LA-UR- 08-3952

Approved for public release; distribution is unlimited.

Title:	Applying Bayesian Belief Networks in Rapid Response Situations
Author(s):	William L. Gibson, Deborah A. Leishman, and Edward Van Eeckhout
Intended for:	Presentation at the 42nd Hawai`i International Conference of System Sciences and publication in the conference proceedings

Los Alamos
NATIONAL LABORATORY
EST.1943

Form 836 (7/06)

Applying Bayesian Belief Networks in Rapid Response Situations

Abstract

We have developed an enhanced Bayesian analysis tool called the Integrated Knowledge Engine (IKE) for monitoring and surveillance. Our enhancements are suited for Rapid Response Situations where decisions must be made based on uncertain and incomplete evidence from many diverse and heterogeneous sources. The enhancements extend the probabilistic results of the traditional Bayesian analysis by (1) better quantifying uncertainty arising from model parameter uncertainty and uncertain evidence, (2) optimizing the collection of evidence to reach conclusions more quickly, and (3) allowing the analyst to determine the influence of the remaining evidence that cannot be obtained in the time allowed. These extended features give the analyst and decision maker a better comprehension of the adequacy of the acquired evidence and hence the quality of the hurried decisions. We also describe two example systems where the above features are highlighted.

1. Introduction

For some time now, we have been applying Bayesian Belief Networks (BBNs) to problems involving multisource data fusion. In simple terms, Bayesian Belief Networks process "evidence" to compute probabilities of "hypotheises". For example, some of our problems have involved monitoring of an adversary's actions to determine intent (hostile or benign), or monitoring of a remote facility to determine what types of covert processing might be done there. In these cases the evidence might be extracted from textual intelligence messages acquired from overhead reconnaissance assets or other types of intelligence. In these applications, evidence is costly and risky to obtain and one would want to optimally task the intelligence gathering assets to collect the best evidence to reach conclusions quickly and with reasonable costs. In these problems, the hypothesis nodes in the BBN will likely represent competing alternatives as to what the adversary is really doing. Another type of problem involves near-real-time surveillence for the purpose of threat detection and identification. In such cases, the evidence is extracted from real-time sensor data feeds as well as other types of sources. All of these problem types can require rapid response depending on the severity of the threat identified.

The consequences of a wrong decision are very harsh for these types of rapid response problems. The analyst and decision maker not only want the answer, but want to know the uncertainty in the answer. They want a high quality answer quickly, and if that is not possible, they want to know if the answer is not high quality. To help with this, we have developed some enhancements to traditional Bayesian Analysis which quantify the uncertainty of statements which are themselves probabalistic in nature (such as the results from Bayesian analysis). We have found that subject matter experts have different ways of expressing uncertainty especially when providing evidence, so we have tried to be careful to attach clear meanings to expressions of evidence uncertainty. The enhancements we will describe have to do with treating model parameter uncertainty, treating evidence uncertainty, determining the optimal evidence to collect next (evidence marshalling), determining the best asset with which to collect the evidence (asset allocation), and finally, determining whether one has enough evidence to make a decision (remaining influence).

Procedurally, our approach was to select a well-known traditional Bayesian Analysis tool called Netica (typically used by researchers), as our Bayesian inference engine. Our enhanced Analysis tool, called the Integrated Knowledge Engine (IKE) wraps around Netica to "weaponize" traditional Bayesian analysis to better handle these types of problems and to make it more suited for use by intelligence analysts and decision makers.

This paper is intended to be expository or practical, rather than academic or theoretical, and is a report on a mature work in progress. After a brief introduction to Bayesian Networks which also defines our terminology, we will discuss two example applications of IKE, and then focus on the mechanics of IKE's key capabilities.

2. Bayesian Belief Networks

Bayesian Belief Networks [1,2] provide a way to conceptualize and model problems which involve trying to reach conclusions based on evidence. Often

they are used to try to understand the causal relationships between a set of variables. BBNs have been successfully applied to various problem domains such as medical (diagnosis), judicial (guilt/innocence), and forensics (what happened), to name a few. A BBN consists of nodes and directed links (arrows) connecting the nodes. One can think of the arrow as representing the biblical "begat" relationship. Thus nodes may be thought to have parent-child relationships. Often in Bayesian modeling, the arrow represents a causal relationship. Hence the rule of thumb – "Parents cause the children".

