An Educational Tool for High-Level Interaction with Bayesian Networks

Peter Haddawy
Joel Jacobson

Department of EE & CS
University of Wisconsin–Milwaukee
Milwaukee, WI 53201
{haddawy, jake}@cs.uwm.edu

Charles E. Kahn, Jr.

Department of Radiology
Medical College of Wisconsin
Milwaukee, WI 53226
ckahn@mcw.edu

Abstract

We present an educational tool for bringing the information contained in a Bayesian network to the end user in an easily intelligible form. The BANTER *shell is designed to tutor users in evaluation of hypotheses and selection of optimal diagnostic procedures.* BANTER *can be used with any Bayesian network containing nodes that can be classified into hypotheses, observations, and diagnostic procedures. We present algorithms for determining optimal diagnostic procedures and for explanation generation.*

1 Introduction

In recent years Bayesian belief networks have become the representation of choice for building decision-making systems in domains characterized by uncertainty. The popularity of Bayesian networks has lead to a proliferation of domain models covering diverse areas [1, 3, 6, 5]. While most applications of Bayesian networks are geared toward providing decision support, the large number of models both currently available and under development provides a wealth of detailed knowledge that could be used for educational purposes as well. Unfortunately, the information contained in these models is not easily intelligible to humans. A vehicle is required to make this information accessible for teaching purposes. The availability of expert system shells for performing inferences over Bayesian network models [2, 8], as well as the recent development of explanation generation algorithms [9] have now made building such a vehicle possible.

We present a generic Bayesian network-based tutoring shell, BANTER, which is designed to tutor users in diagnosis and in selection of optimal diagnostic procedures. BANTER can be used with any Bayesian network containing nodes that can be classified into hypotheses, observations, and diagnostic procedures. The system is designed so that the user need know nothing about Bayesian networks in order to effectively interact with it. In fact, nothing in the way the system interacts with the user would even indicate that the system is using a Bayesian network to perform its reasoning. The user needs only some knowledge of the particular domain and an elementary understanding of probability. BANTER provides the capability to

- compute the posterior probability of a hypothesis,
- determine the best diagnostic procedure to affirm ("rule in") or exclude ("rule out") a hypothesis,
- quiz the user in the selection of optimal diagnostic procedures, and
- explain the system's reasoning.

Since almost all of the system's reasoning is driven by the Bayesian network knowledge base, setting up the system to work with a new network requires only minimal effort.

2 System Capabilities

We will illustrate the capabilities of BANTER with a Bayesian network model of gallbladder disease. When using BANTER in medical domains, the hypotheses are the primary diagnoses, the observations are divided into patient history and physical findings, and the diagnostic procedures are the various available tests. The belief-network model shown in figure 1 was developed to analyze the effectiveness of various diagnostic

1063-6730/94 $4.00 © 1994 IEEE

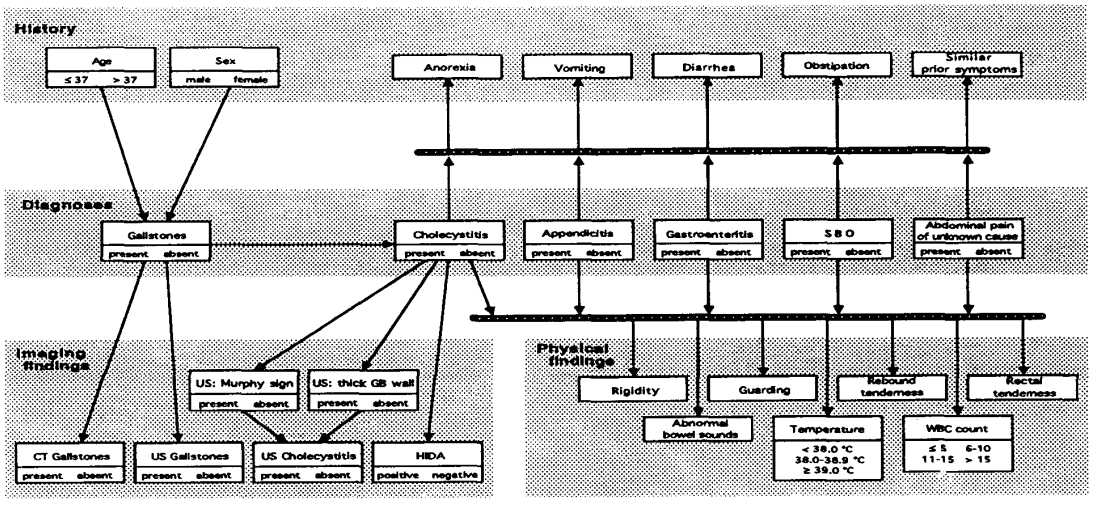

Figure 1: Bayesian-network model of gallbladder disease. The horizontal bars simplify the figure: all five nodes that lead to the bars influence all of the nodes that lead from the bars.

