Examination of Bayesian Belief Network for Safety Assessment of Nuclear Computer-based Systems

ESPRIT DeVa Project 20072

Bev Littlewood, Lorenzo Strigini, David Wright
Centre for Software Reliability, City University

P.-J. Courtois
AV Nuclear

September 29, 1998

Abstract We report here on a continuation of work on the Bayesian Belief Network (BBN) model described in [Fenton, Littlewood et al. 1998]. As explained in the previous deliverable, our model concerns *one part* of the safety assessment task for computer and software based nuclear systems. We have produced a first complete, functioning version of our BBN model by eliciting a large numerical *node probability table* (NPT) required for our 'Design Process Performance' variable. The requirement for such large numerical NPTs poses some difficult questions about how, in general, large NPTs should be elicited from domain experts. We report about the methods we have devised to support the expert in building and validating a BBN. On the one hand, we have proceeded by eliciting approximate descriptions of the expert's probabilistic beliefs, in terms of properties like stochastic orderings among distributions; on the other hand, we have explored ways of presenting to the expert visual and algebraic descriptions of relations among variables in the BBN, to assist the expert in an ongoing assessment of the validity of the BBN.

1 Introduction

This is the second report about an ongoing activity exploring the use of Bayesian Belief Networks (BBNs) in the safety assessment of systems. This activity is part of Task 5.1, "Dependability Evaluation Using Disparate Sources of Evidence", of DeVa, and is related to Task 1.3, "Design Guidelines for Validation and Licensing", which analyses the sources of evidence that will be involved in the construction of a BBN for a safety case.

The benefit sought from the use of BBNs is better ability for assessors to handle the combination of diverse evidence that is the basis of safety assessment. This evidence includes disparate elements like known conformance to standards of design and methodology, demonstrated competence of the organisations involved in producing a system, results of verification and validation activities on various products of the design process. Deriving a single judgement of satisfactory

safety from all this evidence is usually an informal process of "expert judgement", which may be unreliable and is difficult to analyse and verify. The "safety argument" - the reasoning that links the evidence to the final judgement - is mostly in the assessor's mind, and its descriptions on paper are typically limited to enumerations of items of evidence, without a detailed explanation of how these are assumed to support or counter one another.

We hope that the use of BBNs may make this hidden safety argument visible, communicable and auditable. BBNs offer a formal mathematical language for describing reasoning in uncertain situations. The assessor can thus describe his "safety argument" in a form that he can re-examine and "debug". He can describe the causal models that he has assumed to apply to the situation considered. This description implicitly specifies the value of the evidence considered in predicting the safety of the product, and the inference process can be automatically performed by software tools. At the same time, the formal description allows the experts to analyse and discuss the constituent parts of the "safety argument", devise empirical tests of their validity, and so on.

The case study to which this report refers is described in detail in the previous report [Fenton, Littlewood et al. 1998]. We give here a brief description of its context and of the BBN produced. We assume the reader to be already familiar with the language of BBNs; some general references to BBNs are [Pearl 1988, Andersen, Olesen et al. 1989, Jensen 1996, Lauritzen, Spiegelhalter 1988], examples of use are in [CACM1995], and introductory discussions and examples of their possible use in dependability assessment are in [Delic, Mazzanti, Strigini 1995, Delic, Mazzanti, Strigini 1997, Neil, Littlewood, Fenton 1996].

This report concentrates on issues of elicitation and validation of a BBN, as evidenced in the case study.

At this stage, we have completed a first version of the BBN with completely filled "node probability tables" (NPTs)[1]. We thus have a first, unvalidated version of a machine-supported probabilistic model that can answer questions of the form "given this set of observed evidence, what is the probability distribution for the states of this variable (or, in BBN jargon, 'for the states of this node')". The fact of this being a "first, unvalidated version" requires some more discussion, introducing the main topics of this report, i.e., methods for elicitation and validation of BBNs.

The BBN which has been built is complex, and defining its nodes and their relationships required the expert to define his own beliefs to a finer level of detail than usual. It was clearly impossible to obtain complete definitions of each NPT in a single step. The expert would need to refine the description of his beliefs by degrees, starting with a broad brush approximation, then observing it and correcting it by degrees. Indeed, if the expert's way of reasoning about certain aspects of the safety argument has always been to depend on an intuitive judgement process, as honed by experience, the expert starts the exercise of building the BBN without an *explicit* knowledge of his own beliefs about the dependencies among specific variables in the BBN.

The method we used here involved asking questions about orderings: both orderings of perceived relative *importance, or significance* between model variables; and *stochastic* orderings of probability distributions. Having obtained the views of domain experts regarding these orderings, we represented the influence of those variables considered less significant as simple linear perturba-

[1] Also called "conditional probability tables", or CPTs.

tions of smaller NPTs that embody the conditional distributions produced by variation of *only* the *more* significant conditioning variables.

The expert can now study this first description of his belief. In the first instance, he can simply use the BBN thus produced to infer consequences from hypothetical evidence. The first-pass completion of all the numerical NPTs of our model enables precise consequences of our NPT inputs, and of our other model assumptions—when these are taken in conjunction with hypothetical observable variable values—to be produced by BBN propagation tools such as Hugin. These model outputs take the form of updated numerical probability distributions for the goal variables which a safety assessor wishes to predict. We provide some examples of numerical model outputs, and remark on trends and influences that these appear to indicate.

This kind of feed back may be insufficient for the expert to reach confidence that the BBN he has produced is a satisfactory description of his beliefs. The dependencies between variables are described by tables of real numbers, with an infinity of possible variations. Considering the implications of specific hypothetical observations is one way of challenging the accuracy of the BBN (as a representation of the expert's "true" beliefs, or, at a later stage, of some scientific consensus between experts or of the actual uncertainty in the real world), but not necessarily an efficient way.

We have considered some different forms that we believe feedback to the expert might usefully take. These include graphical representations, the calculation of derivatives, and algebraic-based representations of the consequences of model structure. We studied this latter form in some detail: taking advantage of the fact that the topological structure of this BBN is close to that of a *polytree*, we obtained an algorithm producing complete algebraic expressions for all the probabilities of interest in the BBN, as functions of the NPT values and the observations.

In section 2 we recall the context of our case study and the structure of the BBN we produced. In section, 3, we discuss the issues of validation that our methods for feed-back to the expert are meant to address. Section 4 shows some forms of direct feedback obtainable via the use of the Hugin model. Section 4.3 describes how the BBN can treated algebraically in the special case of a polytree topology, and shows an example of application of this method to our BBN. A discussion of our results and future developments follows in section 5. An appendix contains details of how we constructed the large 'Design Process Performance' NPT.

2 The case study and the BBN

2.1 Context of the case study.

This BBN addresses the early part of the lifecycle of a nuclear safety system, during which two documents are produced, the 'System Requirements Document' and the 'Computer System Specification Document', and subjected to various analyses. Its goal variable is 'Safety Adequacy of Computer System Specification', identifying the quality through which the results of this phase affect successive phases of development. These two main documents are produced and modified through the interaction of three 'personae' (each typically consisting of a team or

subset of an organisation): the *system manufacturer;* the *system licensee* (future user of the system) and the *independent assessor* (who works on behalf of a safety authority and is responsible for eventually recommending approval of the system from the safety viewpoint). In the scenario considered here, the independent assessor has [partial] visibility (through access to documents and personnel) of the development process through which these documents are produced and validated, rather than being required to evaluate the finished product and documentation only.

In more detail, the lifecycle of the [computer part of] a safety system typically includes the production and verification of:–

1. a System Requirements Document. This document describes the environment of operation as well as the functions of the safety system. It lists the system's foreseeable failure modes, with probabilities, criticality and intended lines of defence against each of these modes, and assesses the criticality of the system;

2. the 'Computer System Specification Document'. This specifies and justifies, among other things, the allocation of safety functions between hardware and software respectively, and must demonstrate that the computer system architecture satisfies the system and safety requirements, in particular concerning adequate levels of redundancy and diversity, and barriers between safety and non-safety functions. A failure modes and effects analysis should also be included in terms of the software and hardware structural decomposition into system components, and the methods and mechanisms of auto-detection by the system of its own failure should be specified.

2.2 Node definitions and structure of the network topology.

The network topology we elicited is shown in Figure 1. We see that its general structure strongly reflects the life-cycle model used. The graph can be roughly divided into three sub-graphs, along the boundaries indicated in the figure by the dotted lines. The subgraphs concern respectively the quality of the requirements document, the design process that leads from this to the computer system specification, and the quality of the specification document itself.

Here follows the list of the names of the nodes, with terse explanations where necessary. A few conventions must be kept in mind for comprehension of this list: i) the names of nodes are meant to be reasonably self-explanatory, *if read in the context* of the subgraph to which the node belongs: thus, for instance, the node named 'Completeness & Correctness' in the bottom part of the figure refers to the completeness and correctness of the requirement document; ii) we have underscored the names of those nodes that represent observable variables; iii) the possible states of each node are listed between brackets following its name; iv) when a variable is defined in terms of subjective judgement or observation, the observer or judge is the independent assessor, unless otherwise specified. The possible states of a node are usually ordered on a scale of increasing or decreasing 'quality', with one major exception in the 'Safety Adequacy of Computer System Specification' node.

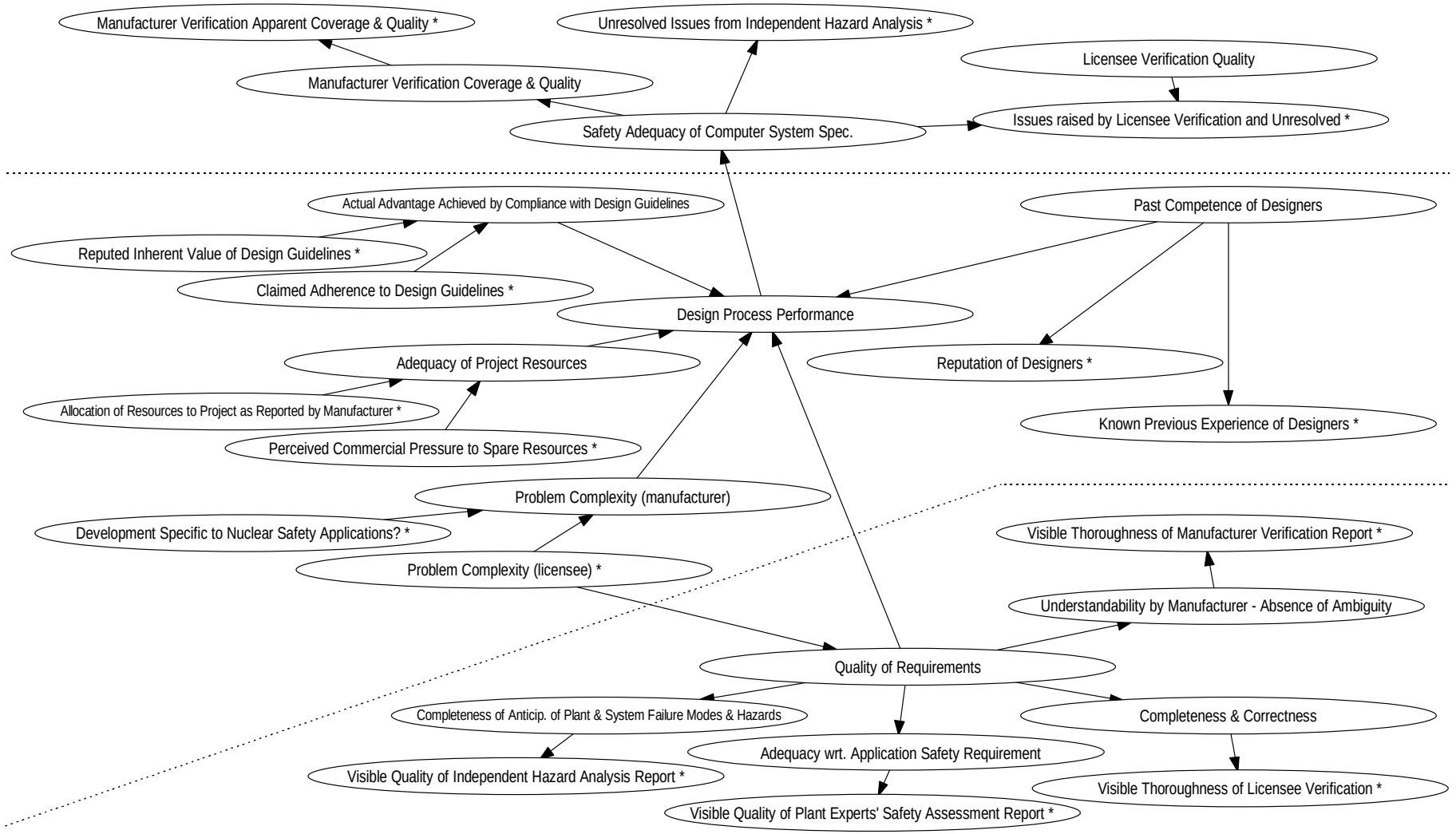

Figure 1: BBN topology. The dotted lines separate the three sub-graphs described in the text.

2.2.1 'Requirements Document' sub-graph

Quality of Requirements ***('Poor' 'OK' 'Good')*** This covers both qualities of contents, like 'consistency' and of presentation, like 'understandability' and 'verifiability'.

Completeness of Anticipation of Plant & System Failure Modes & Hazards ***('Sketchy', 'Satisfactory', 'Detailed')***

Visible Quality of Independent Hazard Analysis Report ***('Superficial' 'Average' 'Thorough')*** This report is produced by the licensee. By judging its visible quality the independent assessor gains evidence about the thoroughness with which hazards were analysed in the requirement document.

Adequacy with respect to Application Safety Requirements ***('Unsatisfactory' 'Satisfactory')***

Visible Quality of Plant Experts' Safety Assessment Report ***('Superficial' 'Average' 'Thorough')*** Observable consequences of its non-observable parent.

Completeness & Correctness ***('No' 'Yes')***

Visible Thoroughness of Licensee Verification ***('Superficial' 'Average' 'Thorough')*** Here 'licensee' means the licensee's system architect.

