Acquisition and Transformation of Likelihoods to Conditional Probabilities for Bayesian Networks

Claus Skaanning[1], Finn V. Jensen[2], Uffe Kjærulff[2], Anders L. Madsen[2]

[1]Hewlett-Packard Company,
Fredrik Bajers Vej 7C, 9220 Aalborg Ø, Denmark
[2]Department of Computer Science, Aalborg University,
Fredrik Bajers Vej 7C, 9220 Aalborg Ø, Denmark

claus_skaanning@hp.com, {fvj,uk,anders}@cs.auc.dk

25 September 1998

Abstract

When developing real-world applications of Bayesian networks one of the largest obstacles is the highly time consuming process of gathering probabilistic information. This paper presents an efficient technique applied for gathering probabilistic information for the large SACSO system for printing-system diagnosis. The technique allows the domain experts to provide their knowledge in an intuitive and efficient manner. The knowledge is formulated in terms of likelihoods, calling for methods to transform it into conditional probabilities suitable for the Bayesian network. The paper outlines a general transformation method based on symbolic propagation in a junction tree.

Key words: Bayesian network, knowledge acquisition, troubleshooting, single-fault assumption, symbolic inference, junction tree

1 Introduction

In the construction of Bayesian-network models, acquiring the probabilistic information from domain experts is often the most difficult part of the construction process. The traditional, straightforward approach to this knowledge acquisition task is difficult both in terms of the amount of time required to complete the task and in terms of the level of professional skill required.

In this paper we will describe a knowledge acquisition approach applied in the construction of a very large Bayesian-network model for printing system diagnosis. For decades, diagnosis has been a challenging application area for AI methodologies due to the complexity and data requirements involved. Much active research has been carried out in this area (de Kleer & Williams 1987, Genesereth 1984, Heckerman et al. 1995, Breese & Heckerman 1996). The purpose of diagnostic systems is to ultimately determine the set of faults that best explains a set of symptoms. A diagnostic system can request information from the world, and each time new information is obtained, it will update its current view of the world.

The diagnostic engine in our work is based on the troubleshooting method suggested by Heckerman et al. (1995). This method provides suggestions for optimal observations, repairs, and configuration changes to obtain further information. The troubleshooter is myopic, i.e., it only has one-step lookahead, and it is based on the single-fault assumption.

The diagnostic model for the printing system is being developed in a joint effort (SACSO — Systems for Automatic Customer Support Operations) between Aalborg University and the Hewlett-Packard Company. The model consists of several components: the application the user is printing from, the printer driver, the network connection, the server controlling the printer, the printer itself, etc. It is a complex task to troubleshoot such a system, and the printer industry spends enormous resources on customer support. The majority of these resources is devoted to labour, much of which can be avoided; for example, when support agents are sent out to repair printers that are not broken, or to solve problems that could have been handled off-location. Therefore, automating the troubleshooting process as much as possible has the potential of cutting down customer-service expenses substantially.

Contrary to the traditional, straightforward approach to knowledge acquisition in Bayesian networks, the knowledge acquisition approach presented in this paper allows for easy and intuitive elicitation of probabilities under the single-fault assumption. The approach provides the domain experts with very focused questions, speeding up the elicitation process and making the elicited probabilities much more accurate.

After brief reviews of Bayesian networks and junction trees, and our printing-system model, we motivate and present the method applied for eliciting the probabilistic knowledge for the model. The knowledge is formulated mostly in terms of likelihoods, calling for methods to transform it into conditional probabilities suitable for the Bayesian network. The paper outlines a general transformation method based on symbolic propagation in a junction tree.

2 Bayesian Networks and Junction Trees

A Bayesian network is defined as a pair (G, p), where $G = (V, E)$ is a graph and p is a probability function defined on the set of (most often) discrete-valued variables corresponding to the set of nodes (or vertices) V of G. The set of links (or edges) E are ordered pairs of nodes. The directed graph G is a graphical representation of the independence properties of p, where a missing link denotes the existence of one or more (conditional) independence statements between the variables associated with the link.

The set of edges, E, may have the properties that (i) whenever $(u, v) \in E$ it holds that $(v, u) \notin E$, and (ii) there is no directed path $u \to v \to \cdots \to w \to u$. In that case G is called a *directed, acyclic graph*. If condition (i) is relaxed, allowing undirected links, and condition (ii) is generalized appropriately, G is called a *chain graph*. For details on directed, acyclic graphs and chain graphs, consult, for example, the works of Lauritzen (1996) and Frydenberg (1989).

The probability function, p, for a Bayesian network, (G, p), factorizes according to G as

$$p(V) = \prod_{v \in V} p(v \,|\, \mathrm{pa}(v)),$$

where $\mathrm{pa}(v)$ denotes the set of 'parents' of v (that is, for each $u \in \mathrm{pa}(v)$, $(u, v) \in E$). Thus, the specification of p is done through the specification of a low-dimensional, conditional probability distribution for each variable given its parents.