imaging procedures for the diagnosis of gallstones and cholecystitis in patients with acute abdominal pain. The sources of data used to construct the model and other details are described in [5].

In this model, the two principal diagnoses are gallstones and cholecystitis (inflammation of the gallbladder). Four diagnoses serve as alternative causes of acute abdominal pain: appendicitis, gastroenteritis, small bowel obstruction (SBO), and abdominal pain of unknown cause. The presence of gallstones influences the probability that cholecystitis is present. The remaining nodes represent a patient's history, physical findings, and test results. The patient history consists of age and sex, which are demographic factors influencing the presence of gallstones, and the patient's reports of anorexia, vomiting, diarrhea, obstipation, and "similar prior symptoms." The physical findings are rigidity, guarding, rebound tenderness, rectal tenderness, abnormal bowel sounds, temperature, and white blood cell (WBC) count. The possible states of each node are shown in the lower half of each box in figure 1; where not listed, the states are "present" and "absent." The model contains two imaging tests for gallstones: ultrasound and computed tomography (CT, or "CAT scan"). Three imaging tests are included for cholecystitis: the sonographic Murphy sign (maximal tenderness upon gallbladder compression during ultrasound examination), thickened gallbladder wall by ultrasound, and radionuclide hepatobiliary imaging ("HIDA").

Given such a network and a specification of which nodes represent history, physical findings, and diseases, BANTER works in three basic modes: query the knowledge base, quiz the user, and explain reasoning. These functions are provided through a graphical interface as shown in figure 2. Only the two principal diagnoses, gallstones and cholecystitis, are listed in the disease menu. The choice of which diseases to display is made when the system is configured for a given network. In this case, we display only the principal diagnoses since the available tests are not useful for diagnosing the four alternative diagnoses contained in the network.

2.1 Querying the Knowledge Base

The user queries the knowledge base by setting up a scenario. A scenario is created by specifying a set of known values for the history and physical findings, as well as a set of diseases of interest. This is done by clicking on nodes in windows displaying for history, physical findings, and diseases of interest. In figure 2 the user has entered the following scenario. A 41-year-old woman presents with anorexia and acute abdominal pain; she denies vomiting, diarrhea, obstipation or similar previous symptoms. Guarding is present with no rigidity or rebound tenderness. There were no abnormal bowel sounds. Her white blood count is 12.6 cells/cm^3. We are interested in diagnosing the presence of gallstones.

The user now can ask the system to compute the posterior probability of the selected diseases or to de-

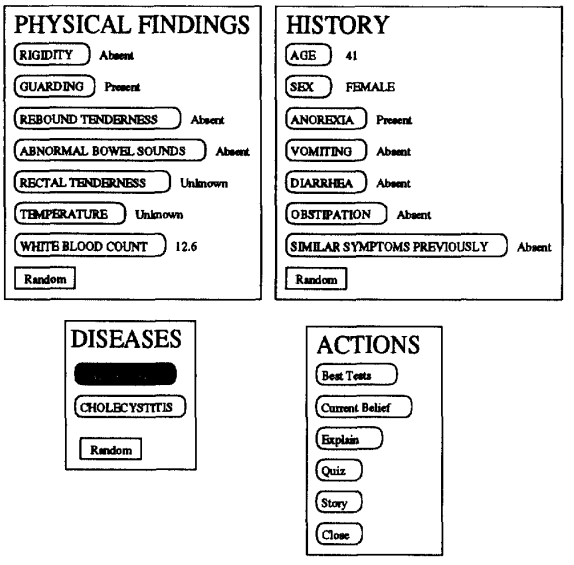

Figure 2: BANTER graphical interface.