Understandability by Manufacturer – Absence of Ambiguity ***('Inadequate' 'Satisfactory' 'Good')***

Visible Thoroughness of Manufacturer Verification Report ***('Superficial' 'Average' 'Thorough')*** Could include the reports on results from the execution of prototypes. Quality of test plan, test coverage, & test reports.

2.2.2 'Design Process' sub-graph

Design Process Performance ***('Unsatisfactory' 'OK' 'Good')***

Actual Advantage Achieved by Compliance with Design Guidelines ***('No' 'Yes')*** Advantages resulting from complying with recommendations pertaining to specifications, design, verification, validation, such as those within specific standards such as the IEC 880.

Reputed Inherent Value of Design Guidelines ***('Low' 'Good')***

Claimed Adherence to Design Guidelines ***('No' 'Yes')***

Problem Complexity (manufacturer) ***('Complex/Difficult' 'Moderate' 'Simple/Easy')*** The manufacturer may replace the licensee's problem by a different problem, of a greater or lesser complexity, or may transform the original problem into a different problem - e.g. for reasons to do with their longer-term commercial intentions for marketing solutions to applications related to this one.

Development Specific to Nuclear Safety Application? ***('No' 'Yes')*** System modules used may, or may not, have originally been intended for use with safety-critical systems, or may not be intended to be used in future only in this application. This requirement for portability to applications of other kinds can create added system complexity. For example additional system configuration variables might have to be introduced into the design for this purpose.

Problem Complexity (licensee) ***('Complex/Difficult' 'Moderate' 'Simple/Easy')*** This node refers to the complexity or difficulty of the original problem owned by the system licensee which motivates the licensee to commission the system.

Past Competence of Designers ***('Low' 'Average' 'Good')***

Known Previous Experience of Designers ***('0 Similar Systems Licensed' '1 Similar System Licensed' '>1 Similar Systems Licensed')*** The state variable here is the number of success stories with 'similar systems'—success stories meaning that the system was licensed and has not since manifested safety defects in operation.

Reputation of Designers ***('Doubtful' 'Average' 'Good')***

Adequacy of Project Resources ***('Inadequate' 'Adequate')***

Perceived Commercial Pressure to Spare Resources ***('High' 'Low')***

Allocation of Resources to the Project as Reported by Manufacturer ***('Inadequate' 'Adequate' 'More than Adequate')***

2.2.3 'Computer System Specification' sub-graph

Safety Adequacy of Computer System Specification ***('Awful' 'Unsatisfactory' 'OK' 'Good' 'Wonderful')*** The node states have the following meanings:

- Awful = It is quite clear from the presentation that there are unresolved problems with the specification
- Unsatisfactory = It appears that there may be unresolved problems here but they are not easy to identify or diagnose with confidence
- OK = We are fairly sure there are no unresolved safety problems but this was not obvious, nor can we now state it with the very highest levels of certainty, because of inadequacies in the presentation
- Good = There are no unresolved safety problems; and the high quality presentation makes this apparent
- Wonderful = The presentation is very clear and it is quite obvious that there are no safety problems with this specification document

Note that it is not correct to think of the state space described here as a one-dimensional five-point ordinal scale. We see it rather as a 'collapsed' version of two distinct dimensions: the true presence or absence of safety-related problems with this specification document; and the clarity of the presentation of the specification – its traceability, the extent to which it exemplifies successful 'design for validation' rather than only design for correct (system safety-related) functionality. This node is not observable in our restricted sense[2]. One of its two dimensions - namely the genuine presence or absence of remaining safety issues - cannot be directly observed in this sense. Neither is it entirely possible for the independent assessor to observe the degree to which the specification is understandable by implementers and amenable to detailed analysis by independent verifiers. These latter qualities are important since the computer system specifications will be the basis of the documentation given to the programmers.

Manufacturer Verification Coverage & Quality ***('Unsatisfactory' 'OK' 'Good')***

Manufacturer Verification Apparent Coverage & Quality ***('Unsatisfactory' 'OK' 'Good')*** As a rule, the findings of the reports presented by the manufacturer to the independent assessor will look reasonably good; hence the diagnostic importance of judging the process that produced these findings.

Issues raised by Licensee Verification and Unresolved ***('0 issues' 'A few issues' 'Many issues')***

Licensee Verification Quality ***('Low' 'OK' 'High')***

Unresolved issues from independent Hazard Analysis ***('No unresolved issues' 'Minor unresolved issues' 'Serious unresolved issues')*** (Independent hazard analysis performed by the regulatory authority itself.)

2.2.4 Further comments on this topology

The current network (Figure 1), is topologically only slightly different from what Pearl calls a *'causal polytree'* structure [Pearl 1988, §4.3]. (It contains a single cycle which if broken by the removal of a single arrow would leave a causal polytree.) We will take advantage of this fact later in Section 4.3 in order to help understand how we can represent algebraically various of the combined effects of what the BBN topology is saying—i.e. taking an algebraic view of some of the consequences of the system of conditional independence assumptions that the topology is stating pictorially. Once a BBN model such as ours here has been constructed, it is possible to input any combination of observations at any subset of the model's nodes and then to output using a computational engine such as Hugin the Bayesian consequences of that combination of node observations for our updated beliefs in the distribution of the other model nodes. However,

[2] By 'observable' for a node in this model we mean merely that we expect to input, into the BBN model when we apply it to the assessment of some system, a precise value (or perhaps in some circumstances a 'likelihood observation', see explanation on p14) for that node. So for our purposes here, 'observation' includes subjective assessment of a node's value (or even in some cases subjective assessment as the definition of the node's meaning) using expert judgement. If it is intended that the expert should make such a judgement of the value of a particular node when assessing a particular system, then, for purposes of brevity in this paper and of consistency with terminology used in the formal theory of BBNs, we classify this act as an 'observation' and term the node 'observable'.

we would only expect in practice to observe the values of some of the nodes of a BBN. These are indicated in Figure 1 by appending an asterisk to the node name. We see that in the expected application of this network, the assumption regarding observability is *almost* simply that all the *leaf nodes* (the nodes having one arrow attached to them) will normally be observable to the independent assessor: and that the other nodes will normally be unobservable to the independent assessor. (The two exceptions are: (i) that 'Issues raised by Licensee Verification and Unresolved' is observable whilst 'Licensee Verification Quality' may well not be observed; and (ii) that 'Problem Complexity (licensee)' will normally be observable.)

We draw attention to these two features of the current topology not because we regard them as particularly desirable or undesirable, but merely as a way of assisting the reader to digest the model topology more easily. They are features peculiar to this BBN model. Each individual BBN may be examined for the accidental presence of specific features that facilitate understanding. It is also possible that future criticism of the structure of this net - whether in the light of output it produces, or simply by inspection - will result in changes to the topology which will change the character of the net.

2.2.5 Model construction

As reported in [Fenton, Littlewood et al. 1998], building the BBN model was an iterative process: although the construction broadly proceeded in 'top-down' fashion from the definition of the nodes to that of the BBN topology to that of the NPTs, elicitation at a later stage in this sequence often prompted reconsideration of previous stages and changes in the information that had been elicited at an earlier stage.

At the writing of the previous report [Fenton, Littlewood et al. 1998], the BBN was mostly completed, with the exception of the contents of the large NPT for the node 'Design Process Performance', which presented a serious problem in having the highest number of parent nodes (five) and thus from some viewpoint the most complex joint conditional probability distribution to describe.

Since then, we have completely specified this table and made subsequent minor changes to other NPTs.

In the appendix, we describe in detail the approach we took to the elicitation of a high dimensions NPT for the Design Process Performance node. This was based on an exhaustive numerical elicitation for the reduced size NPT obtained by fixing the states of a pair of parent nodes identified as least significant of the five parents and then varying the states of the most significant nodes until every combination had been covered. To represent the effects of variation in the states of the two parent nodes believed to be least significant, a simple parametric formula was designed to produce a linear displacement consistent with a short list of rules which the domain experts believed should govern the influence of these two less significant parents.

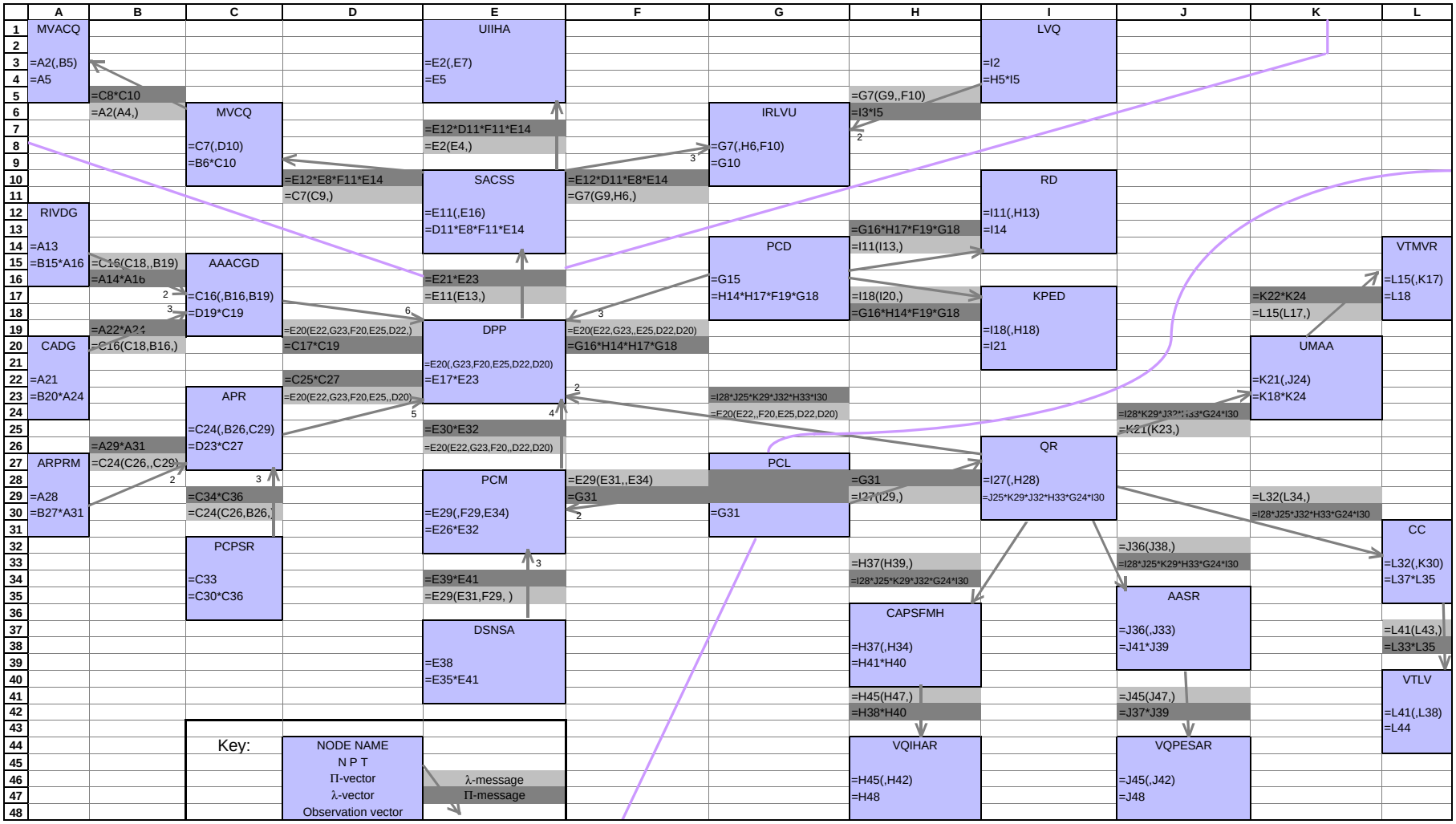

Figure 2: BBN Topology on Grid used to Assign Symbolic Names to Model Inputs, Outputs and Intermediate Quantities

3 Validation & Sensitivity Issues

Ideally, the BBN approach allows experts to express their beliefs about a complex problem within a formal probabilistic framework. In practice, of course, there will be a complex interaction between the expert and the process of elicitation, and this may result in the final BBN being inadequate in various possible ways. The process of elicitation itself may drive the experts to acquire beliefs that they did not previously have. This may be a positive effect - they may be asked questions that they had not previously thought of, and be thus led to deeper analyses, confutation of previous beliefs, etc. On the down side, the manner of the asking may result in experts giving answers that are not an accurate representation of their original, 'natural' beliefs, not through any conscious re-assessment but simply through the need to express themselves in an unfamiliar or inappropriate language.

We would expect the BBN formalism to allow experts to represent their belief in a way that is, on the one hand, detailed and rigorous (and thus amenable to analysis), but also, on the other hand, 'natural' for the experts themselves. For instance, we would expect an expert to choose the topology of a BBN so that its NPTs directly describe the expert's own understanding of the detailed constituent laws governing the dependence among variables[3]. The origin of these laws – detailed beliefs – may vary in nature, from accepted laws (physical or mathematical) to the expert's attempt to describe intuitive, experience-based laws that he *believes* he applies when producing safety judgements without the assistance of formal mathematics.

Multiple validation issues thus arise, which are particularly pertinent in the nuclear application we treated, because empirical data are relatively sparse. In other fields in which BBNs have been used successfully, such as in medicine, there are large empirical data bases and the dependence upon the unaided expert (for selecting topologies and especially for specifying NPTs) is less.

Validation issues may thus arise in at least three guises: Firstly, does the BBN represent the expert's initial understanding of the way he applies judgement to assessing the safety of a system? Issues at this stage may include:

- slips and other errors of execution in using the formalism. BBN support tools will flag those errors that produce an illegal BBN (violating either the axioms of probability or the requirement of an acyclic graph), but of course many other errors are possible;
- lack of self-consistency of the expert's intuition. The BBN was constructed by breaking a problem down into simpler problems, asking the expert to handle these, then using the formalism to construct the big picture, in the form of a multivariate joint probability distribution. The expert may discover that his tentative description of his detailed beliefs in terms of 'local' dependencies between variables contradicts some other aspect of his global beliefs, as evidenced in his judgements;

[3] A BBN is a way of specifying a joint probability distribution for all the variables corresponding to the BBN nodes, i.e., a complex set of probabilistic dependencies among all these variables. In general, a given joint distribution could be represented by anyone of a set of different BBN topologies (with the same nodes but different arcs). Choosing a BBN topology amounts to specifying which such 'local' dependencies shall be used to describe the 'global' set of dependencies (the joint probability distribution).

