User-Centered Methods for Rapid Creation and Validation of Bayesian Belief Networks

Jonathan Pfautz, Zach Cox, Geoffrey Catto, David Koelle,
Joseph Campolongo, Emilie Roth
Charles River Analytics
625 Mt. Auburn St.
Cambridge, MA 02138

Abstract

Bayesian networks (BN) are particularly well suited to capturing vague and uncertain knowledge. However, the capture of this knowledge and associated reasoning from human domain experts often requires specialized knowledge engineers and computational modelers responsible for creating BN-based models. Through our experiences in applying BN modeling techniques across application domains, we have analyzed how these models are constructed, refined, and validated with domain experts. From this analysis, we have identified potential simplifying assumptions and used these to guide the design of computational and user interface methods that support the rapid creation and validation of BN models.

1. INTRODUCTION & MOTIVATION

A Bayesian network (BN) (Jensen, 2001; Pearl, 1988) is a probabilistic model used to reason under uncertainty. BNs have been used by the authors, their colleagues, and numerous other researchers to reason about a wide variety of phenomena (e.g., computer vision (Rimey & Brown, 1994), social networks (Koelle et al., 2006), human cognition (Guarino et al., 2006; Glymour, 2001), and disease detection (Pang et al., 2004)) with notable success. These successes have led to increased interest in BNs as computational methods for domains where the representation of expert reasoning and knowledge is paramount. For example, a knowledge engineer may wish to capture an expert's reasoning about how incoming information should be classified based on the source of the information and associated environment factors (prior work has discussed issues with this elicitation (Tabachneck-Schijf & Greenen, 2006; Mathias et al., 2006)). Or, an expert may want to directly externalize his or her reasoning so that it could be encoded and automated.

However, the expression of a model (i.e., creating a "view of that model") by either a knowledge engineer or a domain expert using most current tools is relatively cumbersome, since the full representation of Bayesian network computation is beyond the mathematical sophistication of many domain experts and some knowledge engineers (who, in some cases, may have backgrounds in Cognitive Science and Psychology, rather than Mathematics or Computer Science). Even with an effort to reduce the complexity and power of a BN representation, there is still the problem of constructing conditional probability tables (CPTs). Since the number of distributions that must be expressed in a CPT grows exponentially in the number of parents and the number of states per parent, a knowledge engineer or domain expert can end up responsible for expressing a vast number of what may end up to be meaningless separate distributions (e.g., a child with 3 states and 7 parents that have 4 states each would require the entry of 49,152 probabilities!). Clearly, this exponential explosion, combined with the underlying sophistication of the representation, presents a challenge.

To some degree, the problem has been recognized by the research community and the subsequent development of "canonical models" that reduce the number of parameters needed to specify a CPT. A canonical model (Diez & Druzdzel, 2006) makes a specific assumption about the type of relationship between a node and its parents. This assumption results in many fewer parameters being needed to specify an entire CPT. There are many types of canonical models used in practice and each assumes a different relationship. Noisy-OR (Henrion, 1989; Pearl, 1988), Noisy-MAX (Diez & Galan, 2003), and Influence Networks (IN) (Rosen & Smith, 1996a; Rosen & Smith, 1996b) are three commonly used canonical models. However, the assumptions underpinning these canonical models may make them less generalizeable, or may introduce other inconsistencies that decrease their utility. Furthermore, it is unclear what advances in user interface designs have been made as a result of these models.

In this paper, we describe our efforts to understand how knowledge engineers and domain experts express their

reasoning and knowledge with the goal of using this understanding to select simplifying assumptions for the construction and execution of Bayesian networks. From these assumptions, we have constructed both an underlying computational representation as well as a prototype user interface to further explore the degree to which users can more rapidly express their reasoning, as well as the degree to which those assumptions may be violated. This approach varies from that of Tabachneck-Schijf et al., (2006), who proposed a more formal user-centered process for knowledge engineering, in that our end goal is to create both computational and user interface methods to allow for *both* knowledge engineers and domain experts to rapidly create, validate, refine, and share their own and others' reasoning. We recognize that from a particular set of requirements, many possible user interface designs are possible; in this paper, we present our overall approach to design and development, and describe the resulting methods only to support our approach. In Section 2, we describe relevant background material. In Section 3, we introduce our approach to analysis of user cognition and decision-making across different application domains. In Section 4, we outline our results from these many analyses and their implications for model construction and validation. In Section 5, we describe the design of underlying algorithms and supporting user interfaces to address our understanding of how to best support model generation, testing, and refinement. Finally, in Section 6, we discuss implications for future research and development efforts.

2. BACKGROUND

A review of Bayesian network literature indicates a clear bias towards mathematical techniques for representation, inference, and learning instead of techniques for eliciting Bayesian networks from experts. The research that does exist related to Bayesian network elicitation focuses on obtaining the structure (random variables, their states, and the causal connections between them) of the Bayesian network, the conditional probability distribution parameters that quantify the network, or both.