A node in a BBN represents a variable that can be in only one of finitely many states. For example if temperature is a variable, one could say that the temperature could be hot or cold. Thus the temperature node would be modeled as having two states: hot and cold. A more complicated problem might require that temperature have four states; freezing, cold, warm, hot. We may not know what state the temperature node is in, in which case we say that the temperature node is unknown. If we get some data that tells us the state is hot, we can enter that finding into the BBN and "set" the temperature node's state to hot. This is called entering evidence (or entering a finding).

Often the top-level nodes in a BBN represent the competing alternatives that we are trying to sort out: is the factory making fertilizer or anthrax, or sarin? These nodes are called hypothesis nodes. Often the bottom level nodes in the network represent things we can observe as evidence: is the factory using low, medium, or high amounts of electricity? These nodes are called evidence nodes. Given the evidence that we have entered into the BBN, we want the BBN to calculate the probabilities of the states of the hypothesis nodes to reach a conclusion such as: The probability that the factory is making fertilizer is 90%; anthrax, 7%; and sarin, 3%. Several algorithms [3] have been developed that can perform this Bayesian Inferencing in a practical and useful manner – provided the network is a BBN.

In order for a network to be a BBN it must satisfy two conditions: (1) It must be an Acyclic Directed Graph – acyclic in the sense that there are no loops in the graph – i.e. a parent may not be the child of one of it's descendents, and (2) The network must satisfy the Markov condition – if the state of all the parents of a node are known, then the state of that node is influenced only by it's descendents. These conditions simplify the problem enough that it can be solved by Bayesian Inferencing algorithms. Condition (1) keeps the algorithms from encountering an infinite loop, and condition (2) allows the model builder and the algorithms to worry only about the immediate relationships between a node and it's parents. The figure below shows a portion of a BBN whose purpose is to diagnose car trouble (from an example by NORSYS). This snippet shows a child node with it's three parents.

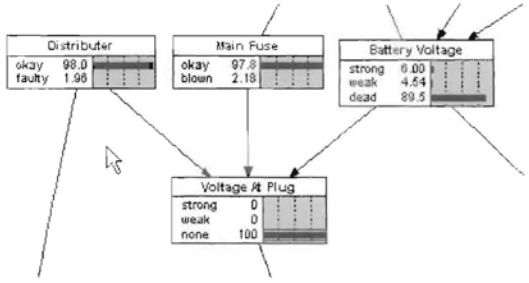

Figure 1. Car Diagnosis BBN Excerpt

To construct a BBN one must provide a table of conditional probabilities (CPs) for each node. These CPs represent our answer to the question: Given the state of all my parents, the probabilities for my states are the following. For example, the CP Table (CPT) for the "Voltage at Plug" or PV node is shown below.

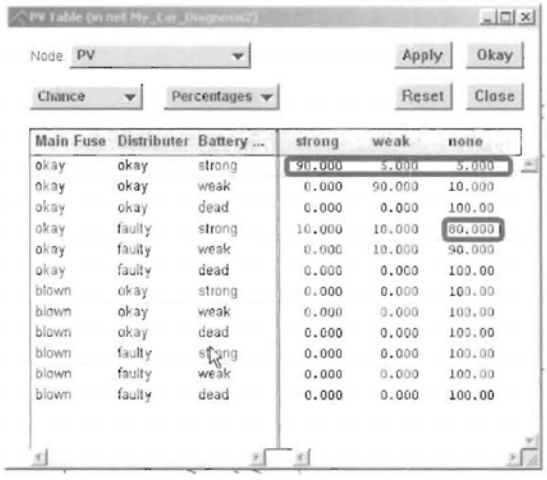

Figure 2. Node conditional probability table

The CP shown in the highlighted cell means: Given that Main Fuse is **okay** and Battery is **okay**, but Distributor is **faulty**, the probability that the Voltage at Plug is **none** is 80%. Because we are reasoning from cause to effect (from parent to child), a subject matter expert (such as an auto mechanic) who is familiar with the system being modeled can usually come up with these conditional probabilities and easily populate the table. We call this eliciting the conditional probabilities. It is much harder to reason from effect to cause, but this is precisely what

the Bayesian inferencing algorithms do for us. We enter the observed effects into the BBN as evidence, and the BBN calculates the probable cause. The CP Table row that is highlighted is representative of any CPT row in that it is a vector of conditional probabilities (expressed as percentages) that add to 100% (the probabilities add to one) and the dimension of the vector is the number of states of the node. There is one CPT row for each possible combination of parent state values. When the node has no parents, the CPT contains only one row. The probabilities in this row are called prior probabilities, they are unconditional probabilities that represent the probability that that node state will occur in the general population.

