeenth Conference (UAI-2001)*, pages 65–74, San Francisco, CA, 2001. Morgan Kaufmann Publishers.

[7] Hei Chan and Adnan Darwiche. A distance measure for bounding probabilistic belief change. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence and Fourteenth Conference on Innovative Applications of Artificial Intelligence (AAAI/IAAI-02)*, pages 539–545, Menlo Parc, CA, USA, July 28– August 1 2002. AAAI Press.

[8] J. Cheng, D.A. Bell, and W. Liu. Learning belief networks from data: An information theory based approach. In *Proceedings of the Sixth ACM International Conference on Information and Knowledge Management (CIKM–97)*, 1997.

[9] Jie Cheng, Russell Greiner, and Jonathan Kelly. Learning Bayesian networks from data: An information-theory based approach. *Artificial Intelligence*, 137:43–90, 2002.

[10] David Maxwell Chickering. A transformational characterization of equivalent Bayesian network structures. In *Proceedings of the Eleventh Annual Conference on Uncertainty in Artificial Intelligence (UAI–95)*, pages 87–98, San Francisco, CA, 1995. Morgan Kaufmann Publishers.

[11] W.S Cleveland and R. McGill. Graphical perception: Theory, experimentation, and application to the development of graphical methods. *Journal of the American Statistical Association*, 79:531–554, 1984.

[12] Roger Cooke. *Experts in Uncertainty: Opinion and Subjective Probability in Science.* Oxford University Press, 1991.

[13] Gregory F. Cooper. The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, 42(2–3):393–405, March 1990.

[14] Gregory F. Cooper and Edward Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9(4):309–347, 1992.

[15] Veerle M.H. Coupé, Niels Peek, Jaap Ottenkamp, and J. Dik F. Habbema. Using sensitivity analysis for efficient quantification of a belief network. *Artificial Intelligence in Medicine*, 17:223–247, 1999.

[16] Veerle M.H. Coupé and Linda C. van der Gaag. Properties of sensitivity analysis of Bayesian belief networks. *Annals of Mathematics and Artificial Intelligence*, 36:323–356, 2002.

[17] Robert G. Cowell. Conditions under which conditional independence and scoring methods lead to identical selection of bayesian network models. In *Uncertainty in Artificial Intelligence: Proceedings of the Seventeenth Conference (UAI-2001)*, pages 91–97, San Francisco, CA, 2001. Morgan Kaufmann Publishers.

[18] Adnan Darwiche. A differential approach to inference in Bayesian networks. In *Uncertainty in Artificial Intelligence: Proceedings of the Sixteenth Conference (UAI-2000)*, pages 123–132, San Francisco, CA, 2000. Morgan Kaufmann Publishers.

[19] A. Philip Dawid. Conditional independence in statistical theory. *Journal of the Royal Statistical Society, Series B (Methodological)*, 41:1–31, 1979.

[20] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm (with discussion). *Journal of the Royal Statistical ciety, Series B*, 39:1–38, 1977.

[21] Marek J. Druzdzel. SMILE: Structural Modeling, Inference, and Learning Engine and GeNIe: A development environment for graphical decision-theoretic models. In *Proceed-*

ings of the Sixteenth National Conference on Artificial Intelligence (AAAI–99), pages 902–903, Orlando, FL, July 18–22 1999.

[22] Marek J. Druzdzel and Henri J. Suermondt. Relevance in probabilistic models: "Backyards" in a "small world". In *Working notes of the AAAI–1994 Fall Symposium Series: Relevance*, pages 60–63, New Orleans, LA (An extended version of this paper is in preparation.), 4–6 November 1994.

[23] Marek J. Druzdzel and Linda C. van der Gaag. Elicitation of probabilities for belief networks: Combining qualitative and quantitative information. In *Proceedings of the Eleventh Annual Conference on Uncertainty in Artificial Intelligence (UAI–95)*, pages 141–148, San Francisco, CA, 1995. Morgan Kaufmann Publishers, Inc.

[24] Marek J. Druzdzel and Linda 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, July–August 2000.

[25] W.K. Estes. The cognitive side of probability learning. *Psychological Review*, 83(1):37–64, January 1976.