Nadkarni & Shenoy (2000) demonstrate an approach for converting a causal map into a Bayesian network. A causal map is similar to the structure of a BN, where an expert draws directed links between concepts to represent causal relationships. Their approach focuses on elicitation techniques for the causal map and proper conversion techniques for tranlating it into a BN. They recommend using canonical models (such as Noisy-OR and Noisy-AND) to simplify parameter elicitation for nodes in the resulting BN with multiple parents. Without easy conversion between the causal map and the BN (and, indeed, bi-directional conversion), then the ability of an expert to rapidly test and validate a BN may be hindered.

Skaanning (2000) presents an automated tool for eliciting diagnostic BNs from domain experts. While the approach is focused on diagnostic tools for printers, it can be used in other domains that follow similar constraints. By asking the domain expert natural language questions, imposing strict constaints on the resulting BN structure, and eliciting reverse parameters (the probability of a cause given the effect), they greatly simplify the elicitation of both structure and parameters. Kraaijeveld et al., (2005) also present their GeNIeRate system that aids in eliciting diagnostic BNs from experts. They simplify the elicitation by assuming the BN has only three levels of variables and that all nodes use the Noisy-MAX canonical model. This approach removes the value of showing an expert an interactive graphical model where they can visualize the reasoning captured as they express it.

Neil, Fenton, & Neilsen (1999) build on recent research in Object-Oriented software design and Object-Oriented BNs to come up with a process similar to the spiral process of software engineering for building complex BNs from smaller BN parts. They define five of these parts, called idioms, in detail and describe how to use them in practice to build large BNs. A process-based approach to simplifying BN construction is indeed useful, and contrasts with our goal of enabling a domain expert to create their own model.

The Weighted Sum Algorithm (WSA) (Das, 2004) reduces the number of parameters to specify a CPT from being exponential in the number of parents to linear in the number of parents. In WSA, the domain expert need only specify one probability distribution over the child node's states for each state of each parent (instead of one such distribution for each combination of parent states). This distribution is conditional on the state of the parent and the most compatible state of each other parent. Therefore, joint effects of parents can still be taken into account while specifying a small number of parameters. After specifying these parameters for compatible parent configurations and a relative weight for each parent, the WSA is used to combine them into the full CPT. This method represents yet another approach to solving the underpinning mathematical issues with making a causal network easier to express. However, this work is focused on computational efficiency, not the ease with with the model can be expressed by a user.

Helsper et al. (2005) describe a method for eliciting qualitative information about the probabilistic relations in a BN from an expert using a dedicated eliciation technique, like {Tabachneck-Schijf, 2006 9201 /id}. This qualitative infromation is then used to constrain the probabilities learned from a small set of data that otherwise would be too small to provide accurate learned probabilities. Again, these approaches are important in cases where dedicated knowledge elicitation is the correct approach, but may be less useful when encouraging domain experts to express their own reasoning.

Helsper, van der Gaag, & Groenendaal (2004) list the three types of effects that parent nodes have on child

nodes: qualitative influence (QI), additive synergy (AS), and product synergy (PS). The specific values of these three effects in a BN constrain the actual values of entries in the CPT. To determine QI, AS, and PS, they propose using "case cards" which ask the domain expert to order the parent configurations by causal effect on the child node. Therefore, an domain expert need only specify an ordering of parent configurations without any conditional probabilities. This approach is currently constrained to nodes with only two states and nodes with only two parents. This approach also does not generate actual CPTs; it only constrains the values the CPTs can have given the higher-level qualitative concepts of QI, AS, and PS. Wiegmann (2005) provides a review of several methods for eliciting probabilities from domain experts as well as combining the probabilities elicited from multiple experts. Their approach in their fielded system is to use multiple elicitation techniques to improve the accuracy of the probabilities that the experts specify. These approaches represent a step towards our stated goal – understanding the way the user may want to express causal relations to provide them with a method for directly expressing those relations into a computational model. This work, more generally, may help to illustrate cases where models developed by domain-experts may be subject to biases or incorrect assumptions about the underlying comptuation.

3. APPROACH

Our goal across projects has been to identify and study the users and experts and their approaches to decision-making in different application domains, with the express purpose of aiding in that decision making with techniques such as BN modeling. To accomplish this goal in each project, we use a particular set of analytic methods generally referred to as Cognitive Systems Engineering (CSE).

The CSE community emerged as experience with the introduction of new technology demonstrated that increased computerization does not guarantee improved human-computer system performance (Woods & Dekker, 2000; Roth, Malin, & Schreckenghost, 1997; Woods, Sarter, & Billings, 1997). Poor use of technology can result in systems that are difficult to learn or use, can create additional workload for system users, or in the extreme, can result in systems that are more likely to lead to catastrophic errors (e.g., confusions that lead to casualties from friendly fire). CSE attempts to prevent these types of failures in the design and development of complex system by addressing design issues through careful analysis of the problem domain, the tasks to be performed by a human-computer system, and the limitations of both the human and the machine.