- the expert (and even the BBN specialist with whom the expert co-operates) may have not sufficiently internalised the subtleties of the BBN formalism, leading to errors in communicating/translating their beliefs correctly in terms of the formalism;

- the formalism may only match the expert's process of judgement if the latter is 'normatively correct', i.e., is a correct application of Bayesian reasoning for *some* prior joint probability distribution of the variables concerned. It is well known that expert judgement seldom approaches such formal perfection. A perfect match is thus unlikely, and mismatches will have to be resolved by the expert diagnosing errors in his previous intuitive judgement and/or in his specification of the BBN;

- a subtler issue involves the question whether the node definitions—particularly the abstract or subjective ones, involving for example (in our BBN) assessments of human and organisational characteristics such as competence and commercial pressure—are sufficiently well understood by everyone that it is possible to interpret confidently reasonably precise conditional probabilities concerning the variable's values, and assumptions about conditional independence relations between them. There are really two aspects to this issue about precision of agreed node definitions: one is that of the boundaries of the classification between discrete states; the more serious one is about what the definitions actually mean, e.g. what is being assumed about other variables, what is being corrected for, etc. For example, regarding 'Quality of Requirements' and 'Problem Complexity': are we sure we know what we would mean if we said that the requirements for the simpler problem A are of a higher quality than those for the more complex problem B? Clearly this problem may arise even for one expert trying to specify the definitions to himself alone, and even more when trying to communicate them to another expert.

- the experts were asked to produce very precise statements – numbers. Did they produce these simply because these were a requirement of the method, or do these numbers in fact represent similarly precise beliefs that the expert did actually hold before the elicitation session? It seems fair to assume that, except for NPTs which the expert produced by formal methods (mathematical or physical laws, inference from large amounts of data) the numerical distribution input to an NPT is just one from a range of similar distributions, among which the expert would be undecided. It is thus important to allow the expert to discriminate between conclusions that could be drawn from any distribution within this plausible range, and conclusions that are only implied by some distributions in the range. This is an application for sensitivity analysis.

The issues here essentially address concerns about whether the BBN has been elicited 'correctly' (i.e. it truly does capture the expert's beliefs). A second order of concerns is whether the beliefs, correctly represented in the BBN, will still represent the expert's (or experts') informed belief after analysis and discussion, in view of the detailed analysis of the previous belief structure, made possible by the BBN, and of additional knowledge (not represented in the BBN) about the problem? Last, another interesting question is whether the captured intuition of the expert really does accurately express the real-world uncertainty. In cases where there is a lot of data it may be possible to address this question - it has been examined in some detail, for example, in software reliability growth modeling using tools such as Prequential Likelihood

[Abdel-Ghaly, Chan, Littlewood 1986]. Such an investigation is, however, beyond the scope of our present work.

4 Results from this Model

To answer the questions raised in the previous section about the validity of the BBN that has been elicited, the expert needs to be able to see forms of feedback from the model. He needs to be able to see various (non-obvious) implications of the model to decide whether any of these are counter to his intuition. This is essentially the presentation of different viewpoints to the expert so that he can gain confidence that the model holds no unwelcome surprises, or, if there are such surprises, provide the opportunity to modify the model appropriately.

Mathematically, one view we can take of this model is that it is a function which maps 'observation states' onto the updated joint distributions, i.e. maps finite vectors of observations onto finite vectors of probabilities. This view assumes that the NPTs are all fixed and constitute part of the model structure. Given that we prefer not to view the elicited NPTs as fixed in stone at this stage, or perhaps not ever, we can enlarge the mathematical function domain to incorporate these NPT numbers. Then, we would have a function whose inputs were a collection of observation vectors *and* a collection of NPTs of the appropriate dimensions attached to each node. In either case, we have then constructed (by defining our model variables and their state spaces, making some conditional independence assumptions, and assuming a standard logic of probabilistic reasoning based on these) a vector-valued function of a vector argument. Our aim in carrying out the sensitivity analysis can be thought of mathematically as the task of gaining a better intuitive and analytical understanding of the properties of this function that we have constructed: In particular, are these properties in accordance with the domain expert's intuition about the practical problem under study, as we would ideally like to be able to believe that both the building blocks from which it was constructed and the formal rules of combination used in the construction should be?

There are many approaches which can be taken to such a task of feedback aimed at improving intuition about the properties of a complicated mathematical function between high-dimensioned spaces. Initially, we can use a numerical BBN computation engine such as Hugin to examine numerically what happens as we change certain argument values. We can plot the results as one or two dimensional projections of the behaviour or produce contour plots etc. Ultimately, a powerful, easy to use, and flexible user interface would greatly assist such a process. Secondly we can attempt to understand symbolically some aspects of the functional dependencies that result from our conditional independence assumptions. We provide some examples of sensitivity analysis output from each of these methods in subsections 4.2 and 4.3.

4.1 A Note on Vector Notation for 'Likelihood' Observations

We should mention here that the BBN propagation algorithm allows a rather general kind of observation, often called a *likelihood* observation, to be input to any node. What this means is

that observation of a node may be input in the form of a vector having one component for each possible state of that node. In the most general allowed interpretation of 'observation', each component of this observation vector represents a probability[4]. It is perhaps easiest to explain the semantics of four different basic forms of such a vector, in approximate order of decreasing frequency of use in a typical, real application of a BBN. But it should be born in mind that, mathematically, the last case is the most general allowed, and the preceding three cases are logically consistent with it, and are in fact special cases of it. Let v be the observation vector input at node X.

- If v is a 'flat' vector – i.e. all of its components are identical – then v represents *no observation at all* of variable X.
- If all the components of v but one are zero, then v represents a direct observation that X has the value corresponding to that component of v that is non-zero. (Sometimes below, we say that X is *fully* observed, to identify this case of direct observation of the exact value of X.)
- If more than one, but less than all, of the components of v are non-zero, and all equal, then v represents the observation that X takes some (unknown) one of the values corresponding to the positive components of v, i.e. what has been observed is merely the fact that X is confined to some proper subset of its state space;
- Lastly, if none of these cases apply, i.e. if there are positive-valued components of v, some of which differ from others in value, then we have the most general case of what is meant by a 'likelihood' observation. What this means is that, in statistical terminology, a 'likelihood function' for X has somehow been obtained, by a process of observation which is independent, given X, of the other nodes of the model. More precisely, we can think of this situation by supposing that some extra node N (perhaps having only two states) is linked into the existing BBN topology by a single arrow only, where this arrow links the new node N to X only. For this interpretation of *likelihood observation* to work properly in the general case, the direction of this arrow must be *from X to N*. Further, the value of N itself must now be assumed subsequently to have been fully observed. Let's say we observed that $N{=}\eta$. The likelihood observation, v at X, is now realised as (a scalar multiple of) the row[5] corresponding to the observed value $N{=}\eta$, of node N's NPT $p(N|X)$.

The reason for including this last, most general form of node observation is simply that it includes the previous three cases, which are obviously useful, and that the existing evidence propagation algorithm extends automatically to allow such a level of sophistication and generality of node 'observation'. To give a brief illustrative example, take the state-space *('Unsatisfactory' 'OK' 'Good')* of the node 'Manufacturer Verification Apparent Coverage & Quality'. Let v be the

[4]However, two vectors which are constant multiples of each other represent identical observations at a node. In other words, one can see the components of an observation vector as unnormalised, relative odds of the different possible states of the node. The only constraint on their values is that they must be non-negative, and at least one must be positive.

[5]We continue to use the convention used in [Fenton, Littlewood et al. 1998] and in the Hugin tool, that an NPT is printed as a 2-dimensional table, with parent node state combinations as column headings, and child node states as row headings

likelihood observation 3-vector input to this node (with its components ordered as for the three possible states above), then to take examples of each of the four cases above in turn: $v = \langle 1, 1, 1 \rangle$ means that this MVACQ model variable has not been observed at all; $v = \langle 0, 1, 0 \rangle$ means that we have observed, for the system under assessment, that the MVACQ is *'OK'*; $v = \langle 0, 1, 1 \rangle$ means we have excluded the possibility that this variable has the value *'Unsatisfactory'* (without, in the process, observing anything directly – i.e. independently of the values and NPTs of the other nodes in the network – that would cause us to believe *'OK'* to be any more, or any less, credible than *'Good'*); and lastly, $v = \langle 0, 1, 2 \rangle$ means we have observed 'something else' to have occurred, which is conditionally independent, given the value of MVACQ, of all other nodes in our model, but which is statistically associated with MVACQ itself, and, in fact, which we assess to be (i) impossible if MVACQ were *Unsatisfactory*, and (ii) precisely twice as probable if MVACQ were *'Good'* as it would be if MVACQ were only *'OK'*.

To keep things simple, we have used only the first two cases, of *no observation* and *full observation* in the examples used in this report. However, where, in §4.3, we have produced symbolic expressions for BBN output, we have always used a vector form of node observation, which is to be understood in the general sense explained here.

4.2 Use of Numerical Hugin Tool

Having input the model of Figure 1, along with finite state spaces and NPTs for each of the 28 model variables, a numerical computation engine such as Hugin, embodying a BBN 'propagation algorithm' consistent with Bayesian updating rules for conditional probability, is able to output updated distributions for any node, conditionally given any 'evidence', that is given any combination of observations for any subset of those variables[6]. As an illustration of the way evidence affects model conclusions, we traversed across a space of possible complete observations for our 15 variables which we intend typically to be observable, and plotted in Figure 3 the resulting distribution of the main goal node *Safety Adequacy of Computer System Specification*. The path through the combined 'observation space' which we have used here, was chosen with the loose idea in mind of progressing from 'least favourable' to 'most favourable' combinations of observations. Though, of course, because of the complexities of the variable interactions involved in this model, such an ordering of combinations according to favourability is not unambiguously defined between every pair of combinations.

This graphical format is clearly appropriate for the viewer to note general trends and exceptions to them. It is especially appropriate for nodes whose states can be considered as ordered on a scale (as is the case for the states of the node "Design Process Performance" though not for all the states of "Safety Adequacy of Computer System Specification"). Then, any line in the graph represents the probability of a node being in a 'higher' state (or 'better' or 'worse' as the case may be, depending on the meaning of the particular ordering) than a chosen threshold. For instance, the line under the region labelled "Good" in the upper half of the figure, if taken as function plot in isolation, represents the probability that the state of the node is worse than "Good", and the dependency of this probability on observed evidence (within the particular set

[6]Unless the observations are deemed inconsistent, according to the NPTs used. Thus, Hugin will refuse to propagate evidence which has zero probability according to the model

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	17	18	19	20	21	22	23	24
1																							
2																							
3	Values of All Observed Nodes															Safety Adequacy of Computer System Spec					Design Process Performance		
4	Manufacturer Verification Apparent Coverage & Quality	Unresolved Issues from Independent Hazard Analysis	Issues raised by Licensee Verification & Unresolved	Reputation of Designers	Known Previous Experience of Designers	Development Specific to Nuclear Safety Applications?	Problem Complexity (licensee)	Allocation of Resources to Project as Reported by Manufacturer	Perceived Commercial Pressure to Spare Resources	Reputed Inherent Value of Design Guidelines	Claimed Adherence to Design Guidelines	Visible Quality of Independent Hazard Analysis Report	Visible Quality of Plant Experts' Safety Assessment Report	Visible Thoroughness of Licensee Verification	Visible Thoroughness of Manufacturer Verification Report	Awful	Unsatisfactory	OK	Good	Wonderful	Unsatisfactory	OK	Good
5	Unsatisfactory	Serious unresolved issues	Many issues	Doubtful	0 Similar Systems Licensed	No	Moderate	Inadequate	High	Low	No	Superficial	Superficial	Average	Average	.1483293	.8501065	.0015508	.0000096	.0000039	.9844198	.0155601	.00002
6	Good	Serious unresolved issues	Many issues	Doubtful	1 Similar Systems Licensed	No	Moderate	Inadequate	High	Low	No	Superficial	Superficial	Average	Average	.4607278	.5338773	.0052906	.000074	.0000303	.9896737	.0102163	.0001099
7	Good	No unresolved issues	Many issues	Doubtful	2 Similar Systems Licensed	No	Moderate	Inadequate	High	Low	No	Superficial	Superficial	Average	Average	.2010787	.5325801	.2586101	.004132	.003599	.9559067	.0362966	.0077967
8	Good	No unresolved issues	0 issues	Doubtful	3 Similar Systems Licensed	No	Moderate	Inadequate	High	Low	No	Superficial	Superficial	Average	Average	.004728	.0265875	.7869248	.0762281	.1055317	.7364815	.1073815	.156137
9	Good	No unresolved issues	0 issues	Good	4 Similar Systems Licensed	No	Moderate	Inadequate	High	Low	No	Superficial	Superficial	Average	Average	.0028375	.0167173	.6000543	.1613983	.2189927	.4420004	.2339939	.3240058
10	Good	No unresolved issues	0 issues	Good	>1 Similar Systems Licensed	No	Moderate	Inadequate	High	Low	No	Superficial	Superficial	Average	Average	.0024263	.0146101	.5653549	.1799043	.2377044	.3779477	.2703621	.3516901
11	Good	No unresolved issues	0 issues	Good	>1 Similar Systems Licensed	Yes	Moderate	Inadequate	High	Low	No	Superficial	Superficial	Average	Average	.002091	.0130136	.5554022	.1949383	.2345548	.3257065	.327263	.3470304
12	Good	No unresolved issues	0 issues	Good	>1 Similar Systems Licensed	Yes	Moderate	Adequate	High	Low	No	Superficial	Superficial	Average	Average	.001782	.0114086	.5260306	.2088517	.2519271	.2775874	.3496794	.3727332
13	Good	No unresolved issues	0 issues	Good	>1 Similar Systems Licensed	Yes	Moderate	Adequate	Low	Low	No	Superficial	Superficial	Average	Average	.0017354	.0111663	.5215981	.2109513	.2545488	.2703258	.3530623	.376612
14	Good	No unresolved issues	0 issues	Good	>1 Similar Systems Licensed	Yes	Moderate	Adequate	Low	Good	No	Superficial	Superficial	Average	Average	.0017342	.0111599	.5214769	.2110074	.2546217	.2701319	.3531482	.3767199
15	Good	No unresolved issues	0 issues	Good	>1 Similar Systems Licensed	Yes	Moderate	Adequate	Low	Good	Yes	Superficial	Superficial	Average	Average	.0017255	.0111146	.5206305	.2113986	.2551308	.2687791	.3537478	.3774731
16	Good	No unresolved issues	0 issues	Good	>1 Similar Systems Licensed	Yes	Moderate	Adequate	Low	Good	Yes	Thorough	Thorough	Average	Average	.0002167	.0025367	.266324	.279714	.4512087	.033754	.2986703	.6675757
17	Good	No unresolved issues	0 issues	Good	>1 Similar Systems Licensed	Yes	Simple/Easy	Adequate	Low	Good	Yes	Thorough	Thorough	Average	Average	.000111	.0012407	.1441606	.2848369	.5696506	.017294	.1398922	.8428138
18	Good	No unresolved issues	0 issues	Good	>1 Similar Systems Licensed	Yes	Simple/Easy	Adequate	Low	Good	Yes	Thorough	Thorough	Thorough	Average	.000115	.001126	.124224	.284724	.589811	.0179126	.1094458	.8726416
19	Good	No unresolved issues	0 issues	Good	>1 Similar Systems Licensed	Yes	Simple/Easy	Adequate	Low	Good	Yes	Thorough	Thorough	Thorough	Thorough	.000115	.001126	.124224	.284724	.589811	.0179126	.1094458	.8726416
20																							
21																							
22																							
23																							
24																							
25																							
26																							
27																							
28																							
29																							
30																							
31																							
32																							
33																							
34																							
35																							
36																							
37																							
38																							
39																							
40																							
41																							
42																							
43																							
44																							
45																							
46																							
47																							
48																							
49																							
50																							
51																							