termine the best tests to rule in and rule out the selected diseases. Requesting the system to compute the best test produces the following response.

```
The best test to rule in GALLSTONES is CT.
The best test to rule out GALLSTONES is
ULTRASOUND FOR GALLSTONES.
```

2.2 Requesting an Explanation

The user can obtain an explanation of the reasoning that lead the system to select these tests simply by selecting the "explain" button in the actions menu. The system starts by explaining how the known history and physical findings influence the probability of gallstones.

```
Before presenting any evidence, the prob of
GALLSTONES being present is 0.128.

The following pieces of evidence are
considered important:
Presence of GUARDING results in a post-test
prob of 0.175 on GALLSTONES.
AGE of 41 results in a post-test prob of
0.172 on GALLSTONES.

Their influence flows along the paths:
GUARDING → CHOLE → GALLSTONES
AGE → GALLSTONES

Presentation of the evidence results in a
posterior prob of 0.227 for the presence of
GALLSTONES.
```

Having explained how the pretest probability of gallstones was arrived at, the system continues by explaining how each possible test further influences the probability of gallstones.

```
The best tests to rule in GALLSTONES:
A positive CT test results in a prob of
0.987 on GALLSTONES.
A positive ULTRASOUND FOR GALLSTONES test
results in a prob of 0.601 on GALLSTONES.
A positive HIDA test results in a prob of
0.406 on GALLSTONES.
A positive ULTRASOUND FOR CHOLE test results
in a prob of 0.344 on GALLSTONES.

Their influence flows along the paths:
GALLSTONES → CT
GALLSTONES → ULTRASOUND FOR GALLSTONES
GALLSTONES → CHOLE → HIDA
GALLSTONES → CHOLE → SONOGRAPHIC MURPHY
   SIGN → ULTRASOUND FOR CHOLE
GALLSTONES → CHOLE → ULTRASOUND THICK GB
   WALL → ULTRASOUND FOR CHOLE

The best tests to rule out GALLSTONES:
A negative ULTRASOUND FOR GALLSTONES test
results in a prob of 0.016 on GALLSTONES.
A negative CT test results in a prob of
0.058 on GALLSTONES.
A negative HIDA test results in a prob of
0.176 on GALLSTONES.
A negative ULTRASOUND FOR CHOLE test results
in a prob of 0.183 on GALLSTONES.

Their influence flows along the paths:
GALLSTONES → ULTRASOUND FOR GALLSTONES
GALLSTONES → CT
GALLSTONES → CHOLE → HIDA
GALLSTONES → CHOLE → SONOGRAPHIC MURPHY
   SIGN → ULTRASOUND FOR CHOLE
GALLSTONES → CHOLE → ULTRASOUND THICK GB
   WALL → ULTRASOUND FOR CHOLE
```

If the user requests an explanation after asking the system to compute the posterior probability of selected diseases, only the first part of the explanation above

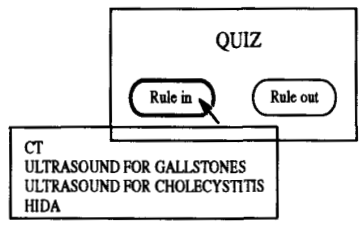

Figure 3: Quiz menu.

is generated.

2.3 Quizzing the User

In addition to asking the system to perform computations, the user can ask to be quizzed in the selection of optimal diagnostic procedures. This can be done in two ways. By clicking on the "quiz" button the user obtains the menu shown in figure 3. He can then specify a scenario and choose the test he thinks best to rule in or rule out the selected disease. When the user clicks on "test" the system checks his answers. If an answer is incorrect, the system tells the user which tests are preferable to the one he selected. He can then request an explanation and the same type of explanation as above will be generated.

The second way the user can be quizzed is by selecting the "story" action. In this mode, the system randomly selects a patient history, a set of physical findings, and a disease of interest, and presents the scenario to the user as a case in English. Below is one story the system generated.

```
Mrs. Jones is 36 years old, and presents
with OBSTIPATION, SIMILAR-SX-PREVIOUSLY, and
denies ANOREXIA, VOMITING.
Her temperature is 38.30. Her white-blood
count is 13.807. Physical examination
reveals RIGIDITY, and no evidence of
REBOUND-TENDERNESS, ABNORMAL-BOWEL-SOUNDS.

What is the best test to perform to rule out
GALLSTONES?
```

At this point, the user can select his answer from the quiz menu and continue as in the other quiz mode described above.