3. The Integrated Knowledge Engine (IKE)

The Integrated Knowledge Engine enhances the traditional Bayesian Analysis provided by the Netica engine, by providing the following additional major capabilities:

- Analysis (Inferencing with Uncertainty)
- Evidence Marshalling (with Uncertainty)
- Asset Allocation
- Remaining Influence

Other useful capabilities (not discussed here) include:

- Graphical User Interface
- Evidence Message Processing
- Message Database Store/Replay
- Geographic Situation Display
- Evidence Message Simulation

IKE's flexible graphical user interface can be easily changed, so each new application of IKE may have it's own look and feel, while the underlying classes that implement the core capabilities remain unchanged.

In the simplest configuration, IKE is used like a hand calculator with the analyst manually entering whatever evidence is deemed appropriate. In some applications, raw intelligence messages arrive at IKE only to be placed in the mailbox for an evidence node (some process has routed the message to the appropriate node). The analyst reviews the message manually and decides whether to enter evidence. In other near-real-time applications, special IKE evidence messages are automatically generated from the multisource data streams and set their evidence into IKE automatically. We have found that viewing incoming data streams as sources of discrete evidence in a Bayesian Belief Network provides a framework (and a simple architecture) for integrating diverse types of input data for knowledge discovery. A typical architecture for near-real-time applications is shown in the figure below.

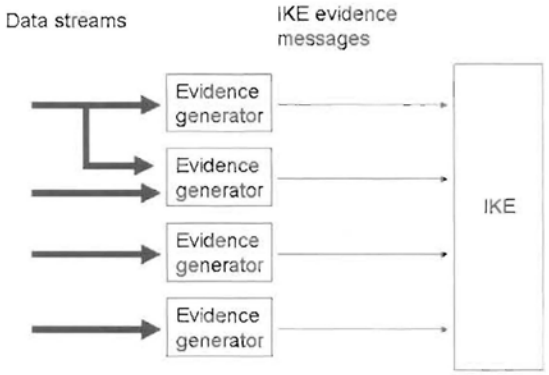

Figure 3. Typical near-real-time architecture

The kind of processing done inside each evidence generator is determined by the type of data source(s) streaming into that generator. It ranges from simple threshold triggering to highly complex textual or image/signal processing. All of the evidence generators know they must generate evidence for one or more evidence nodes in the BBN, so the modeling process that identified the important evidence variables has given definition and purpose to the evidence generation processing.

The Integrated Knowledge Engine is implemented using the Java programming language and interacts with Netica via the NeticaJ Application Programmatic Interface (API) .

4. Example Systems

Traditional BBN models make no distinction between hypothesis nodes and evidence nodes (evidence may be entered at any node). However, when we build a BBN model for a client application, we explicitly identify the hypothesis nodes for the problem and give them labels h1, h2, etc. We identify the evidence nodes as those effects that are actually observable in practice and give them labels e1, e2, etc. In a downward flow layout, the hypothesis nodes tend to be at the top of the diagram and the evidence nodes at the bottom. This helps the analyst and decision maker focus on the nodes that are important to them.

The first example (see figure below) is a version of IKE used in a mailbox/manual mode by the analyst. The right side displays the BBN. The upper left tabulates results for the hypothesis nodes since all may not be visible in the diagram window. The lower left shows the evidence node mail boxes. The analyst reads the messages in the mail boxes and manually sets the evidence into the BBN to perform analysis.

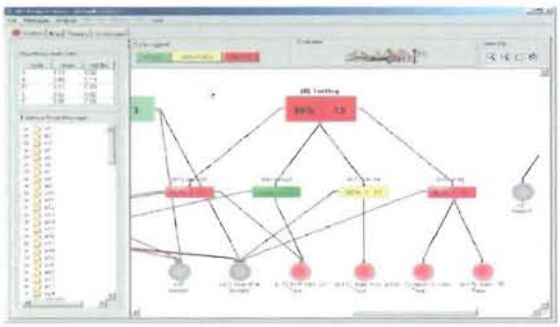

Figure 4. Analysis

The next figure shows the planning tab, used for evidence marshalling and asset allocation. On the simplified diagram, the analyst selects one or more hypothesis nodes and some subset (usually all) of the evidence nodes to marshall over. The lower left displays the prioritized list of best evidence to gather next (the evidence nodes that give the most information about the selected hypothesis nodes), and the lower right shows the best assets to use to collect the evidence.