The currently most efficient, exact method for probability updating in Bayesian networks is the junction-tree method (Jensen et al. 1990). A junction tree defines a clustering of the variables of the Bayesian network such that (i) each node of the tree consists of a cluster of variables, and (ii) the intersection between any two clusters is contained in each cluster on the path connecting the two clusters. Property (ii) makes it possible to perform inference through message propagation in the junction tree (i.e., an inward pass from the leaves of the tree towards a root cluster, which can be chosen arbitrarily, followed by an outward pass from the root towards the leaves). That is, propagation offers a way of computing probability conditional distributions of all unobserved variables given the observed variables.

The computational complexity of propagation of messages in a junction tree may be very big, sometimes unmanageably big. In fact exact updating of probabilities in Bayesian networks is NP-hard (Cooper 1990), although it is very efficient for some classes of Bayesian networks. The selection of clusters determines the computational complexity; however, computing an optimal clustering is also NP-hard (Arnborg et al. 1987). The current network for the printing-system diagnosis contains more than 2500 variables and many loops (creating bigger clusters), but still it can be transformed to a junction tree with reasonably efficient belief updating.

For more details on Bayesian networks and junction trees, see e.g. the book of Jensen (1996).

3 Modeling the Printing System

A conceptual Bayesian-network model of the printing system is shown in Figure 1. This illustrates the, basically, very simple structure of the model, with the state of each non-leaf variable given by a logical-OR combination of the states of its parents. Thus, for example, if $Subcause_1$ is present (i.e., the component modeled by $Subcause_1$ is malfunctioning), *Problem* will be present, as a result of the causal chain $Subcause_1 \longrightarrow Cause_n \longrightarrow Problem_1 \longrightarrow$ *Dataflow problems* $\longrightarrow$ *Problem*.

Notice that a logical-OR table for a problem-defining variable, P, with immediate causes $C_1, \ldots, C_n$ is insufficient for satisfying the conditions

$$p(P = yes \,|\, C_1, \ldots, C_n) = \begin{cases} 1 & \text{if } C_i = yes \text{ and } C_j = no \ \forall i, j : i = 1, \ldots, n,\ j \neq i \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

$$p(P = no \,|\, C_1, \ldots, C_n) = \begin{cases} 1 & \text{if } C_i = no \ \forall i : i = 1, \ldots, n \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Eq. 1 enforces the single-fault assumption among $C_1, \ldots, C_n$, and Eq. 2 makes sure that $P = no$ if and only if $C_1 = no, \ldots, C_n = no$. Both of these requirements cannot be satisfied simultaneously, since each vector $p(P \,|\, C_1 = c_1, \ldots, C_n = c_n)$ must be the zero vector whenever two or more of the c_i's equal *yes*, violating the requirement $\sum p(C \,|\, C_1 = c_1, \ldots, C_n = c_n) = 1$. In Section 4 we will discuss how to deal with this problem.

The single-fault assumption is an instance of a more general modelling problem in Bayesian networks which could be coined *the branching problem*, namely the problem of modelling the fact that a process may follow two distinct and mutually exclusive routes. The problem is very pronounced in decision networks (influence diagrams) where a decision may give rise to different sets of possible future decisions, cf. asymmetric decision trees.

The *Problem* variable indicates whether there is a problem in the printing system (i.e., a system fault). The general printing-system problems are split into several categories, of which the three most important are shown in Figure 1: *Error codes*, *Unexpected output*, and *Dataflow problems*. *Error codes* are problems characterized by an error code being present on the control panel of the printer. *Unexpected output* are problems characterized by output from the printer looking different from what the user expected it to. *Dataflow problems* are problems where the data associated with the print job either gets corrupted or lost somewhere in the flow from application to printer. Each of the three general categories of printing system errors are then further subdivided into specific problems that fall under each category, such as specific control panel error codes under the *Error codes* section, specific categories of unexpected output (not duplexed, spots on the paper, etc.) under *Unexpected output*, etc. Each of these problems are then further subdivided into causes which are sometimes divided into subcauses, sub-subcauses, etc.

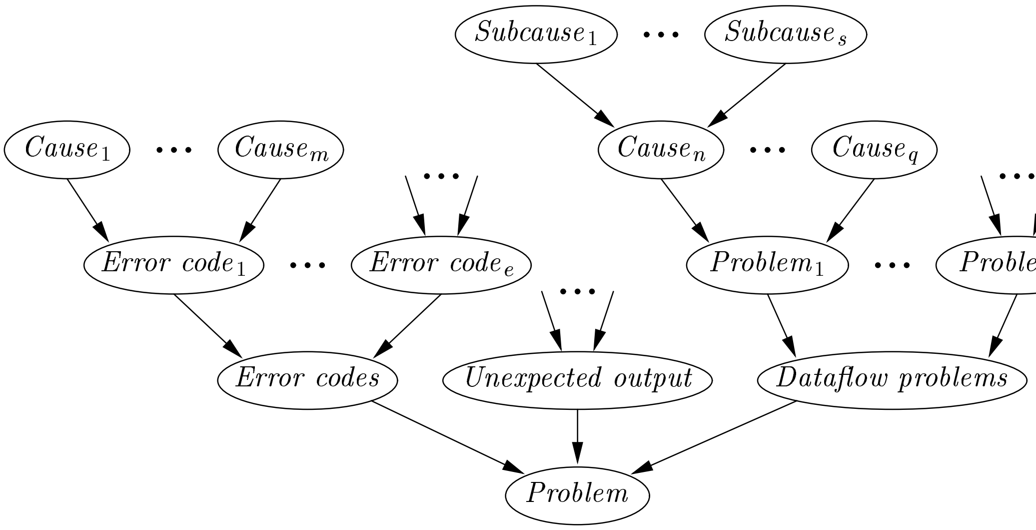

Figure 1: An overview of the Bayesian-network model for printing-system diagnosis. Most of the conditional probability tables are logical ORs, and the leaf variables without parents have prior probabilities.