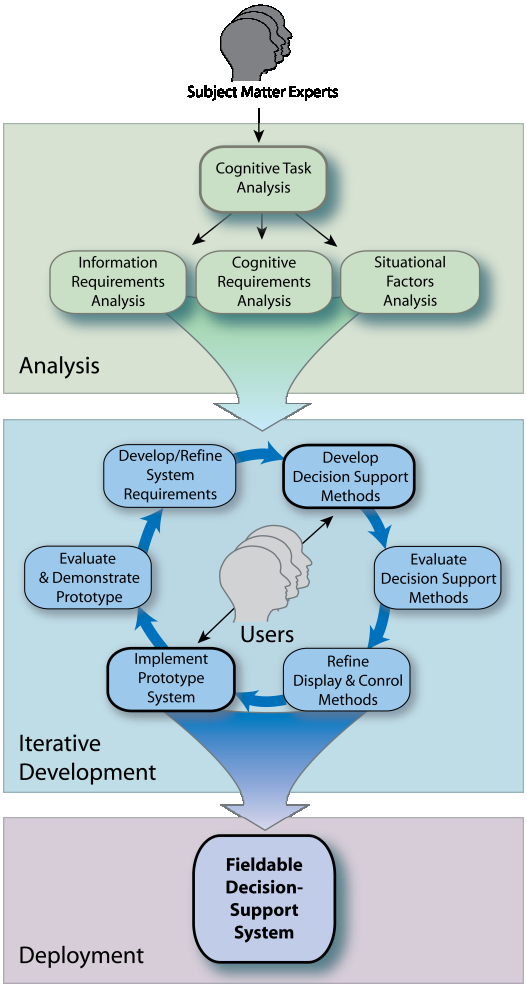

Figure 3-1: Cognitive Systems Engineering Process

While there are many different approaches to the analysis components of CSE, such as Cognitive Task Analysis (Schraagen, Chipman, & Shalin, 2000), Cognitive Work Analysis (Vicente, 1999), Work Centered Support Systems (Eggleston, Roth, & Scott, 2003), and Applied Cognitive Task Analysis (Militello & Hutton, 1998), these methods share a commitment to analyzing the cognitive and environmental demands imposed by the domain of practice and identifying implications for information, visualization, and decision-support requirements. CSE methods generally entail a multi-phase, iterative design approach that includes an analysis phase, a concept development and prototyping phase, and a user evaluation phase. The cognitive analysis phase typically employs knowledge elicitation methods such as interviews of domain practitioners and observations of work in context. These methods uncover the reasoning processes involved in making decisions and performing tasks in the domain and the challenges that arise (Potter, Roth, Woods and Elm, 2000; Roth and Patterson, 2005). The analysis phase supports the development of system requirements that can be used to prototype computational

support tools and user interfaces, including the knowledge elicitation needed to engineer formalized representations of user knowledge and expertise. The development of requirements and capture of expert knowledge is followed by subjective evaluations of a prototype system. These subjective evaluations lead to the development of more robust prototypes, which are then more rigorously evaluated. The results of the evaluation then aid in the refinement of the system requirements. This spiral development process will eventually converge on an optimal human-computer system (although time/cost constraints will influence the number of possible iterations) (Pfautz & Roth, 2005c). This development process is illustrated in Figure 3-1.

4. RESULTS

In applying this process across numerous projects, we have spent thousands of hours with hundreds of domain experts across a wide variety of applications, such as supporting the analysis of weather impacts (Lefevre, Pfautz, & Jones, 2005), military intelligence (Pfautz et al., 2006b) and command and control (Pfautz et al., 2006a). In these projects, we have observed some consistent patterns in how knowledge is (and can be) elicited by a knowledge engineer to develop models. Similarly, we have observed how experts tend to express their reasoning about the domain. In addition, we have identified ways in which we could improve our efforts to encode domain experts' knowledge and reasoning in BN models, and to work with domain experts to validate those models. This meta-analysis of our own development process led us to postulate some key features that may be aided by improved computational methods and associated user interfaces.

The first and foremost finding is that the power of a computational representation is seemingly proportionate to the ease with which expert knowledge can be both encoded and validated. That is, the ability to quickly create a model with domain experts, then to work through a set of cases within that model allows for a very rapid cycle of elicitation, representation, and validation. Rather than the model being reliant on the knowledge engineer's ability to *ex post facto* recall and extrapolate from the statements of the domain expert, the model can be more comprehensively and reliably based on expert knowledge and reasoning. An additional benefit to this ability to rapidly cycle through model development is that time with domain experts is typically constrained, and that multiple sessions would be otherwise required to develop, then refine, then validate a model.