3 Algorithms

3.1 Determining the Best Test

Given some known physical findings and patient history data, the best test to rule in or rule out a hypothesis is determined by positively and negatively instantiating each test outcome and determining the posterior probability of the hypothesis given the test outcome (posttest probability). The best test to rule in the hypothesis is the one that results in the highest postest probability and the best test to rule out the hypothesis is the one that results in the lowest posttest probability. If the user selects more than one hypothesis of interest, this procedure is performed for each of the selected hypotheses.

3.2 Explanation Generation

Following Suermondt's INSITE method [9], explanation generation consists of two procedures. The first one identifies those pieces of evidence that had the most influence on the probability of a given hypothesis. The second takes a set of evidence nodes and a hypothesis node and identifies the strongest and most comprehensible paths linking each evidence node with the hypothesis node. These algorithms are used in two different modes of the system. In the example of the previous section, both algorithms are used to explain the current belief in a disease. We first identify those nodes among the specified history and physical findings that were most influential in producing the reported posterior probability of the disease and then find the paths along which that influence flowed. In generating explanations for the selection of the best test, only the second algorithm is used. Here we already know that we're interested in determining the influence of each test outcome on the disease, so we only need to find the paths of influence.

3.2.1 Identifying Influential Pieces of Evidence

To identify the most influential pieces of evidence, we first determine the influence of each evidence node on a hypothesis, by performing a sensitivity analysis. We remove all evidence from the network and then instantiate each evidence node individually and record the posterior probability of the hypothesis. We then filter out all evidence nodes that do not influence the hypothesis in the direction of its posterior probability given all the evidence. For the remaining nodes, the posterior probabilities are then normalized so that

they fully span the range 0 to 1; call this the *importance* of each node. We define important nodes to be ones with an importance value greater than some threshold. The threshold is selectable by the user and is currently set to 0.7. We normalize the posterior probabilities since we are interested in identifying pieces of evidence with relatively strong influence on the probability of the hypothesis. This is not determined by the absolute value of the posterior probability but rather by the value relative to the prior probability of the hypothesis and the posteriors for the other pieces of evidence.

This algorithm is similar to that of Suermondt [9, Ch4]. Rather than *instantiating* each piece of evidence individually, Suermondt *removes* each piece of evidence individually and computes the posterior probability of the hypothesis without that piece of evidence. An influential piece of evidence is one for which the posterior probability without the evidence is significantly lower than with the evidence. The two approaches identify different pieces of evidence as influential in different circumstances. Suppose we have two pieces of evidence E_1 and E_2 and that $P(H) = 0.1$, $P(H|E_1) = 0.8$, $P(H|E_2) = 0.6$, and $P(H|E_1, E_2) = 0.8$. Then Suermondt's method will flag only E_1 as significant, while our method will flag both pieces of evidence as significant. In such a circumstance a combination of the two methods is probably best. Now suppose that $P(H) = 0.1$, $P(H|E_1) = 0.8$, $P(H|E_2) = 0.8$, and $P(H|E_1, E_2) = 0.85$. Suermondt's method will flag neither piece of evidence as significant, while our method will flag both as significant. In this case it is clear that our method produces the better explanation. Suermondt further discusses using his technique on all possible subsets of the set of evidence in order to identify sets of evidence that may be collectively significant but for which no single element is individually relevant. The combinatorics can make this computationally expensive.

3.2.2 Identifying Paths of Influence

Determining the paths along which an evidence node influences a hypothesis node is a multi-step process. We first identify all paths along which evidence can flow based on d-separation [7]. This set will often be too large to serve as a meaningful explanation, so we limit the explanation to five paths, first selected according to the strength of the path and then according to the length of the path. The rationale behind this scheme is that our foremost objective is to tell the user how the evidence influences the hypothesis. For the explanation to be accurate, it needs to provide the strongest paths. But we may have many paths that provide an equally strong link between evidence and hypothesis. Since a good explanation should be concise and understandable, we choose the shortest paths among those that are equally strong.

$+(E)+$ $-(\neg E)--+$	$-(E)$ $+(\epsilon \wedge \neg E)+-$	$+(\neg\epsilon \wedge \neg E)-$

Figure 4: Traversal chart for identifying active paths.