The probability elicitation for such a structure at first seems simple, as the conditional probability distributions for all variables with parents are simple logical ORs. However, there still is a very large amount of leaf variables that need accurate prior probabilities. Since these variables represent rare events, the probabilities must be very small, say 0.001 or 0.0001. However, it is very difficult for the domain experts to get more precise than this because they only rarely experience the events. In fact, when all other probability distributions are logical ORs, all the sensitivity and precision of the model will be based on these prior probabilities.

The structure indicated in Figure 1 may give the impression that the printing-system model is tree-structured. This is, unfortunately, not true, as some events cause, for example, both unexpected output and data flow problems (e.g., electrical noise on cables). Such common causes severely increase the computational complexity of inference. In general, the ramifications on the knowledge acquisition task is not quite clear. In Section 4 we consider a small example network containing an instance of common causes in which the knowledge acquisition task is only slightly affected.

4 Probability Assessment

We are now going to present methods which may significantly simplify the process of probability assessment for networks of the kind illustrated in Figure 1. We start out by motivating the methods, which are then presented in Sections 4.2 and 4.3 for, respectively, tree-structured networks and networks with a given common-cause structure.

4.1 Motivation

One of the biggest stumbling blocks of the Bayesian network methodology is the requirement for large amounts of probability assessments. As described in the previous section, the printing system model requires prior probabilities for many leaf variables. For any methodology of probability elicitation to be really useful, it must be fast and easy to assess these probabilities. Therefore, one of the main tasks of the SACSO project has been to develop methods that make it as easy as possible for the printer experts to assess the probabilities, without increasing the number of assessments required and without jeopardizing the accuracy and consistency of the assessments. The developed method actually provides an increase in the accuracy of the probabilities over standard methods, and a decrease in the time requirements of the elicitation process. For illustration of the problems associated with the straightforward approach to probability elicitation, we will use the example Bayesian network in Figure 2 that models a parallel cable and its potential influences on the data flow.

The prior probabilities for the leaf variables, *Cable out of spec*, *Defective cable*, *Electrical noise*, *Cable in wrong plug*, and *Cable disconnected* must be assessed. The standard method would be to ask domain experts the following question for each of the leaf variables, C, to uncover its prior probability of occurence:

- What is the probability that C occurs?

Notice that, as discussed in Section 3, the conditional probability tables for variables *Corrupt data* and *Stopped data* cannot just be logical-ORs of their sub-causes, due to the single-fault assumption. That is, for example, if *Cable out of spec* is the actual cause of the system fault, by some probability, p_1, the data will be corrupt, and by probability $1 - p_1$ it will be stopped. Similarly, for the variables *Defective cable* and *Electrical noise*, we need probabilities p_2 and p_3. Thus, in addition to the above questions, we need to get answers from the experts to questions

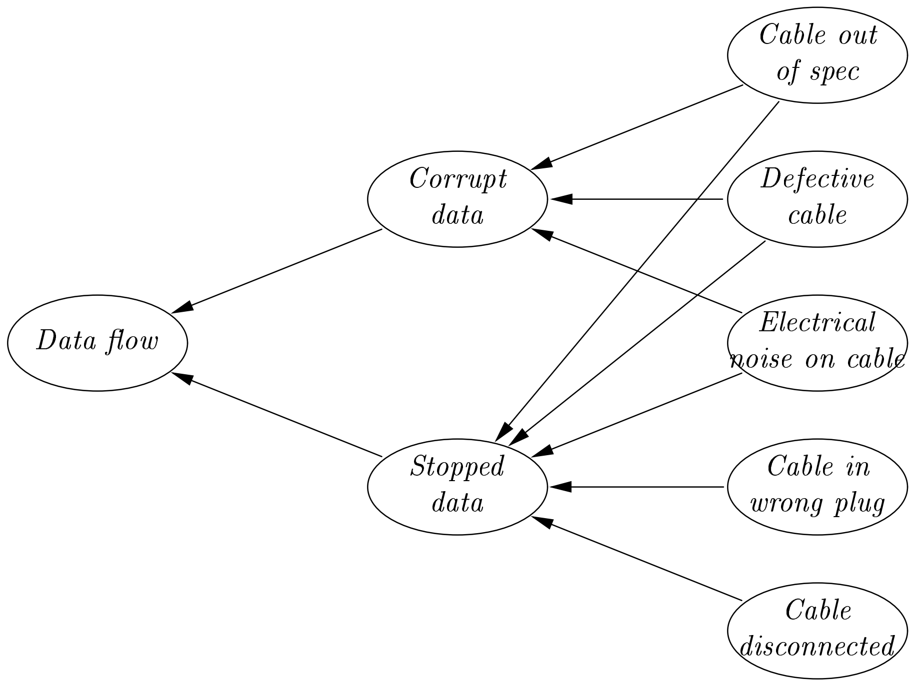

Figure 2: The Bayesian network model for the parallel cable and its potential influences on the data flow.