[26] Deb Feldman-Stewart, Nancy Kocovski, Beth A. McConnell, Machael D. Brundage, and William J. Mackillop. Perception of quantitative information for treatment decisions. *Medical Decision Making*, 20:228–238, 2000.

[27] Dan Geiger and David Heckerman. A characterization of the Dirichlet distribution with application to learning Bayesian networks. In *Proceedings of the Eleventh Annual Conference on Uncertainty in Artificial Intelligence (UAI–95)*, pages 196–207, San Francisco, CA, 1995. Morgan Kaufmann Publishers.

[28] Dan Geiger and David Heckerman. Knowledge representation and inference in similarity networks and Bayesian multinets. *Artificial Intelligence*, 82:45–74, 1996.

[29] David Heckerman. A tutorial on learning with Bayesian networks. In Michael I. Jordan, editor, *Learning in Graphical Models*. The MIT Press, Cambridge, Massachusetts, 1998.

[30] David Heckerman, Dan Geiger, and David M. Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20:197–243, 1995.

[31] David Heckerman, E.J. Horvitz, and B.N. Nathwani. Toward normative expert systems: Part I. the Pathfinder project. *Methods of Information in Medicine*, 31:90–105, 1992.

[32] Max Henrion. Propagating uncertainty in Bayesian networks by probabilistic logic sampling. In L.N. Kanal, T.S. Levitt, and J.F. Lemmer, editors, *Uncertainty in Artificial Intellgience 2*, pages 149–163. Elsevier Science Publishing Company, Inc., New York, N. Y., 1988.

[33] Max Henrion. Some practical issues in constructing belief networks. In L.N. Kanal, T.S. Levitt, and J.F. Lemmer, editors, *Uncertainty in Artificial Intelligence 3*, pages 161–173. Elsevier Science Publishing Company, Inc., New York, N. Y., 1989.

[34] Max Henrion, Malcolm Pradhan, Brendan Del Favero, Kurt Huang, Gregory Provan, and Paul O'Rorke. Why is diagnosis using belief networks insensitive to imprecision in probabilities? In *Proceedings of the Twelfth Annual Conference on Uncertainty in Artificial Intelligence (UAI–96)*, pages 307–314, San Francisco, CA, 1996. Morgan Kaufmann Publishers.

[35] J.G. Hollands and Ian Spence. Judgements of change and proportion in graphical perception. *Human Factors*, 34:313–334, 1992.

[36] J.G. Hollands and Ian Spence. Judging proportion with graphs: The summation model. *Applied Cognitive Psychology*, 12:173–190, 1998.

[37] Stephen C. Hora, Judith A. Hora, and Nancy G. Dodd. Assessment of probability distributions for continuous random variables: A comparison of the bisection and fixed value methods. *Organizational Behavior and Human Decision Processes*, 51:133–155, 1992.

[38] Ronald A. Howard and James E. Matheson. Influence diagrams. In Ronald A. Howard and James E. Matheson, editors, *The Principles and Applications of Decision Analysis*, pages 719–762. Strategic Decisions Group, Menlo Park, CA, 1981.

[39] Daniel Kahneman, Paul Slovic, and Amos Tversky, editors. *Judgment Under Uncertainty: Heuristics and Biases.* Cambridge University Press, Cambridge, 1982.

[40] Oscar Kipersztok and Haiqin Wang. Another look at sensitivity of Bayesian networks to imprecise probabilities. In *Proceedings of the Eighth International Workshop on Artificial Intelligence and Statistics (AISTAT-2001)*, pages 226–232, San Francisco, CA, 2001. Morgan Kaufmann Publishers.

[41] Oscar Kipersztok and Haiqin Wang. Validation of diagnostic models using graphical belief networks. In *Intelligent Systems for Information Processing from Representation to Applications*, pages 443–452, North Holland, 2003. Elsevier Publishers.

[42] Uffe Kjærulff and Linda C. van der Gaag. Making sensitivity analysis computationally efficient. In *Uncertainty in Artificial Intelligence: Proceedings of the Sixteenth Conference (UAI-2000)*, pages 317–325, San Francisco, CA, 2000. Morgan Kaufmann Publishers.