The generation of paths of influence starts with a backward search through the network to mark nodes as being predecessors of the hypothesis node or an evidence node, or both.

MarkNodes(*Network*)
Mark the hypothesis node and all evidence nodes, as well as all direct and indirect predecessors of the hypothesis node and the evidence nodes as *possibly related.* Mark all direct and indirect predecessors of evidence nodes as being *epsilon nodes.* These two markings can be performed in one pass through the network.

Next we use these markings to identify all paths of influence between the evidence nodes identified as important and the hypothesis node. To do this we use the chart shown in figure 4. The chart uses the definition of d-separation to indicate the paths along which evidence can flow through a given node based on whether the node is an evidence node (E), whether it has an evidence node below it (ϵ), and whether the edges are incoming from a parent (+) or outgoing to a child (−). For example, the first entry says that if a node is an evidence node and a path enters the node along an incoming edge then it can exit the node only along another incoming edge.

We find the paths by doing a depth-first search from each important evidence node. This is initiated by calling FindInfluentialPaths with the network, the hypothesis node, and the set of important evidence nodes.

FindInfluentialPaths(*Network, D, E*)

For each evidence node $e \in E$, let N be the set of possibly related direct parents and children of e
For each node $n \in N$
If n is a parent node of e, FindPaths(*Network, n, D, −, (e,n)*)
Else FindPaths(*Network, n, D, +, (e,n)*)

End

FindPaths(*Network, CurrentNode, DestinationNode, Direction, Path*)

If *CurrentNode* = *DestinationNode*
  Add *Path* to the list of paths.
Else
  Given *Direction*, the epsilon value of *CurrentNode*, and whether *CurrentNode* is an evidence node, determine the set N of direct parents and children of *CurrentNode* that are linked by an allowed edge by inspecting the traversal chart.
  For each possibly related node $n \in N$
    If n is not in *Path*
      Add n to the end of *Path*.
      If n is a parent of *CurrentNode*,
        FindPaths(*Network, n, DestinationNode, −, Path*)
      Else FindPaths(*Network, n, DestinationNode, +, Path*)
End

Suermondt's [9, Appendix A] algorithm of identifying direct chains from evidence to a node of interest performs the same function as our algorithm above but is somewhat more complex. His algorithm works in two stages: he first removes all barren nodes and nodes d-separated from the node of interest and then among the remaining nodes finds all chains from evidence nodes to the node of interest. By using the traversal chart in figure 4 we have combined the step for generating paths with the step for identifying d-separated nodes.

We now have a list of all active paths from the important evidence nodes to the hypothesis node. But in any sizable network, the number of active paths can be enormous. Displaying all this information would overwhelm the user. To solve this problem, we use a two-phased scheme. First, we generate only those paths of length less than an prespecified value. This value is selectable by the user and defaults to seven. Limiting the maximum path length speeds up the chain calculation processes considerably. If we still have more than five paths linking an evidence node to the hypothesis node then we sort the paths according to strength. We define the strength of a path using the notion that a chain is only as strong as its weakest link. Specifically, for each node in the path we compute absolute value of the difference between its unconditional probability and its probability conditioned on the evidence node at the end of the chain. The strength of the chain is the minimum of these differences. We then select the paths to display as follows.

1. Take the four strongest paths.
2. Starting from the fifth, take all paths with strength equal to that of the fifth.
3. Of the four chains selected in step 1, keep aside those (N) with strength different than that of the fifth.
4. For all chains that have strength equal to that of the fifth (this may include some of the 4 grabbed in step 1), sort them by length.
5. Keep shortest 5-N.

4 Implementation

BANTER is designed to be both portable and easily configured for new networks. Setting up the system for a new network requires providing only the network definition file, a BANTER defnition file, and a story template file, all described below.

BANTER is implemented in C[1] and runs on top of the HUGIN Bayesian network inference system [2]. HUGIN is used to perform all probability computations using a belief network specified in HUGIN's network definition format. The HUGIN interface consists of a set of functions from the HUGIN libraries which are used to load and compile a belief network, instantiate and uninstantiate nodes, propagate changes in individual nodes throughout the network, and obtain probability values for nodes.