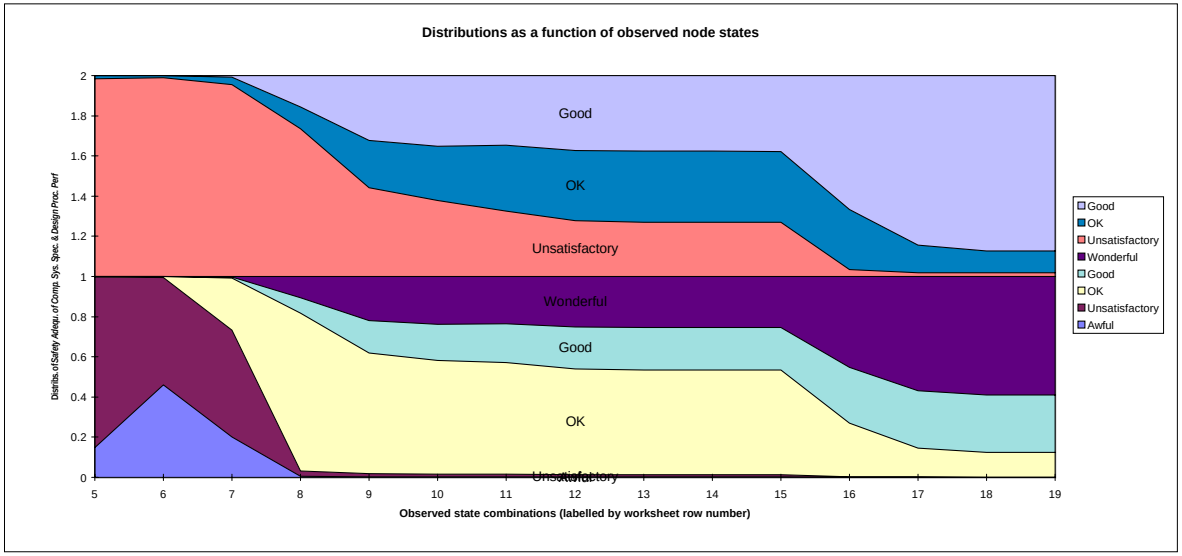

Figure 3: Updated Distributions Plotted as a Function of Some Observation Combinations

of observations represented on the x-axis).

Examining this particular figure, one would naturally expect every curve to be monotonically non-increasing, as the observations have been ordered along the x-axis to represent increasingly favourable evidence. The "spike" in the left-hand side of the bottom graph is an obvious "irregularity", which would be sure to attract the attention of the expert and prompt a re-analysis of the pertinent NPTs. The value of these cues may be of two kinds: on the one hand, the expert may conclude that the irregularity is due to an error in building the NPTs, and thus correct the error; on the other hand, the expert may conclude that the NPTs are a correct representation of his beliefs, and the perceived irregularity indicates a previously ignored, counterintuitive consequence of these beliefs. So, graphs like these can be a powerful visual aid whenever an expert believes a certain set of observations to be ordered from the viewpoint of its implications for a goal variable. Checks for simple properties, like monotonic trends, could also be programmed into an automatic tool which would repeatedly query the BBN engine and search for violations of the properties. Visual feedback, on the other hand, has the advantage that an expert may be surprised by features that do not violate any rule he may have specified beforehand, and yet require a re-analysis of the NPTs.

Plots such as that obtained in Figure 3 raise many questions about the many systematic relationships which doubtless must exist for the mathematical function which our model embodies. We succeeded in taking advantage of the fact that our model topology is close to a polytree in order to investigate several of these analytically.

4.3 Complementary Symbolic Analysis Using Polytree Propagation Algorithm

Our current topology can be treated as a *polytree* [Pearl 1988] provided we are prepared to block one of the nodes in our cycle by conditioning our argument on an assumed observation[7]. This method of conditioning on values of certain nodes in order to break loops in the BBN topology is discussed in [Pearl 1988, §4.4.2]. We use the method here to simplify a symbolic analysis of belief propagation within our network since, through this device, we avoid the need for a graph topological analysis of cliques and a junction tree [Pearl 1988, §4.4.1], [Lauritzen, Spiegelhalter 1988]. In our topology, the 'Problem Complexity (licensee)' node seems a good candidate for breaking our single loop since we would normally expect it to be fully observed by the assessor. If this is the case, then we can simply condition all our reasoning on its observed state. This creates a polytree topology, in which 'Quality of Requirements' becomes a root node, and 'Problem Complexity (Manufacturer)' then has only one parent. (Obviously the NPTs for these two nodes in the resulting simplified polytree topology will become functions of which state for 'Problem Complexity (licensee)' was observed.) Note that in fact this polytree-based analysis can be extended, to give results for inference for the other cases (the uncommon cases where 'Problem Complexity (licensee)' either is not observed at all, or is only observed in the more general 'likelihood observation' sense) by the 'unconditioning' method discussed near the beginning of §4.4.2 of [Pearl 1988].

[7] This must be a complete observation, i.e. the observation likelihood vector must be a vector of several zeros and a single one.

4.3.1 Some Useful Propeties of Polytree BBNs

The polytree evidence propagation algorithm [Pearl 1988, §4.3.1], [Kim, Pearl 1983] is simpler than the corresponding algorithm for general DAG topologies, and was actually solved in the literature some time before the more general DAG propagation problem [Pearl 1988, §4.4.1], [Lauritzen, Spiegelhalter 1988]. It has the advantages that with a polytree:–

- it becomes relatively easier to develop a good intuition about precisely how the belief updating is happening. Something of this intuition may then be conveyed to the domain expert, providing greater confidence in the computations carried out on the model they have helped to construct. This is much preferable to merely leaving the domain expert with the feeling that something nasty and complicated is happening within a black box— Or worse, allowing the risk that the expert may misunderstand the nature of the belief propagation algorithm, and consequently the semantics of the BBN model, as used to produce the model outputs. Hence the relative simplicity in understanding the polytree case may assist the process of continuing interaction and critique of the BBN model once an initial working version of it has been constructed.
- It becomes not too difficult a task to implement the updating algorithm with a symbolic mathematics tool such as Maple, which we have done. This again vastly eases the problem of making the functional relationships embodied by the model more apparent to anyone with some level of mathematical background.

We implemented the polytree propagation algorithm for this particular net symbolically in Maple in order to confirm, and to complement and expand on, the results available from the Hugin tool. This produced the following advanages:–

- Arbitrary (i.e. user specified) numerical precision available. This is effectively equivalent to unlimited precision, since the user can simply continue experimentally to increase the precision used until convergence is reached in the conclusions.
- The ability to substitute any particular observation, NPT, or part of an NPT, such as a single NPT column, as elicited from the domain expert, by an arbitrary parametric function, and to observe the resulting functional form of any selected model output.
- Greater ease and efficiency in obtaining plotted graphical output to illustrate functional relationships.
- The possibility of a gain in intuitive understanding which may result from access to algebraic, as well as visual topological, representations of model assumptions and their consequences. We found that these different forms of representation of model output complemented each other well.

In order to best interpret our algebraic results from this symbolic implementation of the polytree propagation algorithm, a basic understanding of the workings of the polytree proagation algorithm is required.

4.3.2 Brief Notes on Evidence Propagation in Polytrees

Once the topology and the node state spaces are specified, we can think of two kinds of user input to a BBN: NPTs, and *node observations*. There is exactly one of each of these attached to every node in the net, although inputting a flat observation vector to a node is equivalent to saying that no independent observation has been made of that node's value—so that our inference concerning its value must come entirely from reasoning with the combination of any observations that may have been made at other nodes of the net, together with the initially defined joint distribution for all of the nodes, as captured by the topology and the NPTs of all nodes. When any conjunction of node observations have been made, they will obviously each affect beliefs about the nodes observed. The exact change which the combination of node observations imposes on the beliefs about all nodes of the net can be calculated, in the polytree case, entirely by passing two "messages" along each arrow of the net: one, termed a π-message, which follows the direction of the arrow; and the other, called a λ-message which opposes the arrow direction. (π for *probabiity* and λ for *likelihood*). Each message is calculated by the sending node. The node is able to complete the calculation required for each message it must send as soon as it has received the incoming messages from every one of its arrows *except* the one along which it is required to send that message. (So nodes at the 'edge' of the network, in the sense of having only one arrow attached to them, can initiate the process by immediately each sending their single outgoing message.) For, example, in our polytree topology (with the 'Problem Complexity (licensee)' node removed by full observation, as explained above), the 'Safety Adequacy of Computer Systems Specification' node will receive four incoming messages (one of these being a π-message, and the other three being λ-messages. This node is also required to compute and send four outgoing messages. It is able to send its outgoing π-message to the 'Issues raised by Licensee Verification and Unresolved' node, as soon as it has itself received three messages: these being a π-message from the 'Design Process Performance' node, and one λ-message from each of the 'Unresolved Issues from Independent Hazard Analysis' node and the 'Manufacturer Verification Coverage and Quality' node. With a polytree topology, the single-connectedness, and the absence of a requirement to wait for a message from the arrow along which you wish to send, together ensure that the entire message exchanging process of evidence propagation terminates quickly after exactly twice as many messages have been generated as there are arrows in the network.

The π- and λ-messages associated with each arrow produce, and are produced by, two quantities associated with each node, likewise labeled π and λ. These latter are both vectors of length equal to the size of the state space of the node to which they are attached. The most obvious interest of these vectors is that multiplying the π and the λ component corresponding to any given value in that state-space of the node yields the updated probability of that node value, conditioned on all observations on the entire net[8]. This posterior distribution of a node given all observations at itself and at all other nodes, is the primary output of a BBN tool such as Hugin.

Returning to the π- and λ-*messages* associated with the arrows, we note that, regarded as functions, or algebraic expressions in the inputs, the messages that can be sent earlier will tend to be short algebraic expressions, and the last received messages (those arriving at the edge nodes, from where the first messages were sent) will tend to be much longer expressions involving many

[8] except that this product needs renormalizing by dividing by the sum of these products over the state space

model inputs. To be more precise, we can state the following properties concerning the functional dependencies of the quantities we have just mentioned here as follows:-

- Each message, associated with an arrow, is a function of all inputs in the part of the network 'behind' it (regarding the network as divisible into two parts by removal of the arrow along which the message passes) and is not affected at all by any inputs in the part of the network ahead of it. (These statements apply to NPT model inputs as well as to node observation inputs.)
- The π-value associated with a node can be written as a function involving only that node's NPT and all π-messages due to arrive at that node. It is unaffected by the node observation and unaffected by any of the λ-messages due to arrive at the node.
- The λ-value associated with a node can be written as a function involving only the node observation at that node itself and all λ-messages due to arrive at that node. It is unaffected by the node's NPT and unaffected by any of the π-messages due to arrive at the node.