- What is the conditional probability, p_i, that *Corrupt data* $=$ *yes* given $C_i =$ *yes*?

for each variable, C_i, in the set $\{$*Cable out of spec*, *Defective cable*, *Electrical noise*$\}$, where $C_j =$ *no* for all other parents C_j.

These are not easy questions, as the experts must consider the general situation where no information is given (other than the fact that there is a system fault of some kind). As mentioned above, the probabilities must be very small. However, it may not be crucial whether a probability is estimated to 0.0001 or 0.0002 (Henrion et al. 1996), but it is very important that the ratios between the prior probabilities are fairly accurate. For example, if the cable being disconnected is twice as likely as it being in the wrong plug, then the ratio between the probabilities must reflect this. This suggests that a better way of estimating the probabilities could be to consider sets of them jointly. For example, the following questions seem easier to answer than the ones above:

- Which is more likely, the cable being in the wrong plug, or it being disconnected?

- If it is more likely that the cable is disconnected, how much more likely is it?

Still, however, the domain experts are given very little information about the state of the system, potentially making it very difficult to estimate the absolute sizes of the probabilities. The difficulty is further emphasized by the fact that most of the events occur very rarely. If, instead, the experts could work from the assumption that the output is stopped due to the parallel cable, then it would be much easier to answer the above questions. The scheme described in Sections 4.2 and 4.3 for finding prior probabilities takes this into consideration, and requires, basically, the experts to elicit a set of likelihoods at all levels of the Bayesian network, given the information at the lower level.

4.2 An Efficient Method for Tree-Structured Networks

Figure 3 shows an example diagnosis network (similar to, but much smaller than the one in Figure 1) modelling a problem being caused by one of two causes, where the problem is itself assumed to be the cause of some underlying, more abstract problem (system fault). One of these causes, $Cause_1$, has been further detailed, as it can be caused by one of two subcauses. As in the previous examples, the probability tables for *Problem* and $Cause_1$ are simple logical ORs. However, we need prior probabilities for $Cause_2$, $Subcause_1$, and $Subcause_2$. To find these probabilities, we ask the domain experts the following questions:

- Given that there is a system fault, what is the likelihood that *Problem* is the cause of it? Assume that the experts make the assessment that *Problem* is the cause of system faults in 10% of the cases.

- Given that there is a system fault, and *Problem* is the cause of it, what is the likelihood that $Cause_1$ is the underlying cause of *Problem* (and, implicitly, that $Cause_2$ is the underlying cause)? We follow the single-fault assumption, so the experts only have to consider the case where $Cause_1$ is present. Assume that the experts assess that in 40% of the cases $Cause_1$ is the cause, letting $Cause_2$ being the cause in 60% of the cases.

- Given that $Cause_1$ is the cause of the system fault, what is the likelihood that $Subcause_1$ is the underlying cause of $Cause_1$? The experts assess that in 20% of the cases, $Subcause_1$ is the underlying cause, leaving 80% for $Subcause_2$.

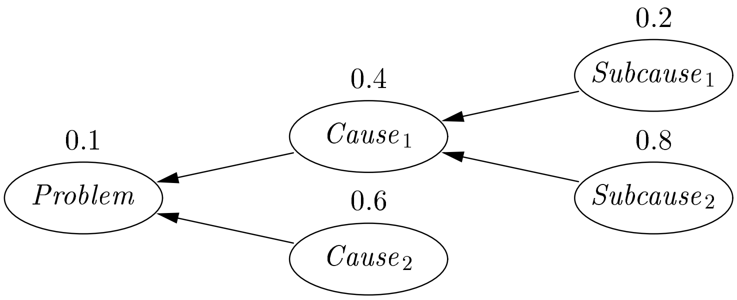

Figure 3: A simple Bayesian network with some likelihoods attached.

Thus, in Figure 3, we have estimates for the following likelihoods (all conditioned on the fact that a system fault is present):

$$p(Problem = yes) = 0.1,$$

$$p(Cause_1 = yes \,|\, Problem = yes) = 0.4,$$
$$p(Cause_2 = yes \,|\, Problem = yes) = 0.6,$$
$$p(Subcause_1 = yes \,|\, Cause_1 = yes) = 0.2,$$
$$p(Subcause_2 = yes \,|\, Cause_1 = yes) = 0.8.$$

It is now easy to find that:

$$\begin{aligned} p(Subcause_1 = yes \,|\, Problem = yes) &= p(Subcause_1 = yes \,|\, Cause_1 = yes) \\ &\quad\; p(Cause_1 = yes \,|\, Problem = yes) \\ &= 0.2 \times 0.4 = 0.08, \end{aligned} \tag{3}$$

and thus

$$\begin{aligned} p(\mathit{Subcause}_1 = \mathit{yes}) &= p(\mathit{Subcause}_1 = \mathit{yes} \mid \mathit{Problem} = \mathit{yes}) p(\mathit{Problem} = \mathit{yes}) \\ &= 0.08 \times 0.1 = 0.008. \end{aligned} \tag{4}$$

In Eq. 3 we really find the joint probability of $\mathit{Subcause}_1$ and Cause_1 given $\mathit{Problem} = \mathit{yes}$, however, it is not necessary to state $\mathit{Cause}_1 = \mathit{yes}$, as this is automatically given when $\mathit{Subcause}_1 = \mathit{yes}$, as Cause_1 is the logical OR of its subcauses. The same is the case in Eq. 4.