[43] S. Kullback and R. A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22:79–86, 1951.

[44] Wai Lam and Fahiem Bacchus. Learning Bayesian belief networks: An approach based on the MDL principle. *Computational Intelligence*, 10(3):269–293, 1994.

[45] Kathryn Blackmond Laskey. Sensitivity analysis for probability assessments in Bayesian networks. *IEEE Transactions on Systems, Man, and Cybernetics*, 25(6):901–909, 1995.

[46] S. Lichtenstein, B. Fischhoff, and L.D. Philips. Calibration of probabilities: The state of the art to 1980. In *Judgement under uncertainty: Heuristics and biases.* Cambridge University Press, Cambridge, 1982.

[47] S. Lichtenstein, P. Slovic, B. Fischhoff, M. Layman, and B. Combs. Judged frequency of lethal events. *Journal of Experimental Psychology: Human Learning and Memory*, 4(6):551–578, November 1978.

[48] Yan Lin and Marek J. Druzdzel. Computational advantages of relevance reasoning in Bayesian belief networks. In *Proceedings of the Thirteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI–97)*, pages 342–350, San Francisco, CA, 1997. Morgan Kaufmann Publishers, Inc.

[49] Urbano Adolfo López Gómez. *Communicating very Low Probability Events.* PhD thesis, Department of Engineering and Public Policy, Carnegie Mellon University, Pittsburgh, PA, May 1990.

[50] Miley W. Merkhofer. Quantifying judgmental uncertainty: Methodology, experiences, and insights. *IEEE Transactions on Systems, Man, amd Cybernetics*, SMC-17(5):741–752, 1987.

[51] M. Granger Morgan and Max Henrion. *Uncertainty: A Guide to Dealing with Uncertainty in Quantitative Risk and Policy Analysis.* Cambridge University Press, Cambridge, 1990.

[52] Agnieszka Oniśko and Marek J. Druzdzel. Effect of imprecision in probabilities on Bayesian network models: An empirical study. In Peter Lucas, editor, *Working Notes of the workshop on Model-based and Qualitative Reasoning in Biomedicine, held during the 9th European Conference on Artificial Intelligence in Medicine, AIME–03*, pages 45–49, Protaras, Cyprus, 2003. Springer.

[53] Agnieszka Oniśko, Marek J. Druzdzel, and Hanna Wasyluk. Extension of the Hepar II model to multiple-disorder diagnosis. In S.T.W̃ierzchoń M.K̃lopotek, M.M̃ichalewicz, editor, *Intelligent Information Systems,* Advances in Soft Computing *Series*, pages 303–313, Heidelberg, 2000. Physica-Verlag (A Springer-Verlag Company).

[54] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1988.

[55] Judea Pearl. *Causality: Models, Reasoning, and Inference.* Cambridge University Press, Cambridge, UK, 2000.

[56] Malcolm Pradhan, Max Henrion, Gregory Provan, Brendan del Favero, and Kurt Huang. The sensitivity of belief networks to imprecise probabilities: An experimental investigation. *Artificial Intelligence*, 85(1–2):363–397, August 1996.

[57] Malcolm Pradhan, Gregory Provan, Blackford Middleton, and Max Henrion. Knowledge engineering for large belief networks. In *Proceedings of the Tenth Annual Conference on Uncertainty in Artificial Intelligence (UAI–94)*, pages 484–490, San Mateo, CA, 1994. Morgan Kaufmann Publishers, Inc.

[58] J. Preece, Helen Sharp, David Benyon, Simon Holland, and Tom Carey. *Human-Computer Interaction.* Addison-Wesley, New York, 1994.

[59] Paul C. Price. Effects of a relative-frequency elicitation question on likelihood judgment accuracy: The case of external correspondence. *Organizational Behavior and Human Decision Processes*, 76(3):277–297, 1998.

[60] P. Rabbitt. The control of attention in visual search. In R. Parasuvaman and D.R. Davis, editors, *Varieties of Attention.* Academic Press, New York, 1984.