A second finding was that while domain experts may not be conversant in computationally sophisticated technologies, they can be systematically guided to express their own knowledge and reasoning. This systematic guidance can come in the form of a knowledge engineer performing a structured interview, but it may also come in the form of the modeling tool itself. That is, the domain expert could conceivably develop the model herself, if the interface to the model were sufficiently simple (and presuming a knowledge engineer was present to provide some guidance). This concept of supporting domain experts in expressing their own models has additional benefits, in that a domain expert has an external record of their own reasoning, which in turn can be used in a collaborative decision-making process, shared and used as the basis for future reasoning, or independently validated by other experts.

These main findings were accompanied by more specific results about the actual construction of Bayesian networks by both our knowledge engineers and the domain experts with whom we work:

(1) The exponential growth of CPTs with the number of parents and parent states was cited most frequently as a cause for frustration. Few domain experts or knowledge engineers were willing to tackle the entry of extremely large CPTs. Domain experts facing this challenge often resorted to saying, "they'd just be making it up" at that level of detail to avoid the task.

(2) The lack of ability with many BN software packages to develop a network and immediately update beliefs (as a function of new nodes, states, CPTs, or causal links) slowed model development.

(3) "Evidence" and "belief" were commonly confused when shown simultaneously. These terms were also commonly confused in discussions about what a BN was representing.

(4) The expression of vague or uncertain knowledge or reasoning varied significantly between users. A great deal of speculation was used in the creation of the CPTs (more so than the relationships among variables)

(5) The selection of variables was a primary challenge. We discovered that expression could be guided towards primarily Boolean and/or Ordinal ranges, although in some cases, Categorical variables were used to simplify the network ("Categorical" refers to non-Boolean, non-Ordinal groups such as {apples, oranges, jet skis}). It was possible to create networks that represented identical reasoning, but that used a different selection of variables and variable types.

(6) Many networks were constructed with the assumption that parents were independent, or when parents had dependent influences, variable names and states could be relatively easily re-defined to preserve parental independence. Working towards networks with parental independence may also have helped to more explicitly represent un-stated or assumed influences among variables.

(7) In some cases, variables were selected such that they caused wholesale disregard for other variables (e.g.,

"if A, then I don't care about anything else, the answer is True".)

(8) The underlying reasoning, while mathematically correct, could be opaque and non-intuitive (particularly when certain non-linearities were exposed). This led to issues of trust in the model and the modeling technique.

These findings are the result of our analysis and experience, not a comprehensive, empirical set of evaluations. However, we would assert that they are highly valuable observations to be used as part of an effort to build better BN-based modeling tools and will, as part of our iterative approach to development, be subjected to more formal evaluation to test the limits of their application in BN modeling.

5. APPLICATION OF RESULTS

The results of this analysis were used in the development of Charles River Analytics' BNet™ suite of products (note: other BN tools (e.g., Elvira, GeNIe) may also perform a subset of features we developed – our goal in this paper is not to perform a product comparison, but simply to show how an analysis of common problems should systematically lead to user interface design elements). Each of the findings has resulted in a particular feature or set of features. For example, our finding that users typically wanted to see updates to beliefs whenever any type of change was made to the network led to the development of a "no-compile" or "mode-less" user interface (see (2) above). Other user interface methods were used to simplify, where possible, the completion of CPTs (e.g., supporting entry of multiple rows simultaneously, supporting cut and paste from spreadsheet applications (see (7) above)). However, while these improvements to BN modeling clearly improve the user experience for an experienced knowledge engineer, they fail to address the broader goal of making BN model construction a more interactive process with domain experts to whom the subtleties of BN representations are not as immediately important as the construction and validation of the model.

Therefore, we focused on two key components of a BN modeling tool that would support a more rapid model creation and validation cycle with domain experts and knowledge engineers, with the greater goal of working towards methods that would allow domain experts to comfortably and intuitively externalize their knowledge and reasoning. The first component consists of an underlying mathematical representation (Causal Influence Models, or CIMs) that is based entirely on BNs, but uses simplifying assumptions much like the canonical models described in Section 1. Compared to these canonical models, these simplifications are based primarily on the desire to support improvements to the user interface, rather than computational efficiency. The second component of our effort was to develop user interface methods that leveraged the results of our analysis and the power of the simplified computational model. Both of these components are described in more detail below.

5.1 CAUSAL INFLUENCE MODELS

One of the key findings from our analysis was that experts tend to express the degree to which certain factors influence the likelihood of other factors independently (and, where they do not consider these factors independently, the act of explicitly expressing these interdependencies leads to a model where new factors are created to represent the dependency) (see (6) above). Therefore, we started with the simplifying assumptions that each parent node influences the child node (causes it to be more or less likely) and these parents act independently. These assumptions, along with a procedure for combining parent influences into a full CPT, lead to the Causal Influence Model (described in full detail in (Cox & Pfautz, 2007).