Thus for all the variables with no parents, we can find the prior probability by multiplying together the likelihoods of all causes on the path from the root to the variable in question.

When using the Bayesian network in Figure 3, we want the posterior probabilities of the variables to be equal to those elicited by the domain experts. However, this is not what we get when directly using the probabilities computed as in Eq. 4 as the prior probabilities in Figure 3. The reason for this is that the network in Figure 3 does not enforce the single-fault assumption, since the conditional probability tables for *Problem* and Cause_1 implement logical ORs of their parent variables. Ideally, these tables should satisfy the conditions expressed in Eqs. 1 and 2, but, as discussed in Section 3, this is impossible.

As an example of the violation of the single-fault assumption, notice that it is possible for both Cause_1 and Cause_2 to be in their *yes* state. Thus, after entering $\mathit{Problem} = \mathit{yes}$ as evidence and computing corresponding posterior beliefs, the probability that Cause_1 is present will be different from the 0.4 specified by the experts, and the ratio between the probabilities of Cause_1 and Cause_2 will be different from 0.4/0.6, due to the fact that Cause_1 has subcauses that can be present simultaneously while Cause_2 does not have subcauses.

The situation must be resolved via specification of joint distributions for the pairs $(\mathit{Cause}_1, \mathit{Cause}_2)$ and $(\mathit{Subcause}_1, \mathit{Subcause}_2)$. This can be accomplished either through introduction of 'constraint variables' or by letting the pairs be chain components of a chain graph with directed links identical to those for the acyclic, directed graph in Figure 3.

In the constraint-variables solution, two (binary) constraint variables, $\mathit{Constraint}_1$ and $\mathit{Constraint}_2$, are introduced, one for each of the pairs $(\mathit{Cause}_1, \mathit{Cause}_2)$ and $(\mathit{Subcause}_1, \mathit{Subcause}_2)$, respectively, where $\mathit{Constraint}_1$ is a child variable of Cause_1 and Cause_2, and $\mathit{Constraint}_2$ is a child variable of $\mathit{Subcause}_1$ and $\mathit{Subcause}_2$ (see Figure 4). Thus, when instantiating the constraint variables to their *on* states, their parent variables become dependent by imposing on them joint distributions given by $p(\mathit{Constraint}_1 = \mathit{on} \mid \mathit{Cause}_1, \mathit{Cause}_2)$ and $p(\mathit{Constraint}_2 = \mathit{on} \mid \mathit{Subcause}_1, \mathit{Subcause}_2)$, respectively. For pedagogical reasons we shall use the constraint-variables solution.

The probability distributions for the variables Cause_2, $\mathit{Subcause}_1$, and $\mathit{Subcause}_2$ are then specified to be uniform, as the constraint variables — when instantiated to their *on* states — enforce the desired prior distributions on the remaining variables.

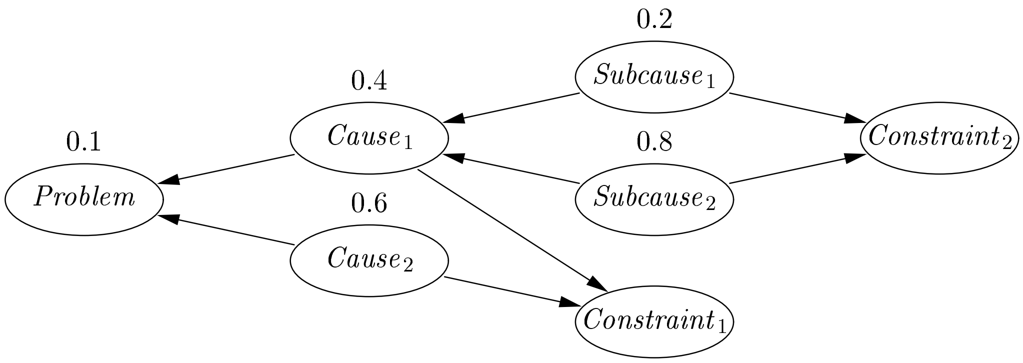

Figure 4: The added constraints enforce the single-fault assumption and correct likelihoods for all variables when *Problem* is given.