BANTER's graphical interface is written using the Xaw graphics toolkit that comes with the X11 public-domain windowing package. Configuring the interface for a new network requires providing only a BANTER definition file, which in the case of a medical domain model contains a list of nodes grouped by history, physical findings, diseases, and diagnostic procedures. Each node must be followed by a value type which can be one of "FLOAT", "INTEGER", "STRING", or "BOOLEAN". These nodes and their corresponding classes will determine which menus each node will appear in and what type of method will be required to set the state of a given node.

The story template file is used to create the text for randomly generated story problems. The system generates a story problem by randomly choosing a set of values for the patient history and physical findings, randomly choosing a disease of interest, and expressing these choices by instantiating the story template.

[1]The software is available via anonymous ftp to ftp.cs.uwm.edu in pub/tech_reports/ai/banter.tar.Z.

Below is the story template used in conjunction with the gallbladder network to generate the example story problem discussed in section 2.3.

{SEX:Mr. Jones:Mrs. Jones:The patient} {AGE:is % years old, and} presents with [HISTORY], and denies (HISTORY). {SEX:His:Her:The patient's} {TEMPERATURE:temperature is %.%%.} {SEX:His:Her:The patient's} {WBC-COUNT:white-blood count is %.%%%.} Physical examination reveals [PHYSICAL-FINDINGS], and no evidence of (PHYSICAL-FINDINGS).
What is the best test to perform to <BOOLEAN:rule in:rule out> [DISEASE]?

5 Future Research

We are interested in two primary areas of future research: working with extremely large networks and generating more informative explanations. BANTER currently works well with relatively small networks. But in the newly emerging network models that contain thousands of nodes, inferences will become too slow to provide acceptable interaction and the explanations produced by the current algorithm will become too lengthy. For a given inference problem, only a portion of a given network model will typically be relevant. We have developed a technique for specifying a Bayesian network as a collection of rules in probability logic and generating that portion of the network relevant to a given computation [4]. Integrating this technique into BANTER will significantly reduce the complexity of inferences in very large networks.

We can provide more informative explanations by associating more semantic information with Bayesian networks. Rather than displaying paths of influence by simply showing nodes with arrows between them, we could indicate how each node influences its successor with terms like "causes", "elevates", and "has manifestation". We can further make explanations more informative, as well as more concise in the case of very large networks, by including abstraction information in the network model. Rather than providing only an explanation of the current scenario, one can move up the abstraction hierarchy to provide an explanation of more general scenarios, of which the current one is an instance.

Acknowledgements

We thank Dr. Finn V. Jensen and the HUGIN group at Aalborg University of generously providing us the use of the HUGIN system for this work. Haddawy was partially supported by NSF grant IRI-9207262. Kahn was supported in part by the 1993 American Roentgen Ray Society Scholarship.

References

[1] B. Abramson. ARCO1: An application of belief networks to the oil market. In *Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence*, pages 1–8, July 1991.

[2] S.K. Andersen, K.G. Olesen, F.V. Jensen, and F. Jensen. HUGIN – a shell for building Bayesian belief universes for expert sytems. In *Proceedings IJCAI-89*, pages 1080–1085, Detroit, Michigan, 1989.

[3] S. Andreassen, M. Woldbye, B. Falck, and S.K. Andersen. MUNIN – A causal probabilistic network for interpretation of electromyographic findings. In *Proceedings IJCAI-87*, pages 366–372, Milan, Italy, August 1987.

[4] P. Haddawy. Generating Bayesian networks from probability logic knowledge bases. In *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, pages 262–269, Seattle, July 1994.

[5] P. Haddawy, C.E. Kahn, and M. Butarbutar. A Bayesian network model for radiological diagnosis and procedure selection: Work-up of suspected gallbladder disease. *Medical Physics*, 21(7):1185–1192, July 1994.

[6] D.E. Heckerman, E.J. Horvitz, and B.N. Nathwani. Update on the Pathfinder project. In L.C. Kingsland, editor, *Proceedings of the Thirteenth Annual Symposium on Computer Applications in Medical Care*, pages 203–207, Los Almitos, CA, 1989.

[7] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo, CA, 1988.

[8] S. Srinivas and J. Breese. IDEAL: A software package for analysis of influence diagrams. In *Proceedings of the Sixth Conference on Uncertainty in Artificial Intelligence*, pages 212–219, July 1990.

[9] H.J. Suermondt. *Explanation in Bayesian Belief Networks*. PhD thesis, Medical Information Sciences, Standford University, March 1992.