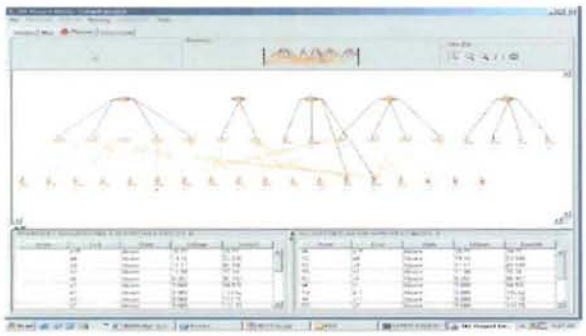

Figure 5. Planning

The second example (see figure below) shows IKE embedded in a near-real-time architecture (like figure 3) called the Remote Perimeter Surveillence (RPS) system used to monitor vehicular and pedestrian traffic on a remote canyon road. The RPS system processes data from a distributed sensor network (DSN) whose nodes have seismic, acoustic, still camera, and radiation sensors and from a video system that does motion detection and object tracking. The evidence generators consume these data streams to produce IKE evidence messages to set evidence into a BBN whose purpose is to determine the probability that the moving object is a vehicle (car, humvee, truck) or a pedestrian, and whether it is suspicious.

Figure 6. Remote Perimeter Surveillance System

The upper left shows the video view from the canyon rim with a bounding box around the detected moving object (a car), middle left is a plot of the raw DSN data, and lower left shows a close-up view from a DSN still camera down in the canyon bottom. The upper right shows the DSN sensor nodes on a geographic situation display, with the nodes reporting shown in red, middle right shows raw/averaged DSN and video data, and bottom right shows the two main IKE hypothesis nodes and some evidence nodes for manual entry.

5. Quantifying Uncertainty

Traditional Bayesian analysis produces results that are probabilities – the probabilities that the node is in each of it's states. For example if the node is Animal (see figure below) the BBN will calculate a probability for each kind of animal.

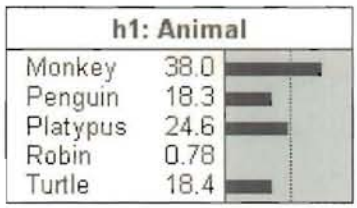

Figure 6. Point Probabilities

The results are not saying the animal is a monkey, but rather that it is most probably a monkey. The results tell you how to bet. These probabilities are sometimes called "point probabilities" or "first-order probabilities", and they are the answers from traditional Bayesian analysis. But how good are these answers?

We know they are affected by how well we elicited the conditional probabilities. The CPs in the CP tables (including the priors) are the parameters of our model, so errors in the model parameters can introduce errors in the answers. If we entered some evidence of which we were not certain, this could also induce an error into the answers.

We wish to put a "one sigma error bar" on the point probabilities, representing one standard deviation, to show how much uncertainty is contributed by the model parameter uncertainty and by the uncertainty in the evidence. To do this we use a Monte Carlo simulation [4].

We need to think of each CPT row as a multi-variate random variable - a vector of scalar random variables in the interval [0, 1] – and here's the catch – whose components must sum to one.

Instead of each CP in the CPT row being just a probability, there is now a mean value for the CP and a sigma value for the CP. Thus the CP is a random variable characterized by it's mean and sigma. The size of sigma expresses how much uncertainty there is in the CP. In this way we have augmented the traditional CP table by including the sigmas along with the means (the original CP values). Thus each CPT row becomes a vector of means and a corresponding vector of sigmas. For each CPT row, we must create a multi-variate random distribution having the same means and sigmas, such that each random vector drawn from the distribution sums to one. We found that a mixture of two Dirichlet Distributions is able to do this [5]. To initialize the Monte Carlo simulation, the BBN is traversed, and for each augmented CPT row, a matching Dirichlet Mixture (DM) is created and assigned to the row.