The desired joint probability distribution for *Problem*, $Cause_1$, and $Cause_2$ is indicated in Table 1. Specifically, notice that (i) the single-fault assumption is

$Cause_1$	*yes*	*yes*	*no*	*no*
$Cause_2$	*yes*	*no*	*yes*	*no*
$Problem = yes$	0	0.04	0.06	0
$Problem = no$	0	0	0	0.9

Table 1: The desired joint probability distribution for *Problem*, $Cause_1$, and $Cause_2$.

enforced by specifying zero probabilities in the 'yes-yes' column, (ii) the remaining three zero entries are given by the logical-OR relationship, (iii) the marginal distribution on $Problem = yes$ is 0.1, and (iv) the likelihoods for $Cause_1$ and $Cause_2$ given $Problem = yes$ are 0.4 and 0.6, respectively, as specified by the experts.

The distribution in Table 1 should result from the probability tables specified for the network. More specifically, it is obtained as the product of the logical-OR table, which is the conditional distribution for *Problem* given $Cause_1$ and $Cause_2$, and the table for the conditional distribution for $Constraint_1 = on$ given $Cause_1$ and $Cause_2$. Therefore, the conditional distribution for $Constraint_1$ given $Cause_1$ and $Cause_2$ must be given as indicated in Table 2. That is, when $Constraint_1$ is instantiated to its *on* state, it enforces the desired prior probabilities for the variables *Problem*, $Cause_1$, and $Cause_2$.

The instantiation of $Constraint_2$ to its *on* state shall then enforce the desired prior probabilities on the variables $Subcause_1$ and $Subcause_2$, without altering the

$Cause_1$	yes	yes	no	no
$Cause_2$	yes	no	yes	no
$Constraint_1 = on$	0	0.04	0.06	0.9
$Constraint_1 = off$	1	0.96	0.94	0.1

Table 2: The conditional probability distribution for $Constraint_1$ given $Cause_1$ and $Cause_2$.

probabilities enforced by $Constraint_1 = on$. Thus, $Constraint_2 = on$ should give rise to a joint probability distribution for $Cause_1$, $Subcause_1$, and $Subcause_2$ as displayed in Table 3.

$Subcause_1$	yes	yes	no	no
$Subcause_2$	yes	no	yes	no
$Cause_1 = yes$	0	0.1	0.4	0
$Cause_1 = no$	0	0	0	0.5

Table 3: The joint probability distribution for $Cause_1$, $Subcause_1$, and $Subcause_2$ given $Constraint_2 = on$.

In Table 3, notice that (i) the marginal distribution on $Cause_1$ is uniform (i.e., the distributions set by $Constraint_1$ are not altered), (ii) the single-fault assumption gets imposed on $Subcause_1$ and $Subcause_2$ through the zero probabilities in the 'yes-yes' column, and (iii) the likelihoods for $Subcause_1$ and $Subcause_2$ given $Cause_1 = yes$ are 0.2 and 0.8, respectively, as specified by the experts.

Again, as Table 3 is the product of the logical-OR table for $Cause_1$, and the table for the conditional distribution for $Constraint_2 = on$ given $Subcause_1$ and $Subcause_2$, the conditional distribution for $Constraint_2$ given $Subcause_1$ and $Subcause_2$ must be given as indicated in Table 4.

$Subcause_1$	yes	yes	no	no
$Subcause_2$	yes	no	yes	no
$Constraint_2 = on$	0	0.1	0.4	0.5
$Constraint_2 = off$	1	0.9	0.6	0.5

Table 4: The conditional probability distribution for $Constraint_2$ given $Subcause_1$ and $Subcause_2$.

For tree-structured networks, as exemplified above, computing the probability tables for the network is straightforward given the likelihoods specified by domain experts. In case of non-tree-structured networks, however, the computation gets much harder, as we shall see next.

4.3 Computing Probabilities for Causal Networks with Loops

For the fragment of the SACSO network displayed in Figure 2 we cannot use the simple method presented in Section 4.2.

Consider the network in Figure 5, which has structure similar to the fragment in Figure 2. Here, instantiation of the variables $Constraint_1$ and $Constraint_2$ to their *on* state enforces the single-fault assumption. Following the discussion in

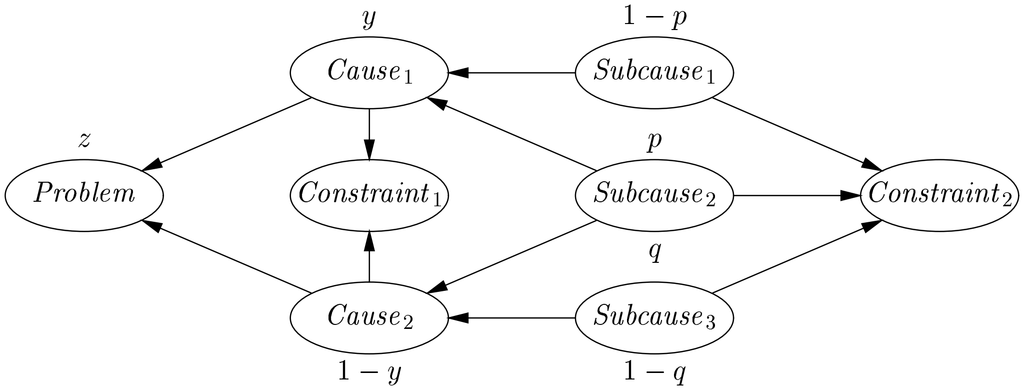

Figure 5: A Bayesian network with likelihood z that *Problem* is the cause of the system fault, likelihood p that $Subcause_2$ is the cause given that $Cause_1$ is present, likelihood q that $Subcause_2$ is the cause given that $Cause_2$ is present, and conditional probability y that $Cause_1$ is present given that $Subcause_2$ is the cause.