In our CIM, the user first specifies a baseline probability distribution over the states of the child node. These baseline probabilities represent the *a priori* likelihood of the child states, without the effects of any of its parents. Next, the user specifies the influence that each parent state has on each child state. This influence is a number in the range [-1, +1] and represents the amount that the parent state increases or decreases the baseline probability of the child state.

Once the user specifies the baseline probabilities and influences, they are used to calculate all of the probabilities in the CPT. For each row and child state in the CPT, we combine the influence of each parent state in that row on the child state into the overall parent influence. While many combination functions are possible, we typically use the mean of the parent influences since it is a linear function causing positive and negative influences to balance each other out and therefore produces CPTs that represent linear relations (see (8) above). This overall parent influence is then used to either increase or decrease the baseline probability of the child state, and this result is used as the conditional probability of the child state given the parent states in that row.

The CIM requires a number of parameters that is only linear in the number of parents, as opposed to exponential for a CPT (addressing (1) above). The baseline probabilities and influences are also easily specified by domain experts since they do not involve the joint effects of numerous parent states. In our experience, a domain expert is much more likely to accurately specify a small number of simple parameters. This is the primary benefit of using the CIM instead of a full CPT. Of course, the user can always further refine the actual CPT later, by hand or with data, meaning that the full representational capabilities of the BN model are accessible as needed.

While the IN canonical model (Rosen & Smith, 1996c) also provides these same benefits, the CIM overcomes the IN's two primary drawbacks. First, the CIM can be used with any discrete node, regardless of the number or meaning of its states, while the IN is restricted to Boolean nodes only. This capability was based on our observation that, while experts can be guided to formulate their reasoning in purely Boolean terms, it is often more representative or concise to support Ordinal and Categorical terms. Second, the IN model uses a non-linear function to combine parent influences which has been known to produce unintuitive CPT entries in certain cases (e.g., where beliefs become overly sensitive to evidence from a particular parent). The CIM uses a simple linear function to combine parent influences, thus avoiding this problem.

For each parent, the user must specify a number of influences equal to the product of the number of parent states and child states for the CIM. While this is certainly better than the CPT, it can still result in too many parameters for a user to specify. We simplify the CIM even further by making assumptions about the types of the parent and child nodes and the type of influence that the parent has on the child. These assumptions allow us to completely specify the CIM using only a single parameter for edges between Boolean and Ordinal nodes, and for edges connected to a Categorical node only a single parameter for each state of the Categorical node is needed. For example, in a network with only Ordinal or Boolean variables, a child with 7 parents would require the entry of only 7 values. In a full BN representation, the user would need to specify a minimum (assuming only 2 states per variable) of 128 values.

5.2 USER INTERFACES

The CIM, and the underpinning assumptions that it uses, is based on the goal of improving interfaces to BN models to the level where they can be dynamically and interactively created with or by domain experts. This goal has led to the development of a number of specific user interface methods that have been incorporated into our BNet™ product suite for further evaluation and refinement. These methods are described in more detail below.

First, we addressed the need to allow for dynamic updating of the model as new links, variables, states, and evidence are created (see (1) above). This is supported by the underlying computation, but results in a network that actively animates as it changes. For example, a user adjusting an evidence slider will see all of the beliefs in the networks move to indicate the change. This animation, in particular, supports current efforts to understand how best to visualize causality (Ware, 2000). We have also developed methods to help draw attention to changes in a network, so that many variables may be simultaneously monitored for change (e.g., through "snapshots" of the beliefs in the network compared to the network after evidence has been posted), although others have developed similar user interface designs to achieve this capability (e.g., Elvira: http://www.ia.uned.es/~elvira)

Next, we addressed the need to develop methods that render nodes differently as a function of the type of variable they represent (see (5) above). The following figures show the prototype node renderers for Boolean (Figure 5-1 and Figure 5-2), Ordinal (Figure 5-3 and Figure 5-4), and Categorical (Figure 5-5 and Figure 5-6) node types:

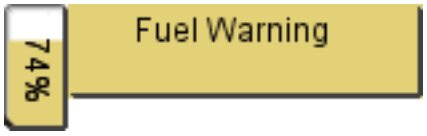

Figure 5-1: A Boolean node

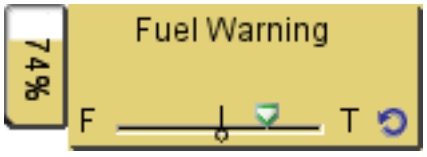

Figure 5-2: A Boolean node expanded to show the evidence/baseline slider

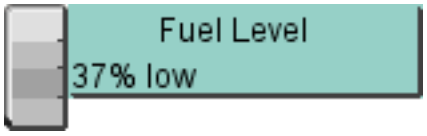

Figure 5-3: An Ordinal node

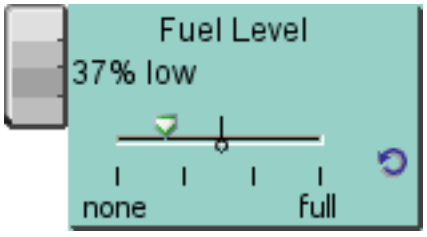

Figure 5-4: An Ordinal node expanded to show the evidence/baseline slider

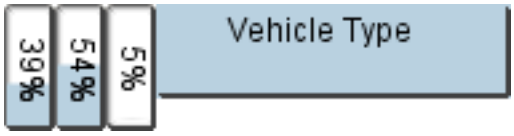

Figure 5-5: A Categorical node

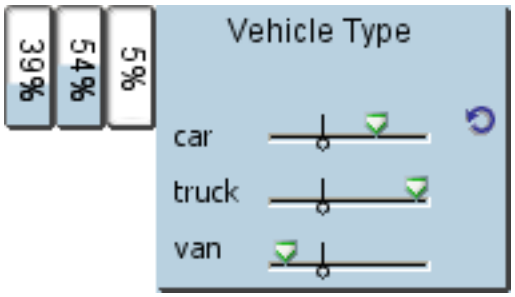

Figure 5-6: A Categorical node expanded to show the evidence/baseline sliders

We attempted to address the confusion about "beliefs" versus "evidence" that we observed with many of our domain experts by visually separating these two elements (belief being represented by the vertical bar, evidence via the horizontal slider) (see (3) above) and only expanding the node to show the evidence slider(s) when the user hovers the mouse over the node. This also provided additional affordances (or lack thereof) to show that evidence is something that could be entered by the user. The expanded nodes in Figure 5-2, Figure 5-4, and Figure 5-6 show an evidence knob above the slider track and also a baseline knob below the track. The baseline knob can be removed to simplify the user interface, if necessary.

The Boolean node simply displays the belief for the true state in a vertical bar and provides a slider that allows the user to (optionally) set the evidence between true and false. The Ordinal node (shown with 4 states to represent the intermediate steps from "none" to "full") necessarily breaks the vertical belief bar into segments to represent the distribution across the states. A straightforward defuzzification of these results has also been developed to represent this belief as a non-segmented bar. The user can (optionally) set the evidence using a single slider that is fuzzified into values for each state. The Categorical node displays the belief for each state in its own vertical bar and provides a separate evidence slider for each state. We are continuing to prototype additional visualization and user interface methods for all of these nodes.

We have also developed methods for expressing probability ranges and baseline probabilities in an attempt to address (4) above (Figure 5-7), although we believe these methods may add a level of complexity that may inhibit usability and have developed versions of the interface that omit these methods. Given that experts may struggle to express in any numerical form their uncertainty about evidence, we developed an interface that allows the user to "grow" a confidence interval around any evidence that is posted. Similarly, we provided a "dual slider" interface where the baseline probability can be expressed. We believe these two additional capabilities (expressing a confidence interval and a baseline) may represent a higher level of complexity in the user interface that may be too sophisticated for some very rapid model development environments, but may be used in the refinement of initial models developed with an interface permitting more limited expressivity.

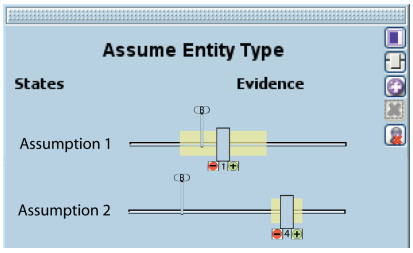

Figure 5-7: Methods for expressing confidence intervals on posted evidence and baselines for individual states

In addition to using our experience with how experts may tend to express their knowledge and reasoning to guide the design of node visualization and user interface methods, we also developed methods for exploiting the CIM's ability to more rapidly quantify causal relations. Because the CIM allows the influence between Boolean and Ordinal nodes to be expressed as a single number, we developed the interface shown in Figure 5-8.

Figure 5-8: Interface for manipulating the strength of a causal relationship between variables

In this case, the user has moused-over the link between the two nodes, and buttons have appeared allowing the user to change the excitatory or inhibitory nature of the link. By adjusting this to various ends of a discrete range (or a continuous range using a slider rather than the two buttons shown) for each parent, the user completely specifies the CPT for the child. In the cases where a child or parent variable is Categorical, additional buttons appear for each state. This interface becomes clumsy for Categorical-to-Categorical variable relationships, as the relationship between each child state and each parent state must be specified. In practice, however, we have found that experts are able to reformulate their reasoning in a form that uses primarily Ordinal or Boolean types. In addition, we have also developed prototype methods for collapsing certain structures of Boolean nodes into Categorical structures (and vice versa).

Although the CIM allows parent influences to be specified in the range [-1, 1], our experience has suggested ±5 steps to be a reasonable level of granularity so we actually elicit an integer in the range [-5, 5] for each parent's influence on the child (but note that establishing the optimal level of granularity with empirical evaluation remains a goal of future work). In Figure 5-6, we show these levels of link strength and that we adjust the link's hue, saturation, and width to indicate its influence (providing multiple, redundant visual cues). Alternatively, we could allow the user to adjust a continuous value between [-1, 1] with a slider and continuously vary the link's visual parameters.