Each of these four kinds of intermediate data item, involved in the evidence propagation process, has a straightforward interpretation in terms of a conditional probability, derived from the initial (prior to the input of node observations) joint probability distribution which the model's topology and all NPTs together embody. Using Pearl's [Pearl 1988] notation to refer to subsets of the node observations, let node U be a parent of node X and denote the conjunction of all observations at any nodes on the U-side of the arrow from U to X by e^+_{UX} and all observations at nodes on the X-side of this arrow by e^-_{UX}; so that, in an obvious notation, $e = \{e^+_{UX}, e^-_{UX}\}$ is the entire set of node observations on the network. (e stands for 'evidence', which is a term used synonymously with 'combination of node observations'.) This method of partitioning the total evidence into two parts is required for the conditional probability interpretation of the π- and λ-messages associated with the network arrows. For the π- and λ-vectors associated with individual nodes, a different partitioning of the evidence e is employed. For any node X with parents U_i, and children V_j, let e^+_X denote the conjunction over all U_i of the evidence $e^+_{U_iX}$, and let e^-_X denote the evidence at X itself in conjunction with all of the evidence $e^-_{XV_j}$ over all of X's children. Similarly, then we clearly have a partition $e = \{e^+_X, e^-_X\}$ and these two definitions provide two alternative ways of partitioning the evidence input to the model into two disjoint subsets—the first way based on selecting any arrow, and the second based on selecting any node. With these definitions, the four items of intermediate data associated with belief propagation are defined as the following conditional probabilities. For the messages associated with the *arrow* from U to X, we have π- and λ-messages defined as

$$\pi_X(U) = p(U|e^+_{UX}) \qquad \lambda_X(U) = p(e^-_{UX}|U) \tag{1}$$

both being vectors of length equal to the size of the state-space of the parent node U[9]. For the π- and λ-values associated with *node* U the definitions in terms of conditional probability are

$$\pi(U) = p(U|e^+_U) \qquad \lambda(U) = p(e^-_U|U) \tag{2}$$

[9]so, in using the notation $\pi_X(U)$ and $\lambda_X(U)$, the node subscript X indicates dependence on the *identity* of the node X only, whereas the bracketed argument (U) indicates dependence also on the value which could be taken by node U within its finite state-space. Hence we speak of these, for each pair of adjacent nodes (U, X) with U being the parent of X, as vector quantities, the vectors having one component for each possible value $U{=}u$ within U's state space.

both being vectors of length equal to the size of the state-space of U. See [Pearl 1988, §4.3.1] for the precise numeric calculations implied by these definitions (1), (2)[10].

4.3.3 Application of Polytree Algorithm in Symbolic Representations of Output

To some extent the value of different forms of symbolic model representations and model feedback will vary between individual experts. One problem with any of these forms is cognitive overload due to the large number of variables involved and the attempt to formulate non-trivial systems of probabilistic influence relationships between them. The topological network diagram is obviously one attempt to deal with this. It is successful in that it is easy to appreciate its structure, as compared to the structure of a long algebraic expression for a conditional probability involving many variables. But this simplification is achieved at the expense of introducing questions as to how well the ramifications of the topology's formal interpretation by the numerical BBN evidence propagation tools are appreciated by the domain expert involved in the construction of this topology. We suggest that, in the case of a large proportion of the experts likely to be involved in our kind of application, the generation (ideally automated) of selectable, more conventional algebraic forms of representation, for use in parallel with diagramatic forms of the BBN model topology, and numerical forms of BBN model output, may provide a valuable form of feedback to the expert for improved understanding, communication and validation purposes.

For the purposes of explaining the generation of symbolic, algebraic representations of the model, it is perhaps easiest to think 'in the reverse direction' to the direction indicated by the message passing algorithm described above and in [Pearl 1988] in connection with numerical belief updating from numerical values of all inputs. Selecting any goal node, X, its updated distribution given all the evidence e is the normalised component-wise product of its π- and λ-vectors, $\pi(X)$ and $\lambda(X)$ defined above.

$$p(x|e) = \text{(normalising constant)} \times \pi(x)\lambda(x) \tag{3}$$

for any value x of the node's state. Using a symbolic implementation of the algorithm, such as the one we produced using Maple, either of these vectors can be expanded into an expression involving incoming messages to node X together with the NPT and observation (if any) at node X itself. Each of these incoming messages can itself be expanded in terms of the NPT and observation at an adjacent node, together with messages arriving at nodes one or two steps remote from X in the polytree topology[11]. If this procedure is iterated to its conclusion, NPT and observation vector inputs at edge nodes of the net are finally reached, in all directions, fanning out from X along all branches of the tree structure. By this final stage of full expansion, the

[10] We should mention a minor point applying to all four of these definitions. The conditional probability distributions defined are *not* required to be normalised. All four items here are vectors. During all intermediate propagation calculations their components are not required to sum to 1, so that, strictly it is only proportionality, and not equality that is, required by these definitions.

[11] Note that in the case of the first iteration, looking at the messages arriving at X directly, the π-vector at X involves only π-messages, and the λ-vector at X involves only λ-messages. But in the subsequent iterations of expansion of the algebraic message-symbols involved, this division need no longer apply, and topologically more remote messages of *either* kind can become involved in the successively further expanded expressions for the π-vector at X, and similarly for the λ-vector at X.

expression has grown through successive expansions of symbols representing messages until it no longer contains anything but model inputs[12].

In §4.3.2, we described for each node of a polytree BBN topology, two model inputs (the node NPT and node observation vector) and two model outputs (the π- and λ-vectors of the node, though, as we explained, it is actually the normalised component-wise product $\lambda(x)\pi(x) \;/\; \sum_\xi \lambda(\xi)\pi(\xi)$ of these that is the single output vector of primary interest for that node). We described also for each arrow of the topology, two intermediate values required in the course of the calculation of these model outputs, called the π- and λ-messages associated with that arrow. In the examples of algebraic representations of the updating algorithm given below we have dealt with the problem of choosing algebraic symbols for the 160 ($=27\times4 + 26\times2$) quantities[13] implied by this as follows. We imagine the nodes and arrows to be each represented by columns of 5 and 2 cells, respectively, located on a spreadsheet-like rectangular grid, as indicated on Figure 2, and name each quantity according to the coordinates of the cell containing it[14]. Whilst not supplying names that are in any sense mnemonic of the node definitions, this approach does at least have the advantage of allowing the reader to quickly interpret any variable contained in the expressions that we produce with reference to the graphical topology diagrams of Figures 1 and 2.

This naming convention leads to evidence propagation expressions like

$$\mathrm{I29}_u = \mathrm{J25}_u\,\mathrm{K29}_u\,\mathrm{J32}_u\,\mathrm{H33}_u\,\mathrm{G24}_u\,\mathrm{I30}_u$$

which initially expands to

$$\mathrm{I29}_u = (\sum_\gamma \mathrm{K21}_{\gamma\,u}\,\mathrm{K23}_\gamma)\,(\sum_\beta \mathrm{L32}_{\beta\,u}\,\mathrm{L34}_\beta)\,(\sum_\alpha \mathrm{J36}_{\alpha\,u}\,\mathrm{J38}_\alpha)\,(\sum_\upsilon \mathrm{H37}_{\upsilon\,u}\,\mathrm{H39}_\upsilon)\,\mathrm{G24}_u\,\mathrm{I30}_u.$$

This latter expression still contains π- and λ-message symbols which may be expanded further, and this process repeated, with the resulting expressions for $\mathrm{I29}$ gradually getting larger by stages through successively expanding those symbols which denote messages. Note that in the second expression above, and in subsequent further-expanded replacements of it, we can assume—in accordance with our indications in §1 about which model variables we typically would expect to be observable in applications of this model—that the observation vectors $\mathrm{I30}$, $\mathrm{H40}$, $\mathrm{J39}$, $\mathrm{L35}$, and $\mathrm{K24}$ will be flat vectors used to indicate no observation at all. We will continue to make this assumption in the successive further expansions of the λ-vector $\mathrm{I29}$. If we did not make such an assumption but instead left all observation vectors arbitrary, we would simply get slightly longer, but otherwise similar expressions. Generally, expressions for expanded, or partially expanded model output, in the form of π- or λ-vectors of a node, take the form of multiple sums (over the

[12] In general it will by then contain an explicit dependence on every single NPT and observation in the entire network, which will be multiplied into a large product with corresponding index-variables identified and summed over *all* variables, *except* the node X itself and any nodes that have been fully observed. This product is simply the original joint distribution for all the model variables, multiplied by their observation (likelihood) vectors. Of course, for certain specific 'fortuitous' values of the model inputs, simplifications can occur—the most common one being that in which a flat λ-vector at a node destroys the functional dependence which the progression of this expansion process would otherwise have transmitted between any pair of its parents.

[13] After the removal of the 'Problem Complexity (Licensee)' node, by conditioning all probabilities on its fully observed value, we obtain a polytree topology having 27 nodes and 26 arrows.

[14] Figure 2 also shows the formula for each cell in terms of its immediately precedent cells, expressed in a straightforward shorthand. c.f. equations (1) and (2) and [Pearl 1988, pp182-3].

fixed, full range of each of a set of indices) of a product of indexed symbols. Provided we agree always to postpone the normalisation of probability distributions until immediately prior to their final output by the model, we can ignore constant scale factors in such sums at all intermediate stages of reasoning. So the effect of inputing a *'no-observation'* flat observation vector is simply to make the symbol for that observation vector independent of its solitary index in the product which is being summed. Hence its symbol becomes a constant, and effectively is removed from the expression. Thus, continuing to remove the symbols $\mathrm{I30}$, $\mathrm{H40}$, $\mathrm{J39}$, $\mathrm{L35}$, and $\mathrm{K24}$ whenever they are found within message-expansions, we obtain

$$\mathrm{I29}_u = \left\{\sum_\gamma \mathrm{K21}_{\gamma\,u}\,\mathrm{K23}_\gamma\right\}\left\{\sum_\beta \mathrm{L32}_{\beta\,u}\,\mathrm{L34}_\beta\right\}\left\{\sum_\alpha \mathrm{J36}_{\alpha\,u}\,\mathrm{J38}_\alpha\right\}\left\{\sum_v \mathrm{H37}_{v\,u}\,\mathrm{H39}_v\right\}\mathrm{G24}_u$$

and if we expand the four node λ-vectors $\mathrm{K23}$, $\mathrm{L34}$, $\mathrm{J38}$, and $\mathrm{H39}$, but choose *not* to expand the λ-message $\mathrm{G24}$ we obtain

$$\mathrm{I29}_u = \left\{\sum_\gamma \mathrm{K21}_{\gamma\,u}\,\mathrm{K18}_\gamma\right\}\left\{\sum_\beta \mathrm{L32}_{\beta\,u}\,\mathrm{L37}_\beta\right\}\left\{\sum_\alpha \mathrm{J36}_{\alpha\,u}\,\mathrm{J41}_\alpha\right\}\left\{\sum_v \mathrm{H37}_{v\,u}\,\mathrm{H41}_v\right\}\mathrm{G24}_u$$

and then, expanding the four resulting new λ-messages to

$$\begin{aligned}\mathrm{I29}_u = &\left\{\sum_\gamma \mathrm{K21}_{\gamma\,u}\left(\sum_z \mathrm{L15}_{z\,\gamma}\,\mathrm{L17}_z\right)\right\}\left\{\sum_\beta \mathrm{L32}_{\beta\,u}\left(\sum_y \mathrm{L41}_{y\,\beta}\,\mathrm{L43}_y\right)\right\}\\ &\left\{\sum_\alpha \mathrm{J36}_{\alpha\,u}\left(\sum_x \mathrm{J45}_{x\,\alpha}\,\mathrm{J47}_x\right)\right\}\left\{\sum_v \mathrm{H37}_{v\,u}\left(\sum_w \mathrm{H45}_{w\,v}\,\mathrm{H48}_w\right)\right\}\mathrm{G24}_u,\end{aligned}$$

and trivially, replacing these last four λ-vectors by the four observation vectors to which they are equal[15], we obtain

$$\begin{aligned}\mathrm{I29}_u = &\left\{\sum_\gamma \mathrm{K21}_{\gamma\,u}\left(\sum_z \mathrm{L15}_{z\,\gamma}\,\mathrm{L18}_z\right)\right\}\left\{\sum_\beta \mathrm{L32}_{\beta\,u}\left(\sum_y \mathrm{L41}_{y\,\beta}\,\mathrm{L44}_y\right)\right\}\\ &\left\{\sum_\alpha \mathrm{J36}_{\alpha\,u}\left(\sum_x \mathrm{J45}_{x\,\alpha}\,\mathrm{J48}_x\right)\right\}\left\{\sum_v \mathrm{H37}_{v\,u}\left(\sum_w \mathrm{H45}_{w\,v}\,\mathrm{H48}_w\right)\right\}\mathrm{G24}_u \qquad (4)\end{aligned}$$

This shows the fully expanded (except for our decision not to expand the symbol $\mathrm{G24}$) λ-vector of the 'Quality of Requirements' node under the assumption of general likelihood observation for the four observable nodes here. We should note that it is likely in practice that *full* observation will be made for many of the observable nodes. (See §4.1.) The effect of this on an expanded expression will be to remove the summation sign corresponding to the indexing variable for states of each fully observed node, and to interpret what was previously this dummy summation variable, as now a free variable which is assigned the value of the observed state of this node (because, within the sum, the observation vector symbol concerned will evaluate to zero for every other value of this index, by the definition of full observation). For example, if 'Visible Thoroughness of Licensee Verification' and 'Visible Thoroughness of Manufacturer Verification Report', having observation vectors $\mathrm{L44}$ and $\mathrm{L18}$, respectively, are fully observed,

[15] If a node has no children, then its λ-vector is identical with its observation vector.

then the summation signs with indices z and y in (4) are effectively removed, provided we interpret these two ex-dummy variables as now being assigned the observed states of these two nodes. It follows that $\mathrm{L44}_y$ and $\mathrm{L18}_z$ (being indexed *solely* by a variable whose summation sign has been removed) become constants and disappear. So the assumption of full observation for $\mathrm{L44}$ and $\mathrm{L18}$, together with our *non*observability assumptions explained above for $\mathrm{I30}$, $\mathrm{H40}$, $\mathrm{J39}$, $\mathrm{L35}$, and $\mathrm{K24}$ result in the full expansion

$$\mathrm{I29}_u = \left\{\sum_\gamma \mathrm{K21}_{\gamma\, u}\mathrm{L15}_{z\,\gamma}\right\}\left\{\sum_\beta \mathrm{L32}_{\beta\, u}\mathrm{L41}_{y\,\beta}\right\}$$
$$\left\{\sum_\alpha \mathrm{J36}_{\alpha\, u}\left(\sum_x \mathrm{J45}_{x\,\alpha}\,\mathrm{J48}_x\right)\right\}\left\{\sum_v \mathrm{H37}_{v\,u}\left(\sum_w \mathrm{H45}_{w\,v}\,\mathrm{H48}_w\right)\right\}\mathrm{G24}_u. \qquad (5)$$

A different, frequently occurring simplification (which we have not used in the particular derivation of (4) and (5)) arises from a default assumption that NPT *inputs* to the model *are* normalised – i.e., NPT 'column' totals, are equal for every column of the NPT (–thinking of an NPT, in its standard printed form, laid out with its first, 'conditioned' index forming the row headings, and the combinations of its other 'conditioning' indices forming the column headings). Thus, in the expressions for π- and λ-vectors, the *unweighted* sum over the *first* index of any NPT (which includes in particular evary multiply-indexed symbol in these expressions) can always be removed.