Section 4.1 of the probability assessment for the network fragment modelling the parallel cable component, we see that a probability elicitation process based solely on likelihood estimations is insufficient. In particular, when $Subcause_2$ is known to be the cause of the system fault, we need to elicit the probability, y, that $Cause_1$ is present (and the probability $1-y$ that $Cause_2$ is present). (Note that under the single-fault assumption $Cause_1$ and $Cause_2$ are mutually exclusive.)

Apart from this minor complication, the knowledge acquisition process is similar to the one for tree-structured networks presented in Section 4.2. That is, we get the likelihood z that *Problem* is the cause of the system fault, the likelihood p that $Subcause_2$ is the cause given that $Cause_1$ is present, and the likelihood q that $Subcause_2$ is the cause given that $Cause_2$ is present.

However, the prior probabilities $z_i = p(Subcause_i = yes)$, $i = 1, 2, 3$, are not easily determined. The approach we are going to take is based on solving a set of linear equations for z_1, z_2, and z_3.

We have the following equations:

$$p(\mathit{Problem} = \mathit{yes}) = z \tag{5}$$

$$p(\mathit{Cause}_1 = \mathit{yes} \,|\, \mathit{Subcause}_1 = \mathit{no}, \mathit{Subcause}_2 = \mathit{yes}) = y \tag{6}$$

$$p(\mathit{Cause}_2 = \mathit{yes} \,|\, \mathit{Subcause}_2 = \mathit{yes}, \mathit{Subcause}_3 = \mathit{no}) = 1 - y \tag{7}$$

$$p(\mathit{Subcause}_2 = \mathit{yes} \,|\, \mathit{Cause}_1 = \mathit{yes}) = p \tag{8}$$

$$p(\mathit{Subcause}_1 = \mathit{yes} \,|\, \mathit{Cause}_1 = \mathit{yes}) = 1 - p \tag{9}$$

$$p(\mathit{Subcause}_2 = \mathit{yes} \,|\, \mathit{Cause}_2 = \mathit{yes}) = q \tag{10}$$

$$p(\mathit{Subcause}_3 = \mathit{yes} \,|\, \mathit{Cause}_2 = \mathit{yes}) = 1 - q \tag{11}$$

First, we need to establish expressions for the likelihoods z, p, and q in terms of y, z_1, z_2, and z_3. We do that by performing three propagations in the parameterized Bayesian network in Figure 5 with tables given as follows:

1. Logical-OR table for $p(\mathit{Problem} \,|\, \mathit{Cause}_1, \mathit{Cause}_2)$.

2. Logical-NAND table for $p(\mathit{Constraint}_1 \,|\, \mathit{Cause}_1, \mathit{Cause}_2)$.

3. Logical-NAND table for $p(\mathit{Constraint}_2 \,|\, \mathit{Subcause}_1, \mathit{Subcause}_2)$.

4. Tables for $\mathit{Subcause}_i$, $i = 1, 2, 3$, given by $p(\mathit{Subcause}_i) = (z_i, 1 - z_i)$.

5. The table for $p(\mathit{Cause}_1 \,|\, \mathit{Subcause}_1, \mathit{Subcause}_2)$ given by

$\mathit{Subcause}_1$ $\mathit{Subcause}_2$	yes yes	yes no	no yes	no no
$\mathit{Cause}_1 = \mathit{yes}$	1	1	y	0
$\mathit{Cause}_1 = \mathit{no}$	0	0	$1 - y$	1

6. The table for $p(\mathit{Cause}_2 \,|\, \mathit{Subcause}_2, \mathit{Subcause}_3)$ given by

$\mathit{Subcause}_2$ $\mathit{Subcause}_3$	yes yes	yes no	no yes	no no
$\mathit{Cause}_2 = \mathit{yes}$	1	$1 - y$	1	0
$\mathit{Cause}_2 = \mathit{no}$	0	y	0	1

To perform inference in such a parameterized Bayesian network, we have implemented the Hugin inference algorithm (Andersen et al. 1989) in Maple V®. Expressions for z, p, and q are now derived as follows:

1. Perform propagation in the parameterized Bayesian network, with evidence $Constraint_1 = yes$ and $Constraint_2 = yes$. This determines an expression for z.

2. Perform propagation with additional evidence $Problem = yes$ and $Cause_1 = yes$. This determines an expression for p.

3. Perform propagation with evidence $Constraint_1 = yes$, $Constraint_2 = yes$, $Problem = yes$ and $Cause_2 = yes$. This gives us an expression for q.