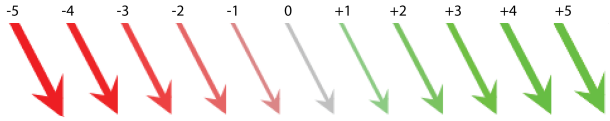

Figure 5-9: Visualization of link strength and inhibitory/excitatory influence

By providing this additional visualization, we allow the user to see, at a glance, not only the relationships among the variables in the network, but also the proportional strength of those relationships. This imparts, to a degree, the sensitivity of the variables to each other, meaning that differences in sensitivities can be rapidly adjusted to match domain expert expectations (e.g., "I realized I made this a much bigger influence than I should have when I looked at all of these other variables"). We have discovered potentially challenging interactions between the names of variables and the implications of the red and green hues used in the visualization. There may be some natural mappings of the color to what the variable is intended to represent ("Happy" is more green than red, and "Being Happy" has an inhibitory influence on "Moping", but the red link may be perceived as causing something to be "less happy"). Similarly, the semantics of the variable names can interact with the visualization of causal links (e.g., "Not Happy" has an excitatory influence on "Moping"). We have, in cases where the variable semantics can be more structured, provided mechanisms for "flipping" the terminology and associated link colors.

An additional challenge faced in a network where experts can easily express their own reasoning is the need for tools to support validation. In the application of these methods across domains, we have discovered that invalid causal chains may be formed simply because it was easy to do so. As a result, we have developed methods that do relatively simple translation of the causal paths in the network into text (e.g., variable A with a +5 link to variable B would result in "A has a strong positive influence on B"). This method appears to be effective for simple models or models with relatively short causal pathways (or for models that follow particular structural patterns).

6. CONCLUSIONS & FUTURE WORK

All of the above interfaces have been implemented and developed as part of a future release of our BNet™ product suite, and as part of many of the custom applications we develop for our clients. As a result, we have used these interfaces and will assert that domain experts without a significant understanding of the underlying computational representation can rapidly externalize and validate their own reasoning more easily than with current off-the-shelf BN modeling packages. We have observed experts, particularly working collaboratively, use our prototype user interface to develop sophisticated models with very little intervention from either knowledge engineers or computational modelers. While this empowers the domain experts to create their own models, it also introduces an opportunity for representational errors that would not otherwise occur with an experienced knowledge engineer "in the loop" along the lines suggested by (Tabachneck-Schijf et al., 2006).

With these encouraging experiences, but without any comprehensive empirical evaluation, we hope to pursue additional efforts to resolve some of the user interface and computational challenges that have arisen from this effort (e.g., semantic issues with node names and the natural mappings of certain colors to certain meanings). We also hope to pursue experimental efforts to more formally test the simplifying assumptions in the CIM, and other observation- and analysis-based design choices. In addition, the user interface designs presented, while capturing the desired functionality, will still require additional refinement to ensure consistency across variable types (and, ideally, additional customizability). Finally, we are also in the process of developing methods for using the simplified interface to aid in the expression of cases that can be used within existing BN learning algorithms.

Acknowledgements

We would like to express our appreciation for the invaluable testing and feedback provided by Karen Harper, Chen Ling, Sam Mahoney, Sean Guarino, and Eric Carlson. In addition, we would extend our deepest gratitude to Greg Zacharias for his continued funding and support of our work with Bayesian networks.

References

Cox, Z. & Pfautz, J. (2007). *Causal Influence Models: A Method for Simplifying Construction of Bayesian Networks.* (Rep. No. R-BN07-01). Cambridge, MA: Charles River Analytics Inc.

Das, B. (2004). Generating Conditional Probabilities for Bayesian Networks: Easing the Knowledge Acquisition Problem. http://www.arxiv.org [On-line].

Available: http://www.citebase.org/cgi-bin/citations?id=oai:arXiv.org:cs/0411034

Diez, F. J. & Druzdzel, M. (2006). *Canonical Probabilistic Models for Knowledge Engineering.* (Rep. No. CISIAD-06-01). Madrid: UNED.

Diez, F. J. & Galan, S. F. (2003). An Efficient Factorization for the Noisy MAX. *International Journal o Intelligent Systems,* 18165-177.

Glymour, C. (2001). *The Mind's Arrows: Bayes Nets and Graphical Causal Models in Psychology*. Cambridge, MA: The MIT Press.

Guarino, S., Pfautz, J., Cox, Z., & Roth, E. (2006). Modeling Human Reasoning About Meta-Information. In *Proceedings of 4th Bayesian Modeling Applications Workshop at the 22nd Annual Conference on Uncertainty in AI: UAI '06*. Cambridge, Massachusetts.