Notice that we could if we wished, expand also the λ-message $\mathrm{G24}$ in (4). This expansion would, if extended to its final conclusion, lead to a complicated expression substituted for $\mathrm{G24}$ in (4) which would (in general – but not in some particular input cases, depending on the values for some of the NPTs and observations) involve all the NPTs and observation vectors in the whole of the middle and top portions of our topology. Alternatively, we can leave the unexpanded term $\mathrm{G24}$ in place, as a concise representation for the combined effect of evidence and beliefs contained in the top and middle parts of our net on conclusions at nodes in the bottom part. In fact this option not to expand forms the basis of one of two obvious approaches to simplification of the algebraic expressions resulting from application of this polytree evidence propagation algorithm, which can be used to make them more comprehensible and useful to human beings:–

- Cut down the number of unassigned symbolic variables and use mostly, but not exclusively, numerical values, i.e. fill in *nearly* all the model inputs numerically, leaving just a few as unassigned symbols. We will show some examples of this later. The disadvantage is the difficulty of determining the extent to which any qualitative relationships which emerge (between the small number of variables which are left symbolic) depend on the precise values of the numbers assigned to the bulk of the other numerical model inputs. Of course this matters less if, in a particular model application, we can say that we have faith in the values used for all of the numerical model inputs we have assigned and regard these as 'validated'. Then we can say in consequence that algebraic relationships, sensitivities, and trends apparent in the algebra relating the remaining algebraic model inputs become likewise validated qualitative and quantitative relationships between unknowns, in the particular model application in question.

- Create limited expansions of model outputs, such that some variables in the polytree expansion process described above are expanded to their ultimate representation directly in

terms of model inputs, while other strategically selected chains of expansion are stopped when the message associated with a certain arrow is reached ($\mathrm{G24}$ in the example we just gave). At this point, the π- or λ-message not further expanded summarizes the impact of some subtree on the variable X for which the model output is being determined (and in fact on any other variable on the X-side of the arrow concerned). In view of the simple conditional-probability interpretation of this message given in equations (1), this provides an intuitively appealing way of summarizing relationships embodied in the model symbolically whilst controlling the complexity of the algebraic expressions presented to the expert, and at the same time referring the reasoning closely to the graphical polytree representation of the dependency model. Extending this idea into a proposed tool requirement, we might imagine that a graphical tool for making the selections involved might instructively facilitate this kind of activity, allowing a process of automatically assisted, logical model-structure feedback to the model builders. For example, one can envisage, a user or domain expert going through the following steps in using a tool: perusing the topological representation of the model; selecting a node (or nodes, in a slightly more sophisticated version) whose updated distributions they wish to view, for a while, in terms of providing a model output; requesting expansions of the algebra of that node's dependence on model inputs at other nodes, and some intermediate π- and λ-messages selected in order to control the growth of the resulting expressions and gain greater insight. The limits of such partially constrained chains of successive expansion might perhaps be selected graphically by the user by means of incorporating user-control over expanding graphical indicators of the dependencies involved – in a similar sort of way to the choices available from the graphical 'trace precedents' commands incorporated in modern spreadsheet 'auditing tools'. Having been shown the locations of the dependencies on the topology graph, the user would then be enabled to specify which chains of dependence are to be expanded algebraically, and how far along paths through the tree structure each expansion is to be continued, before its termination in terms of an unexpanded message. Then the tool might display automatically for the user the resulting algebraic expressions indicating precisely the form of the dependencies they had requested in this way, which could be printed or exported to other tools for plotting, symbolic differentiation, sensitivity analysis, etc.

It is useful to peruse these expressions in conjunction with having an eye on the topology—using the topological structure as a guide to the decomposition into meaningful sub-expressions which is necessary in order to mentally appreciate a more complex expression. There are surely some psychological considerations involved here. These may vary from one expert to another. In fact, with different forms of representation of the model, there are different views one can take of which is the 'primary' one. An expert who is very familiar with probabilistic reasoning and with factors such as the analytical meaning of conditional independence relations, may from the outset view the topology they helped to construct as a visual representation of underlying algebraic relations. Indeed this kind of expert might arguably view the topology as ultimately being a mere visual aid to the parsing of complicated algebraic relationships which actually are what really characterize the model. A different expert may view the arrows of the topology in terms of their being direct representations of causes and influences he perceives to be present in the world, with less awareness of the way that the BBN tool will interpret these in terms of its underlying formal logic of relations between, and factorizations of, specifically *joint and conditional probability distributions*.

At any intermediate stage, a remaining unexpanded π- or λ-message symbol is identical to a summarized form of the influence of all observations (and all elicited NPTs) more remote along that branch of the tree, that have not yet been reached in the expansion process. And in this way one can summarize the influence of observations in any subnet or subnets on the remainder of the network (or on a part of the remainder). There are some obvious examples: The 'Quality of Requirements' subnet evidence and assumptions, at the bottom, affects the rest of the net via only the single π-message

$$\mathrm{G23}_u = \mathrm{I27}_{u\,\mu} \left\{ \sum_{\gamma} \mathrm{K21}_{\gamma\,u} \left\{ \sum_{z} \mathrm{L15}_{z\,\gamma}\, \mathrm{L18}_z \right\} \right\} \left\{ \sum_{\beta} \mathrm{L32}_{\beta\,u} \left\{ \sum_{y} \mathrm{L41}_{y\,\beta}\, \mathrm{L44}_y \right\} \right\} \left\{ \sum_{\alpha} \mathrm{J36}_{\alpha\,u} \left\{ \sum_{x} \mathrm{J45}_{x\,\alpha}\, \mathrm{J48}_x \right\} \right\} \left\{ \sum_{v} \mathrm{H37}_{v\,u} \left\{ \sum_{w} \mathrm{H45}_{w\,v}\, \mathrm{H48}_w \right\} \right\}.$$

μ is our symbol for the known state of the fully observed 'Problem Complexity (licensee)' node, as required for our polytree assumption. This π-message in turn influences the 'Design Process Performance' node through its π-vector to give, in turn, a 'transmitted' effect on the top part of the net captured algebraically within the π-message

$$\mathrm{E16}_q = \sum_{a,d,g,r,u} \left[\mathrm{E20}_{q\,u\,r\,g\,d\,a}\, \mathrm{G23}_u \left\{ \mathrm{G15}_r \left(\sum_{s} \mathrm{I11}_{s\,r}\, \mathrm{I14}_s \right) \left(\sum_{t} \mathrm{I18}_{t\,r}\, \mathrm{I21}_t \right) \right\} \left\{ \sum_{h} \mathrm{E29}_{g\,\mu\,h}\, \mathrm{E38}_h\, \mathrm{E41}_h \right\} \left\{ \sum_{f,e} \mathrm{C24}_{d\,e\,f}\, \mathrm{A28}_e\, \mathrm{A31}_e\, \mathrm{C33}_f\, \mathrm{C36}_f \right\} \left\{ \sum_{c,b} \mathrm{C16}_{a\,b\,c}\, \mathrm{A13}_b\, \mathrm{A16}_b\, \mathrm{A21}_c\, \mathrm{A24}_c \right\} \right] \quad (6)$$

passed to the 'Safety Adequacy of Computer System Specification' node. Here, in our expansions for the π-messages from QR to DPP, and from DPP to SACSS, we have continued to substitute flat observation vector where indicated by our default assumptions of non-observability of some nodes, as indicated in §2.2.

Reviewing this last expression we can illustrate the flexibility of the choice whether or not to expand successive messages, by simplifying (6) so that the influence on $\mathrm{E16}$ of the π-message $\mathrm{G23}$ may be perceived more plainly

$$\mathrm{E16}_q = \sum_{a,d,g,r,u} \mathrm{E20}_{q\,u\,r\,g\,d\,a}\, \mathrm{G23}_u \mathrm{F20}_r \mathrm{E25}_g \mathrm{D22}_d \mathrm{D20}_a$$

as a function of the large NPT $\mathrm{E20}$ at the DPP node, together with the other π-messages arriving at DPP. These four messages each summarise the influence of a different branch of the middle part of our topology on the π-vector at the 'Design Process Performance' node, as comprehensively expanded in (6).

One way of understanding this expression is in terms of the way it can be said to 'transform' the π-message $\mathrm{G23}$ (which summarises the effect of all evidence from the bottom 'Quality of Requirements' part of the net) by its passage through the central 'Design Process' evidence part of the net, to its ultimate effect captured in the π-message $\mathrm{E16}$, which summarises the

influence, regarding inference on the top part of the net, of all evidence (strictly) below the 'Safety Adequacy of Computer System Specification' node. This transformation is actually just a linear one, equivalent to multiplication of the message $\mathrm{G23}$ by the 3×3 matrix

$$\mathrm{M}_{qu} = \sum_{a,d,g,r} \mathrm{E20}_{q\,u\,r\,g\,d\,a}\,\mathrm{F20}_r\mathrm{E25}_g\mathrm{D22}_d\mathrm{D20}_a$$

where, to expand this fully in terms of model inputs, we would expand along the 4 other branches feeding into the DPP node as in (6). Thus, M is a function of the four π-messages, $\mathrm{F20}$, $\mathrm{E25}$, $\mathrm{D22}$, $\mathrm{D20}$ which are in turn functions of the directly design-process-related model inputs[16]. Perhaps of somewhat lesser interest in the typical application, we mention in passing that the corresponding propagation of evidence in the reverse direction in the form of the λ-messages $\mathrm{E17}$ and $\mathrm{G24}$ satisfies a similar relation, except, this time we multiply $\mathrm{E17}$ by the *transpose* of the same matrix M in order to transmit evidence from the top part of the net in the reverse direction and so obtain the the message $\mathrm{G24}$ that summarises the influence of evidence from the top and middle parts of the net on updated beliefs about the variables directly associated with the true Quality the Requirements document.

One advantage of a symbolic implementation of the polytree updating algorithm for this net is that we can easily inspect the detailed workings of our model assumptions and observations through examination and presentation of the values of intermediate quantities such as messages, and other intermediate quantities involved in these, such as the matrix M we have just identified. Taking one case from Figure 3 as an example, with the observed values in row 12 of this figure, the matrix M takes the value

$$\mathrm{M}_{qu} = \sum_{a,d,g,r} \mathrm{E20}_{q\,u\,r\,g\,d\,a}\,\mathrm{G15}_r\;\mathrm{I11}_{s\,r}\,\mathrm{I18}_{t\,r}\,\mathrm{E29}_{g\,\mu\,h}\,\mathrm{E38}_h\,\mathrm{C24}_{d\,e\,f}\,\mathrm{A28}_e\,\mathrm{C33}_f\,\mathrm{C16}_{a\,b\,c}\,\mathrm{A13}_b\,\mathrm{A21}_c \tag{7}$$

where here, if we compare with the expansions of the four π-messages as contained in (6), the Σ-signs indicating summation over the state-spaces of the seven observed nodes in this part of the net have been removed, as justified on p24, because in this example all of our direct node observations are full observations of a node state. The observation vector symbols, $\mathrm{I14}$, $\mathrm{I21}$, $\mathrm{E41}$, $\mathrm{A31}$, $\mathrm{C36}$, $\mathrm{A16}$, $\mathrm{A24}$, have too been removed, because each of their single non-zero components is assumed to be identified by an assignment of a state to the now free variables which were previously the dummy summation variables. I.e., interpret the RHS of (7) as using the assignments s=*'Good'*, t=*'>1 Similar Systems Licensed'*, h=*'Yes'*, e=*'Adequate'*, f=*'High'*, b=*'Low'*, c=*'No'* from row 12 of Figure 3. With these observations, we obtained

$$M = \begin{bmatrix} .648761 & .207397 & .176486 \\ .258643 & .624433 & .276069 \\ .0925957 & .168171 & .547444 \end{bmatrix}$$

where the states of 'Design Process Performance' are the row names, indexed by q, and the states of 'Quality of Requirements' are the column names, indexed by u, in each case given in the order of 'increasing favourability' as they are listed in §2.2. Continuing to use the observations from

[16]—remembering that in order for our polytree-based analysis to be valid, these must include full observation of the state of the 'Problem Complexity(licensee)' node, with a unit vector input for $\mathrm{G31}$.

row 12 of Figure 3 now for the bottom and top parts of the net results in the 'incoming' messages to the middle part of the net[17]

$$\mathrm{G23} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \qquad \mathrm{E17} = \begin{bmatrix} .0737048 \\ .232889 \\ .693406 \end{bmatrix}$$

The resulting two messages which leave the middle part of the net are

$$\mathrm{E16} = \sum_u \mathrm{M}_{q\,u}\mathrm{G23}_u = \begin{bmatrix} .648761 \\ .258643 \\ .0925957 \end{bmatrix} \qquad \mathrm{G24} = \sum_q \mathrm{M}_{q\,u}\mathrm{E17}_q = \begin{bmatrix} .190030 \\ .305930 \\ .504040 \end{bmatrix}$$

We have printed these to 6 significant digits, though using our Maple implementation, arbitrary accuracy is possible, which may turn out to be desirable if very small probabilities are introduced into NPTs[18].