Finally, we let Maple V® solve the system of equations with respect to z_1, z_2, and z_3. This results in the expressions

$$\begin{aligned}
z_1 &= \frac{zy^2p(q-1)}{qyz(p+y-2) - q(y^2-2y-z+1) - py^2} \\
z_2 &= \frac{pqz}{q(y^2-2y+1-z(y^2-2y+1-p(y^2-y+1))) + py^2(1-z)} \\
z_3 &= \frac{qz(p-1+y(py-2p-y+2))}{q(2y-y^2-1+pz(1-y)) + py^2(z-1)}
\end{aligned}$$

Choosing values $p = 0.3$, $q = 0.8$, $y = 0.4$, $z = 0.1$, we find $z_1 = 0.041558$, $z_2 = 0.104058$, and $z_3 = 0.010345$. Entering these values into the above parameterized tables, and performing propagation with evidence $Constraint_1 = on$ and $Constraint_2 = on$, we get the desired likelihoods.

As the method for knowledge acquisition in tree-structured networks presented in Section 4.2 and the method for a particular common-cause structure presented above covered all situations in the SACSO network, we have not yet tried to develop methods for more complex commmon-cause structures.

5 Discussion and Conclusions

The method presented in the present paper allows for easy and intuitive elicitation of probabilities, and transformation of them into conditional probabilities suitable for the specification of a Bayesian network for diagnosis that follows the single-fault assumption.

The knowledge acquisition is eased in the way that the domain experts have much more information to base their estimates on, and as a result of this, the knowledge acquisition becomes both faster and more accurate.

As mentioned in Section 4, the experts elicit probabilities based on the single-fault assumption for the simple reason that it greatly reduces the time consumption of the process. If multiple simultaneous faults were not ruled out, the experts would have to elicit a probability for every possible combination of causes which would exponentially increase the necessary time to complete elicitation process.

The use of constraint variables described in Sections 4.2 and 4.3 enforce the single-fault assumption. This is, however, not strictly necessary. It is easy to modify the constraint probabilities such that configurations, where more than one cause is present, are also possible. Given no information about the likelihood of causes occurring simultaneously, the best assumption we can make is that they are independent, so that the probability of causes C_1 and C_2 co-occurring is $p(C_1)p(C_2)$. This information can be entered into the tables for the constraint variables for all combinations of causes, and the model will then allow multiple causes, assuming them independent.

Even though it is easy to implement such a scheme it may not be desirable, as the single-fault assumption empirically seems much better than the independence-of-causes assumption. A malfunctioning printing system is usually troubleshot shortly after the malfunction occurred, so it is unlikely that something else in the system has become malfunctioning in the meantime. It is also very unlikely that two (or more) components become malfunctioning at almost the same time and caused the problem. Also, if a multiple-fault situation has occured, it will most often be due to one of small number of special events like, for example, stroke of lightning, use of excessive physical force, coffee poured into the printer, etc., which can easily be detected through interviewing. Therefore it seems very reasonable to assume that only a single component is causing the problem at hand.

Efficiency issues have not been the focus of the SACSO project so far, as the memory requirements of the constructed model have not exceeded the available capacity. As it is likely that there will be connections between different parts of the model in future versions, the memory requirements are expected to grow, and we may be forced to consider these issues. With the current unfinished version it takes a few seconds to answer a query. If, in the finished model, memory requirements become too big this can always be fixed by disconnecting the various parts of the model and handling queries independently of each other, a solution which is not completely satisfactory, though.

References

Andersen, S. K., Olesen, K. G., Jensen, F. V. & Jensen, F. (1989). HUGIN — A shell for building Bayesian belief universes for expert systems, *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pp. 1080–1085.

Arnborg, S., Corneil, D. G. & Proskurowski, A. (1987). Complexity of finding embeddings in a k-tree, *SIAM Journal on Algebraic and Discrete Methods*

8: 277–284.

Breese, J. S. & Heckerman, D. (1996). Decision-theoretic troubleshooting: A framework for repair and experiment, *Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence*, Morgan Kaufmann Publishers, San Francisco, pp. 124–132.

Cooper, G. F. (1990). The computational complexity of probabilistic inference using Bayesian belief networks, *Artificial Intelligence* **42**: 393–405.

de Kleer, J. & Williams, B. (1987). Diagnosing multiple faults, *Artificial Intelligence* **32**: 97–130.

Frydenberg, M. (1989). The chain graph Markov property, *Scandinavian Journal of Statistics* **17**: 333–353.

Genesereth, M. (1984). The use of design descriptions in automated diagnosis, *Artificial Intelligence* **24**: 311–319.

Heckerman, D., Breese, J. S. & Rommelse, K. (1995). Decision-theoretic troubleshooting, *Communications of the ACM* **38**(3): 49–56. Special issue on real-world applications on Bayesian networks.

Henrion, M., Pradhan, M., del Favero, B., Huang, K., Provan, G. & O'Rorke, P. (1996). Why is diagnosis using belief networks insensitive to imprecision in probabilities?, *Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence*, Morgan Kaufmann Publishers, San Francisco, California, pp. 307–314.

Jensen, F. V. (1996). *An Introduction to Bayesian Networks*, UCL Press, London.

Jensen, F. V., Lauritzen, S. L. & Olesen, K. G. (1990). Bayesian updating in causal probabilistic networks by local computations, *Computational Statistics Quarterly* **4**: 269–282.

Lauritzen, S. L. (1996). *Graphical models*, Oxford Statistical Science Series, Clarendon Press, Oxford, England.