Then on each cycle of the simulation:

- Each CP Table Row draws its CP values from its associated DM (including Priors)
- Retract all evidence from the BBN
- Enter current evidence for this cycle (can include uncertain evidence)
- Ask NETICA to inference and obtain all the state probabilities
- Accumulate state probabilities for calculating means and sigmas
- Accumulate state probability minimums and maximums

When the desired number of cycles have been completed:

- Calculate state probability means and sigmas from accumulated data
- Fetch the state probability minimums and maximums

The results (now with our desired error bar) are displayed as in the figure below. The analyst can easily ascertain at a glance how good the answer is (how much uncertainty is caused by model parameter uncertainty and evidence uncertainty (discussed below).

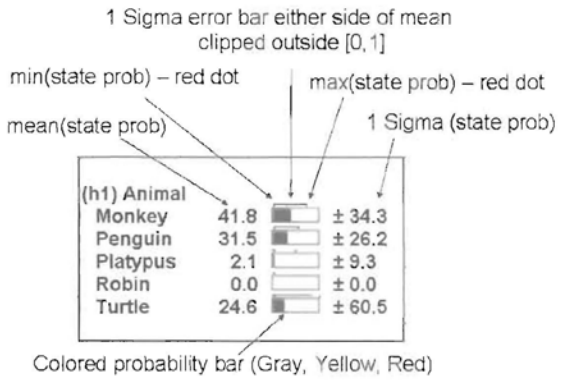

Figure 7. IKE Results for a node

Currently IKE deals with evidence uncertainty by dithering the evidence during the Monte Carlo Simulation. Rather than entering evidence in the traditional way by picking one of the node's states as the finding for that node, and holding that setting constant during the Monte Carlo, we enter a "certainty" value for each state of the node. These state certainties are percentages and must sum to 100. For example, for a temperature node, we might enter certainties of: 80 for hot, 15 for warm, and 5 for cold. This could be interpreted as saying... I'm 80% certain that the temperature is hot. I might be wrong, but even if I am it is still more likely to be warm than cold...say 15% for warm, and only 5% for cold.

In the Monte Carlo simulation of say, 1000 cycles, we interpret this to mean that for 800 of the cycles temperature should be set to hot; for 150 cycles, temperature = warm, and for 50 cycles temperature = cold. When dithering more than one evidence node, care must be taken not to induce correlations between the dithered nodes. This can be done by drawing a selector variable from a uniform distribution.

Another way to deal with uncertain evidence at modeling time is to place a "noisy observer" node below the real evidence node as its only child. The observer node represents a noisy sensor or an imperfect observer and its false positive and false negative rates can be entered into the observer's CPT. One then enters the evidence into the observer node rather than the real evidence node. This could be done at run time by creating the observer node on the fly and inserting it into the BBN at run time. An area of research for us is the use of Bayesian "likelihoods" to express evidence uncertainty.

5. Optimizing Evidence Collection

To perform evidence marshalling in IKE, the analyst selects one or more hypothesis nodes and then runs evidence marshalling. The system will then rank order all the evidence nodes in terms of how much information gain each evidence node contributes to the group of selected hypothesis nodes. Thus the system identifies the most influential evidence to collect next. In so doing, the system acknowledges the evidence currently set in the analysis, and calculates the information gains over the remaining unknown evidence nodes.

Like the IKE Analysis function, evidence marshalling uses Monte-Carlo to compute the variance of the results...in this case the variance of the information gains. This variance is used in a utility function to penalize an evidence node if the variance of its information gain is relatively larger than that of another evidence node having nearly equal information gain, causing the node to rank lower.

Information gain is computed from entropy, a concept from thermodynamics. As a system becomes more disorganized its entropy increases and its information gain decreases. When applied to information theory, entropy and information gain are inverse measures of the influence one node has on another. We use a Netica method to calculate the information gain IG(h:e) between a hypothesis node and each of the evidence nodes. When the user selects multiple hypothesis nodes h1,h2,h3 to marshall over, we would have liked to compute the joint information gain IG(h1,h2,h3 : e). Since Netica has no method for joint information gain, we approximate the desired result as avg[IG(h1:e), IG(h2:e), IG(h3:e)]. An area of research for us is to compute the joint information gain, and also to enhance this sensitivity analysis to find the evidence that best reduces uncertainty and that best discriminates between competing hypotheses.