Supposing we wish to ask questions about the sensitivity of these numbers to our particular model inputs, the symbolic implementation now allows us to produce analytic answers to any question of this sort. We can replace any number of numerical model inputs (i.e. observation vectors, or NPTs) by parametric expressions of arbitrary form, and examine the resulting form of arbitrarily selected π- or λ-messages, or the π or λ-vectors at arbitrary nodes (including the normalised component-wise product of a pair of these latter quantities, which is identical to the distribution of a node, as updated to take account of all evidence input to the model). We intend to give several examples of these sorts of symbolic outputs as part of a project 'demonstrable' based on symbolic analysis of evidence propagation in this net. To provide one example here, we suppose it was the case that the expert was confident in the observations and NPTs producing the four π-messages, $\mathrm{F20}$, $\mathrm{E25}$, $\mathrm{D22}$, $\mathrm{D20}$ but was less confident in the construction of the large 'Design Process Performance' NPT described in the appendix. One obvious question here concerns the sensitivity of the model output to the choice of values for the k-parameters, used to incorporate the influence of the evidence concerning design guidelines, and allocation of resources to the design process. If we hold the observations within the middle part of the topology at their values as listed in row 12 of Figure 3, hold the value of the k_g parameter, used to take account of 'Design Guideline' evidence to the value of 0.1 used in the appendix (or to zero, as indicated there), but replace the k_r-matrix by the following symbolic version

$$K_r = \begin{bmatrix} 0 & 0 & 0 \\ k_{2,1} & k_{2,2} & k_{2,3} \\ k_{3,1} & k_{3,2} & k_{3,3} \end{bmatrix}$$

then when the resulting symbolic expression (see Appendix) for the DPP NPT is substituted into the symbolic implementation of the propagation algorithm, this outputs the resulting symbolic value of the matrix M

$$M = \begin{bmatrix} .623795 & .150877 & .116870 \\ .276876 & .669164 & .296184 \\ .099329 & .179959 & .586946 \end{bmatrix} + k_{2,1}\begin{bmatrix} .000724 & .002534 & .002896 \\ -.000362 & -.001810 & -.001810 \\ -.000362 & -.000724 & -.001086 \end{bmatrix} +$$

$$k_{2,2}\begin{bmatrix} .005793 & .011586 & .013033 \\ -.004344 & -.008689 & -.008689 \\ -.001448 & -.002896 & -.004344 \end{bmatrix} + k_{3,1}\begin{bmatrix} .016784 & .039163 & .047555 \\ -.011189 & -.033568 & -.011189 \\ -.005595 & -.005595 & -.036366 \end{bmatrix} + k_{3,2}\begin{bmatrix} .089515 & .201409 & .201409 \\ -.067136 & -.156651 & -.067136 \\ -.022379 & -.044758 & -.134273 \end{bmatrix}$$

[17] we will normalise messages when printing them here to assist with interpretation

[18] For our first pass elicitation, we have tended to use zero probabilities within NPTs in several places, which explains the unit vector value obtained for the π-message $\mathrm{G23}$

Giving a sense of how sensitive this matrix is to choice of k_r-parameters. The calculation of symbolic derivatives and the presentation of plots of the resulting sensitivity of various goal functions can likewise be obtained.

To give one example, we reset all the ks to their elicited values, as given in the appendix, except for the single k corresponding to 'Past Competence of Designers'=Good 'Problem Complexity (manufacturer)'=Moderate, (denoted $k_{3,2}$ above) which we leave free to vary. We then plot the updated distribution of the DPP node as a function of this k, using the observations from row 12 of Figure 3. The probabilities here are actually rational functions of k (of first order on both

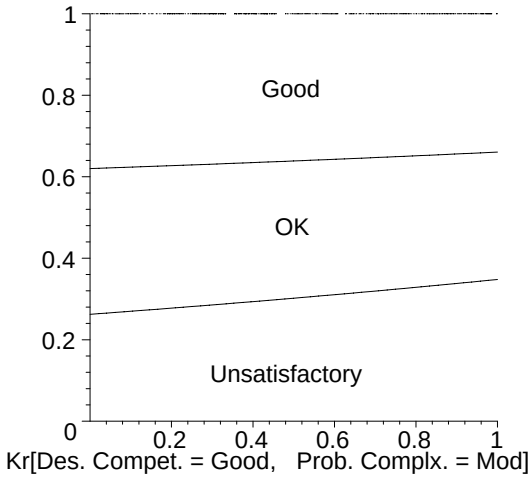

Figure 4: Updated Distributions of DPP Plotted as a Function of symbolic NPT parameter $k_{3,2}$

the numerator and the denominator). But they can be seen to be approximately linear, for these values of the model observations. We calculated this expression automatically using the Maple implementation of the polytree algorithm. We could do the same for any π-vector, λ-vector, or NPT associated with any model variable.

4.4 Some Planned Extensions of this Symbolic BBN Analysis

Once analytic expressions have been obtained, for model outputs, it would be possible to use a search algorithm to investigate the maximum deviation in some important model output achievable by allowing all of a subset of the model inputs to vary arbitrarily within given intervals, such as fixed percentage ranges. Here the availability of explicit analytic expressions for such dependencies suggests the use of analytic differentiation methods in obtaining global maxima and minima in such problems.

Such experiments will help us to understand better the level of precision that we would like from the domain expert in the numbers we request them to produce.

Given the general linear vector 'function-composition' form of the π- and λ-expansions at nodes, it seems likely that interesting expressions for analytic derivatives could be obtained in the form of Hessian matrices of an individual output probability vector as a function of an individual vec-

tor input (whether the latter were an observation likelihood vector, or one column of an NPT). Using these, directional derivatives (in directions that retain the normalisation of a probability distribution, i.e. directions normal to the vector $\langle 1 \rangle$) seem likely to be of interest. Obvious questions, involving consideration of these quantities, would include the discovery of the 'direction' of perturbation of some NPT column which would result in the fastest rate of increase of some output probability, together with an expression for that rate.

5 Discussion and Conclusions

We regard the process of BBN construction as an iterative process of building successive prototypes, examining their outputs (e.g. via sensitivity analyses of various kinds) giving feedback to domain expert, leading to rethinking and adjusting of the BBN to fit the expert's model and assumptions, and of the model and assumptions themselves when the analysis leads the expert to challenge and then refute his initial beliefs.

The goal is for this process to converge towards increasingly more realistic objective models, with ever improving consensus between different domain experts about which are the more appropriate models; or at least, failing this, towards increasing clarification of exactly how and where different schools of thought over this question differ, and increasing confidence that those differences are 'real', rather than consequences of communication and terminological problems.

Even without such convergence, the development of a BBN is likely to be a useful exercise for the individual domain expert, providing a formalism for them to examine the consistency of their own subjective opinions[19]. But in any case, the phases of elicitation and validation are bound to be necessary in any use of BBNs, and actually to overlap and depend on similar methods: after an initial phase of direct elicitation, analysis of the resulting BBN and feedback to the expert about its implications are necessary even just to assure that the BBN represents the expert's 'intuitive' beliefs. This step may be considered as part of the elicitation process, or a first phase of validation against the expert's own initial beliefs; subsequent phases will then include validation against additional knowledge available to the expert or to other experts.

In this report, we have discussed a few methods for initial, direct elicitation and for validation (in this wider sense of the word):

- production of large NPTs via elicitation of simple relations between the distributions of the relevant variables;
- direct feed-back and visualisation from use of the BBN 'as is' with an automatic BBN tool;
- algebraic descriptions of the model specified by the BBN, with potential for use in several ways, for instance:

[19] The question of how to assist the experts in choosing BBN structures that are likely to converge (quickly if at all possible) to consensual descriptions of the dependencies among the variables of interest (or at least to descriptions that will clarify the issues of disagreement) is clearly important. Some steps in this direction have been made in the SERENE project [Neil, Galliers 1997].

- as demonstrated here, studying the dependencies of the probabilities of interest on parameters used for specifying the NPTs in functional form;
- fast sensitivity analyses, to clarify which parts of the BBN need special attention in refinement and validation;
- using optimisation algorithms to prove that some form of prediction is the 'most pessimistic' possible, given certain shared beliefs about the BBNs;
- in general, proving special properties of a BBN, which may simplify the task of collecting evidence or of validating the BBN.

The algebraic method has clear potential, but also presents serious mathematical challenges. Our preliminary exploration has produced a machine-assisted solution suitable for our BBN (and any BBN for which reduction to a "polytree" is a practical step).

In many cases, safety assessment can benefit from the application of fairly simple BBNs. These can be used to clarify the issues concerning the use of some specific evidence (e.g., the BBN in [Neil, Littlewood, Fenton 1996] can be used to show how the inference to be drawn from software defect counts would vary with the testing effort employed and with assumptions about the effectiveness of this effort) or to describe a "backbone" argument, connecting few essential items of evidence, with NPTs filled on the sole basis of extensive empirical evidence or scientifically accepted reliability models (an example is in [Delic, Mazzanti, Strigini 1997]). Tools such as the ones investigated here would be valuable in the construction and analysis of such simple BBNs. The BBN in this case study, however, is more complex and involves many "soft" issues, where, for lack of (empirical or theoretical) knowledge, the precise form of the dependency between some variables would be difficult to define satisfactorily. It would be unlikely that multiple experts would agree on this detailed specification, and even a single expert may have serious difficulties in specifying his genuine belief in such detailed a fashion. Here, the methods we have discussed may help in attaining one of several goals:

- By analysing the implications of his first attempt at defining NPTs, and proceeding by successive refinements, an expert may be able finally to specify his own belief in the dependencies in question to the required level of detail. This part of the safety argument would then rely on the reliability of the intuitive judgement of the specific expert;
- the refinement process may lead to defining what needs to be investigated so that the dependence in question can be specified and justified on scientific bases, so that consensus (or informed discussion) about this component of the safety argument becomes possible among experts;
- the "soft" relationships are such that experts cannot hold such detailed beliefs as required to complete the BBN, but "coarser grained" beliefs will actually be accepted by scientific opinion: for instance, that observing a higher value of variable A implies a "stochastically higher" value for variable B. These exploratory methods may then be used to verify whether such "coarse" beliefs are sufficient to warrant conclusions of practical utility. E.g., one could prove that if a given "coarse" belief *C* about a certain dependency is true, then a specified way of artificially completing the description of the dependency (adding "spurious" information in the pertinent NPTs) is guaranteed not to err on the side of optimism:

that is, that predictions thus obtained are the most pessimistic that a consistent observer, sharing belief *C*, could obtain from the evidence available.

A general issue that we have not addressed is the 'validation' of the BBN approach itself, i.e., determining in which circumstances, and with which practical assistance by tools like those discussed here and by BBN experts, it can realise its inherent advantages (support for detailed analysis and machine-checked mathematical consistency) when used by safety experts in their customary working conditions. We have preliminary responses to the effect that the BBN approach is useful, at least for the purpose of self-clarification indicated above, but the general question can only be addressed via the accumulated experience of multiple uses. Individual case studies can only provide proofs of feasibility, and guide the development of auxiliary tools as described in this report. An example of the practical questions that can only be addressed by more practical and long-term use is that of the risk of 'tunnel vision': any systematic method of analysis risks making the analysts prone to ignoring any factors that are not considered by the systematic method. In the case of Bayesian approaches to uncertain prediction, the question that this general problem raises is whether the requirement that the assessors completely describe the sample space in advance of observations will prevent them from taking proper account of unexpected evidence. For BBNs, there may be concerns about evidence that experts choose not to represent by any variable in the BBN, because they know that observations of this evidence are usually unavailable or non-significant. Were the rare case to occur in which significant evidence of that type becomes available, would the experts ignore it because the process of constructing a BBN has made them less sensitive to evidence that is not formalised in the BBN? More general questions concern the effectiveness of BBNs as aids for overcoming cognitive bias or group effects [Kahneman, Slovic, Tversky 1982, Wright, Ayton 1987, Strigini 1994]. These problems, avoided at the level of using the BBN by applying the formal rules of probability, may yet affect the way the BBN itself is built, to a degree that will vary with the context of application, the professional backgrounds of the experts, etc.

In conclusion, in this case study we have explored some of the issues that arise in building and validating BBNs for safety assessment, and produced some useful tools to support these phases. Developing the BBN has been useful for the expert to question and analyse his criteria of judgement. Further experimentation with feedback methods and support tools will also help us to develop improved guidelines to assist choices about the topology and complexity of a BBN, so as to facilitate its refinement and its use in communication between experts.

Appendix: Construction of the Design Process Performance Node Probability Table

Here we describe our process of constructing a 'Design Process Performance' NPT for the BBN described in this report. The 'Design Process Performance' node in our BBN has 5 parents (see Figure 1), three of which are three-state nodes (the states listed in §2.2). The other two parents each have only two states. The resulting requirement is for an NPT with $108 = 3^3 \times 2^2$ parent state combinations, each of which must produce a specified numerical conditional distribution over the three states of 'Design Process Performance', i.e. the use of a numerical belief propagation tool with this net requires us to produce 216 independent numbers in order to define the required

NPT. This task can become confusing for the domain expert. There is a demanding requirement to ensure consistency with intuitively expected trends between these numbers through purely case-by-case numerical elicitation. Although, no doubt, it would be possible to produce tools which would ease a task of this kind, through providing flexible feedback of subsets of these numbers, it seems attractive to attempt, instead, to formulate some systematic parametric model for this NPT, which would explicitly embody the trends by which the expert would understand the structure that such an NPT ought to have. We performed this task using the following steps, approximately in the order listed as follows:

- To begin with, the expert was able to order the parent nodes in what he felt was an order of decreasing 'significance' in terms of the size of each parent node's influence on, or statistical association with, the value of 'Design Process Performance'[20]. The order thus obtained was
  - Quality of Requirements
  - Past Competence of Designers
  - Problem Complexity (manufacturer)
  - Adequacy of Project Resources
  - Actual Advantage Achieved by Compliance with Design Guidelines.