Helsper, E. M., van der Gaag, L. C., Feelders, A. J., Loeffen, W. L. A., Geenen, P. L., & Albers, A. R. W. (2005). Bringing Order into Bayesian-Network Construction. In *Proceedings of K-CAP'05*. Banff, Alberta, Canada.

Helsper, E. M., van der Gaag, L. C., & Groenendaal, F. (2004). Designing a Procedure for the Acquisition of Probability Constraints for Bayesian Networks. In *Engineering Knowledge in the Age of the Semantic Web* (pp. 280-292). Berlin: Springer-Verlag Berlin Heidelberg.

Henrion, M. (1989). Some Practical Issues in Constructing Belief Networks. In L. Kanal, T. Levitt, & J. Lemmer (Eds.), *Uncertainty in Artificial Intelligence 3* (pp. 161-173). North Holland: Elsevier Science Publishers.

Jensen, F. V. (2001). *Bayesian Networks and Decision Graphs*. New York: Springer-Verlag.

Koelle, D., Pfautz, J., Farry, M., Cox, Z., Catto, G., & Campolongo, J. (2006). Applications of Bayesian Belief Networks in Social Network Analysis. In *Proceedings of 4th Bayesian Modeling Applications Workshop at the 22nd Annual Conference on Uncertainty in AI: UAI '06*. Cambridge, Massachusetts.

Kraaijeveld, P., Druzdzel, M., Onisko, A., & Wasyluk, H. (2005). GeNIeRate: An Interactive Generator of Diagnostic Bayesian Network Models. In *Proceedings of Working Notes of the 16th International Workshop on Principles of Diagnosis (DX-05),* (pp. 175-180).

Lefevre, R., Pfautz, J., & Jones, K. (2005). Weather Forecast Uncertainty Management and Display. In *Proceedings of 21st Int'l Conf. on Interactive Information Processing Systems (IIPS) for Meteorology, Oceanography, and Hydrology*. San Diego, CA.

Mathias, K., Isenhour, C., Dekhtyar, A., Goldsmith, J., & Goldstein, B. (2006). When Domains Require Modeling Adaptations. In *Proceedings of 4th Annual Bayesian Modeling Applications Workshop at UAI '06*. Cambridge, MA.

Nadkarni, S. & Shenoy, P. P. (2000). A Causal Mapping Approach to Constructing Bayesian Networks. School of Business Working Paper NO.289.

Neil, M., Fenton, N., & Nielsen, L. (1999). Building Large-Scale Bayesian Networks. submitted Knowledge Engineering Review . Ref Type: Generic

Pang, B., Zhang, D., Li, N., & Wang, K. (2004). Computerized Tongue Diagnosis Based on Bayesian Networks. *IEEE Transactions on Biomedical Engineering,* 51(10), 1803-1810.

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

Pfautz, J., Roth, E., Bisantz, A., Llinas, J., & Fouse, A. (2006a). The Role of Meta-Information in C2 Decision-Support Systems. In *Proceedings of Command and Control Research and Technology Symposium*. San Diego, CA.

Pfautz, J., Roth, E., Powell, G., Fichtl, T., Guarino, S., & Carlson, E. (2006b). Cognitive Complexities Impacting Army Intelligence Analysis. In *Proceedings of Human Factors and Ergonomics Society 50th Annual Meeting*. San Francisco, California.

Rimey, R. & Brown, C. (1994). Control of Selective Perception Using Bayes Nets and Decision Theory. *International Journal of Computer Vision,* 12(2-3), 173-207.

Rosen, J. & Smith, W. (1996a). Influence Net Modeling With Causal Strengths: An Evolutionary Approach. In *Proceedings of Command and Control Research and Technology Symposium*.

Rosen, J. A. & Smith, W. L. (1996b). Influencing Global Situations: A Collaborative Approach. *US Air Force Air Chronicles*.

Rosen, J. A. & Smith, W. L. (1996c). Influence Net Modeling With Causal Strengths: An Evolutionary Approach. In *Proceedings of the 1996 Command and Control Research and Technology Symposium*.

Skaanning, C. (2000). A Knowledge Acquisition Tool for Bayesian-Network Troubleshooters. In *Proceedings of The Sixteenth Conference on Uncertainty in Artificial Intelligence*. Stanford, CA: Stanford University.

Tabachneck-Schijf, H. & Greenen, P. (2006). Preventing Knowledge Transfer Errors: Probabilistic Decision Support Systems Through The Users' Eyes. In *Proceedings of 4th Annual Bayesian Modeling Applications Workshop at Uncertainty in AI '06*. Cambridge, MA.

Ware, C. (2000). *Information Visualization: Perception for Design*. New York: Morgan-Kauffman.

Wiegmann, D. A. (2005). *Developing a Methodology for Eliciting Subjective Probability Estimates During Expert Evaluations of Safety Interventions: Application for Bayesian Belief Networks.* (Rep. No. AHFD-05-13/NASA-05-4). Aviation Human Factors Division Institute of Aviation, University of Illinois.