[61] Silja Renooij and Linda van der Gaag. Evidence-invariant sensitivity bounds. In Max Chichering and Joseph Halpern, editors, *Proceedings of the Twentieth Conference on Uncertainty in Artificial Intelligence*, pages 479–486, Arlington,VA, 2004. AUAI Press.

[62] J. Rissanen. Modeling by shortest data description. *Automatica*, 14:465–471, 1978.

[63] G. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6:461–464, 1978.

[64] Ross D. Shachter. Evaluating influence diagrams. *Operations Research*, 34(6):871–882, November–December 1986.

[65] Solomon E. Shimony. The role of relevance in explanation I: Irrelevance as statistical independence. *International Journal of Approximate Reasoning*, 8(4):281–324, June 1993.

[66] Michael Siegrist. The use or misuse of three-dimensional graphs to represent lower-dimensional data. *Behavior & Information Technology*, 15(2):96–100, 1996.

[67] D. Simkin and R. Hastie. An information processing analysis of graph perception. *Journal of the American Statistical Association*, 82:454–465, 1987.

[68] John A. Sparrow. Graphical displays in information systems: Some data properties influencing the effectiveness of alternative forms. *Behaviour & Information Technology*, 8(1):43–56, 1989.

[69] Ian Spence. Visual psychophysics of simple graphical elements. *Journal of Experimental Psychology: Human Perception and Performance*, 16:683–692, 1990.

[70] C. Spetzler and C-A.S. Staël von Hostein. Probability encoding in decision analysis. *Management Science*, 22:340–358, 1975.

[71] Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction, and Search.* Springer Verlag, New York, 1993.

[72] Edward R. Tufte. *The Visual Display of Quantitative Information.* Graphics Press, Cheshire, Connecticut, 16th edition, 1998.

[73] Linda van der Gaag, Silja Renooij, Cilia Witteman, Berthe Aleman, and Babs Taal. How to elicit many probabilities. In *Proceedings of the Fifteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI–99)*, pages 647–654, San Francisco, CA, 1999. Morgan Kaufmann Publishers.

[74] Linda C. van der Gaag and Veerle M.H. Coupé. Sensitivity analysis for threshold decision making with Bayesian belief networks. In E. Lamma and P. Mello, editors, *AI*IA 99: Advances in Artificial Intelligence, Lecture Notes in Artificial Intelligence*, pages 37 – 48. Springer-Verlag, Berlin, 2000.

[75] T.S. Verma and Judea Pearl. Equivalence and synthesis of causal models. In P.P. Bonissone, M. Henrion, L.N. Kanal, and J.F. Lemmer, editors, *Uncertainty in Artificial Intelligence 6*, pages 255 –269. Elsevier Science Publishing Company, Inc., New York, N. Y., 1991.

[76] Haiqin Wang, Denver Dash, and Marek Druzdzel. A method for evaluating elicitation schemes for probabilistic models. *IEEE Transactions on Systems, Man, and Cybernetics–Part B:Cybernetics*, 32(1):38–43, February 2002.

[77] Haiqin Wang and Marek J. Druzdzel. User interface tools for navigation in conditional probability tables and graphical elicitation of probabilities in Bayesian networks. In *Uncertainty in Artificial Intelligence: Proceedings of the Sixteenth Conference (UAI-2000)*, pages 617–625, San Francisco, CA, 2000. Morgan Kaufmann Publishers.

[78] Haiqin Wang, Irina Rish, and Sheng Ma. Using sensitivity analysis for selective parameter update in Bayesian network learning. In Kai Goebel and Piero Bonissone, editors, *Information Refinement and Revision for Decision Making: Modeling for Diagnostics,Prognostics and Prediction*, AAAI 2002 Spring Symposium, Technical Report SS-02-03, pages 29–36, Menlo Park, CA, March 2002. AAAI Press.

[79] J.C. Watts. Navigation in the computer medium: A cognitive analysis. In *Proceedings of the Human Factors and Ergonomics Society, 38th Annual Meeting*, Santa Monica, CA, 1994. Human Factors and Ergonomics Society.

[80] D.D. Woods. A concept to improve the cognitive coupling of person and computer. *International Journal of Man-Machine Studies*, 21:229–244, 1984.