  In recording this ordering, we must acknowledge that such an ordering is, at least at this stage, a vague and informal notion. Attempts to treat it more rigorously would raise all sorts of questions which we are deliberately ignoring here. For example, it is easily conceivable that the order of 'significance' of knowledge about two parents may reverse depending on *which* two respective states these parents are assumed known to have. Equally, the relative significance of two parents, even when discussing a fixed pair of their respective states, may reverse depending on the state of the expert's knowledge about others of the five parent nodes. It is probable that acceptable interpretations of these sorts of complexities regarding the behaviour of comparative significance-orderings of hypothetical knowledge of the states of different parent nodes could be formulated more rigorously in the form of mathematical properties of the NPT, perhaps using concepts analogous to partial and conditional correlation. We have not attempted to formally address such issues in the current report.

- A table of 27 columns, corresponding to the 27 possible state combinations of the three most significant parent nodes in this list, was produced. The domain expert was able to fill in this reduced size NPT, using these combinations, based on the assumption that the two other parent nodes, considered to be of lesser significance, are known to be fixed in their most favourable states. I.e., numbers were elicited for the conditional probability distribution of the 'Design Process Performance' node, under the hypothetical conditioning assumptions that

[20]Note that this does not necessarily comply with the order of significance according to which evidence actually obtainable in a real case would impact on beliefs about the 'Design Process Performance', because in practice there will be the confounding factor of different levels of availability of evidence regarding the value of different parent nodes. The order under discussion here is an order of significance, in the *hypothetical* situation that the expert was able to *know with certainty*, or, in our previous terminology, to 'fully observe' the value of one or another of these parent nodes.

- the three nodes at the top of the above list were assumed known to take, in turn, each one of their 27 state-combination; and, for each of these 27 combinations,
- 'Adequacy of Project Resources' was assumed known to have the value 'Adequate'; and 'Actual Advantage Achieved by Compliance with Design Guidelines' was assumed known to have the value 'Yes'.

This 27×3 table was filled in with numerical probabilities by a process involving consideration of selected small groups of columns, such as pairs and triples. The expert attempted to produce the correct trends by eye according to intuition concerning the problem domain. Some iteration and adjustment and readjustment of the numbers was employed here.

- A set of informal verbal rules were then devised which it was felt should govern the impact on these 27 numerical conditional distributions of changing either one or both of the states of the two less significant two-state nodes, contained at the bottom of the list above. The rules eventually settled on were as follows:-
  - If it is known that 'Past Competence of Designers' = 'Good', then the value of 'Actual Advantage Achieved by Compliance with Design Guidelines' has no impact on the distribution of 'Design Process Performance'.[21]
  - The size of the *impact on our beliefs*, about the 'Design Process Performance' node, of the *value* of the 'Adequacy of Project Resources' node should increase as the *value* of 'Problem Complexity (manufacturer)' increases (in the sense of the state-space ordering 'Complex/Difficult' $>$ 'Moderate' $>$ 'Simple/Easy').
  - The impact on our beliefs, about the 'Design Process Performance' is greater for the 'Adequacy of Project Resources' node than it is for the 'Actual Advantage Achieved by Compliance with Design Guidelines' node. (I.e. these two nodes are indeed significantly ordered as indicated by their relative positions in the list above.)
  - If it is known that 'Past Competence of Designers' = 'Low' then the values of neither the 'Actual Advantage Achieved by Compliance with Design Guidelines' node, nor the 'Adequacy of Project Resources' should have any impact on beliefs about the 'Design Process Performance' node[21].
  - Neither the size of the impact of the 'Adequacy of Project Resources' node (on beliefs concerning the 'Design Process Performance' node) nor that of the 'Actual Advantage Achieved by Compliance with Design Guidelines' node is affected by the value of the 'Quality of Requirements' node.

At this stage, our approach was to attempt to formulate a simple parametric model for the effect of the two least significant parent nodes which would be faithful to these intuitive rules. We mention in passing that in recent literature there have been some other approaches than to go as far as to generate a full numerical NPT, as we do here, in this sort of situation. See the 'qualitative' BBN work of [Wellman 1990, Druzdzel, van der Gaag 1995].

[21] These two rules—the first and fourth in this list—are examples of a restricted kind of conditional independence relation. The sense in which they are not full conditional independence relations, of the kind that can be built into the BBN topology, is that the analogous statements required for that (e.g. with 'Average' replacing 'Good' in the first rule), are not all assumed to hold.

It is fairly clear that those of the above rules which refer informally to the 'size of the impact' on the 'Design Process Performance' node imply the use of a stochastic ordering of some kind. That is, we need to be able to require that the conditional distribution of the 'Design Process Performance' node should be 'increased' or 'decreased' by different amounts that can be compared in the sense of an ordering of probability distributions on a 3-state space. The state space of the 'Design Process Performance' node is clearly totally ordered, 'Unsatisfactory' $<$ 'OK' $<$ 'Good', which leads to an immediately appealing definition of a partial ordering of distributions on this node: Two distributions p_1 and p_2 would satisfy $p_1 < p_2$ if and only if it was possible to transform p_1 into p_2 by a series of steps, each consisting of moving a positive amount of probability mass from a lower to a higher state in this state-space ordering. In terms our graph in Figure 3 this simply means that one conditional distribution is 'greater than' another if *both* of the lines in the graph (we are referring to the upper graph of the 'Design Process Performance' distributions) are lower at the first distribution. If, instead, one line is higher, and the other lower, then the two distributions are incomparable, in the sense of this stochastic ordering. (In fact, in Figure 3, the 'Design Process Performance' distributions are in strictly increasing order as we move from left to right through the parent state combinations.) It is also of interest to interpret this ordering, in the case of distributions on a state-space of size 3, in terms of the 'triangle diagrams' we showed in [Fenton, Littlewood et al. 1998, Figure 3, p506][22]. In terms of this diagram, in which each distribution corresponds to a point in the triangle, two distributions are ordered (with the left-most one being the greater, in terms of a 'favourability' ordering) if the straight line joining them has gradient between $\pm 60^\circ$, and otherwise (if the line is any steeper than this) are incomparable. Of course, for the Design Process Performance' node the state names at the three corners would be appropriately substituted ('many issues' $\rightarrow$ 'Unsatisfactory', a 'few issues' $\rightarrow$ 'OK', '0 issues' $\rightarrow$ 'Good'). Speaking in terms of the resulting triangle diagram for the DPP node, we decided that we could adequately represented the effect of either of the two parent nodes of lesser significance being assigned its unfavourable value (instead of its favourable value, as had been assumed in eliciting the 27 column, reduced NPT) by a 'movement' of the resulting conditional distribution of 'Design Process Performance' in a straight line towards the 'bottom' element of our stochastic ordering. The bottom element in the ordering is the righthand, bottom corner of the triangle, corresponding to a certainty of 'Unsatisfactory' 'Design Process Performance'. In fact, we regard our decision here as somewhat arbitrary, and a subject for future review. It seems that the deterioration in parent node conditions ought to result in movement of the 'Design Process Performance' in *some direction* which corresponds to a decrease in optimism of or beliefs with respect to this ordering; but this is a *partial* ordering, and there are *many* such directions of movement which constitute a reduction in optimism, and from which we have chosen one. For example we might as well instead have moved the distribution in the direction of motion directly *away* from the *top* element (the left-hand bottom corner of the triangle). We decided to allow the *distance* of movement to be parameterised as a fraction of the original distance from the bottom distribution. If 'Adequacy of Project Resources' changed its state from 'Adequate' to 'Inadequate' we would allow this to push our conditional distribution of 'Design Process Performance' along the straight line between its original position and the bottom distribution, by a fraction k_r of its starting distance from the bottom distribution. Alternatively, changing the state of parent node 'Actual Advantage Achieved by Compliance with Design Guidelines' from 'Yes' to 'No' would move the distribution towards the bottom distribution in the ordering by a fraction k_g of

[22]The name of the node whose NPT is illustrated in [Fenton, Littlewood et al. 1998] has now been changed to the more explicit 'Issues raised by Licensee Verification and Unresolved'.

its original distance from it. These two operations are commutative: if *both* lesser significant parent nodes are changed from favourable state, to unfavourable state, we would perform these to operations in succession, in either order, and achieve a common result.

We have chosen to motivate, this definition of our parametric representation of perturbations in the 27-column, reduced NPT geometrically, regarding probability distributions on a three-state space essentially as vectors confined to a 2-dimensional triangular surface. Algebraically, what we have just described is equivalent to the transformation from a distribution p to a 'smaller', or 'less favourable/optimistic' conditional distribution p' defined by the equation

$$\begin{aligned}\langle\, p'(\text{Unsatisfactory}),\, p'(\text{OK}),\, p'(\text{Good})\,\rangle = \\ \langle\, 1-(1-k)(1-p(\text{Unsatisfactory})),\, (1-k)p(\text{OK}),\, (1-k)p(\text{Good})\,\rangle\end{aligned}$$

where

$$k = \begin{cases} 0 & \textbf{if } \text{APR} = \text{'Adeq.'}, \textbf{ and } \text{AAACDG} = \text{'Yes'}, \\ k_r & \textbf{if } \text{APR} = \text{'Inad.'}, \textbf{ and } \text{AAACDG} = \text{'Yes'}, \\ k_g & \textbf{if } \text{APR} = \text{'Adeq.'}, \textbf{ and } \text{AAACDG} = \text{'No'}, \\ 1-(1-k_r)(1-k_g) & \textbf{if } \text{APR} = \text{'Inad.'}, \textbf{ and } \text{AAACDG} = \text{'No'} \end{cases} \tag{8}$$

Finally, we completed our realisation of our list of rules for this 'depressing perturbation' of our beliefs about DPP by allowing both degradation parameters k_r and k_g to depend on the values of the parent nodes 'Problem Complexity (manufacturer)' and 'Past Competence of Designers' as indicated in the following table

$k_r \mid k_g$		Problem Complexity (manufacturer)		
		Complex/Difficult	*Moderate*	*Simple/Easy*
Past	*Low*	$0 \mid 0$	$0 \mid 0$	$0 \mid 0$
Compet.	*Average*	$.4 \mid .1$	$.3 \mid .1$	$.1 \mid .1$
Designers	*Good*	$.3 \mid 0$	$.2 \mid 0$	$.1 \mid 0$

References

[Abdel-Ghaly, Chan, Littlewood 1986] A. A. Abdel-Ghaly, P. Y. Chan, and B. Littlewood. Evaluation of Competing Software Reliability Predictions. *IEEE Transactions on Software Engineering*, 12:950–67, 1986.

[Andersen, Olesen et al. 1989] S. K. Andersen, K. G. Olesen, F. V. Jensen, and F. Jensen. HUGIN—A Shell for Building Bayesian Belief Universes for Expert Systems. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 1080–84, Detroit, 1989.

[CACM1995] Real-World Applications of Bayesian Networks. *Communication of the ACM, Special Issue*, 38(3):24–57, 1995.

[Delic, Mazzanti, Strigini 1995] K. A. Delic, F. Mazzanti, and L. Strigini. Formalising a software safety case via belief networks. Technical report, CSR, City University, London, July 1995. Tech. Report SHIP/T/046 of E.U. Environment Program SHIP project (EV5V 103).

[Delic, Mazzanti, Strigini 1997] K. A. Delic, F. Mazzanti, and L. Strigini. Formalising Engineering Judgement on Software Dependability via Belief Networks. In *DCCA-6, Sixth IFIP International Working Conference on Dependable Computing for Critical Applications, "Can We Rely on Computers?"*, Garmisch-Partenkirchen, Germany, 1997.

[Druzdzel, van der Gaag 1995] M. J. Druzdzel and L. C. van der Gaag. Elicitation of Probabilities for Belief Networks: Combining Qualitative and Quantitative Information. In *Proceedings of the* 11^{th} *Annual Conference on Uncertainty in Artificail Intelligence*, Montreal, August 1995.

[Fenton, Littlewood et al. 1998] N.E. Fenton, B. Littlewood, M. Neil, L. Strigini, Wright D.R., and P.-J. Courtois. Bayesian Belief Network Model for the Safety Assessment of Nuclear Computer-based Systems. Project deliverable, CSR, City University, London, January 1998. Second Year Project Deliverable of ESPRIT DeVa project 20072.

[Jensen 1996] F. V. Jensen. *An introduction to Bayesian networks*. UCL Press, University College London, 1996.

[Kahneman, Slovic, Tversky 1982] D. Kahneman, P. Slovic, and A. Tversky, Editors. *Judgment Under Uncertainty: Heuristics and Biases*. Cambridge University Press, 1982.

[Kim, Pearl 1983] J. H. Kim and J. Pearl. A Computational Model for Combined Causal and Diagnostic Reasoning in Inference Systems. In *Proc.* 8^{th} *International Joint Conf. on AI*, Karlsruhe, Germany, 1983.

[Lauritzen, Spiegelhalter 1988] S. L. Lauritzen and D. J. Spiegelhalter. Local Computations with Probabilities on Graphical Structures and their Application to Expert Systems. *Journal of the Royal Statistical Society, Series B*, 50(2):157–224, 1988. with discussion.

[Neil, Galliers 1997] M. Neil and J. Galliers. Task 1.3 Report. Technical report, CSR, City University, London, 1997. ESPRIT Framework IV IT Program Project SERENE 22187, Doc. No. SERENE/1.3/CSR/3010/R/2.

[Neil, Littlewood, Fenton 1996] M. Neil, B. Littlewood, and N. Fenton. Applying Bayesian Belief Networks to Systems Dependability Assessment. In *Proceedings of Safety Critical Systems Club Symposium*, pages 84–95, Leeds, 1996. Safety-Critical Systems Club, Springer-Verlag.

[Pearl 1988] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Mathematics and Its Applications. Morgan Kaufmann, San Mateo, California, 1988. Revised 2^{nd} printing 1991.

[Strigini 1994] L. Strigini. Engineering judgement in reliability and safety and its limits: what can we learn from research in psychology? Technical report, CSR, City University, London, 1994. Tech. Report SHIP/T/030 of E.U. Environment Program SHIP project (EV5V 103).

[Wellman 1990] M. P. Wellman. Fundamental Concepts of Qualtiative Probabilistic Networks. *Artificial Intelligence*, 44:257–303, 1990.

[Wright, Ayton 1987] G. Wright and A. Ayton. *Judgemental Forecasting*. John Wiley, Chichester, 1987.