If the analyst has the ability to task assets to go collect evidence, IKE has a capability to perform an optimal asset allocation based on the optimal list of evidence nodes produced by evidence marshalling. Beforehand, one must make a mapping that identifies which assets can collect (evidence for) which evidence nodes. Then one adds asset characteristics like availability (is the asset available), cost (how costly is the asset to use for collecting this evidence), timeliness (how quickly can this asset collect this evidence), and risk (how risky to collect this evidence). One then adds collection constraints (find me an immediately available asset that can collect the evidence within 30 minutes at the lowest cost and risk) and turns this into an integer linear programming optimization problem (a la operations analysis) which can be solved by a linear programming solver. IKE uses a freeware solver called LP Solve. The result is a list of the best assets with which to collect the optimal list of evidence. When the analyst then goes to the collection community with hat in hand to beg for an asset, hopefully the fact that IKE says this is the most effective evidence and the most effective asset will hold some sway.

6. Determining Remaining Influence

Knowing when you have enough evidence to make a decision can be difficult. After obtaining some evidence, the IKE analysis might be pointing to a particular answer, and the uncertainty in that answer due to model parameter error and evidence uncertainty might even be small (see figure below), but might there not be another piece of remaining evidence that would overturn that answer if that evidence could be collected?

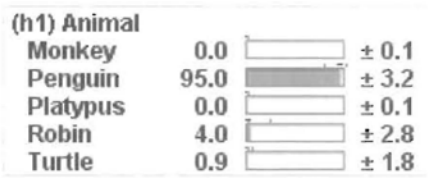

Figure 8. Enough evidence for decision?

One way to get a handle on this might be to run evidence marshalling and look at the remaining evidence nodes that rise to the top of the list. If their information gains are small, one might feel confident that remaining evidence will not greatly affect the current answer. However, the information gain scale has not yet been calibrated (future research) and, marshalling uses it only in a relative way, so looking

at the information gain values is not really helpful at this time.

IKE can produce a definitive answer to this quandary by performing an analysis of "remaining influence". Given the evidence already entered in the system, IKE generates all possible combinations of the remaining evidence findings. Then, on each cycle of the remaining influence simulation, it sets one possible remaining evidence combination into the BBN and accumulates results just as in the Monte Carlo simulation discussed previously. The simulation results are displayed in the same way, but now the interpretation is different. The data displayed has nothing to do with model parameter uncertainty or evidence uncertainty, but rather comes from the dithering of the remaining evidence through all possible combinations. For each node state the mean now represents the average state probability over all the remaining evidence – which may not be very informative. But the standard deviation is informative – it tells us how much variation in the answer is possible due to the influence of the remaining evidence. The max and min markers (the red dots of figure 7) are also informative – especially the max. The max marker records the maximum probability that occurred during the simulation. This implies that there was a remaining evidence combination that caused that maximum value. If a state having this max value would overturn the answer currently in the system, then this is the combination of remaining evidence that would do it. The figures below show two results from the remaining influence simulations. The first result shows that the current answer will not be overturned – hence enough evidence has been obtained for a decision. The second result shows that the current evidence is not sufficient for a decision, but if the decision must be made anyway, at least there is some feeling for the alternatives.

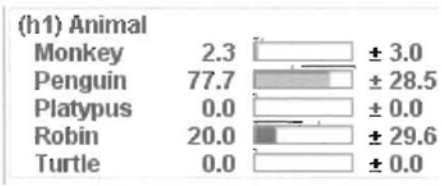

Figure 9. Current answer (Penguin) is firm

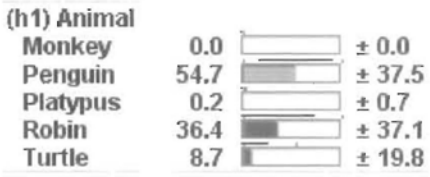

Figure 10. Robin could overturn Penguin

7. Conclusions

The Integrated Knowledge Engine extends traditional Bayesian analysis by giving the modeler a way to express model parameter uncertainty and by giving the analyst a way to express evidence uncertainty. A Monte Carlo simulation wrapped around a traditional Bayesian analysis tool then allows these effects to be included in the enhanced analysis. IKE also provides enhancements to optimize the collection of evidence and to understand when enough evidence has been obtained for a solid decision, and if not, a better understanding of the alternatives. As mentioned in the introduction, all these enhancements are well suited for rapid response situations.

Although IKE has been a mature capability for several years now, there are still several areas of research, as this is a work in progress. We would like to move towards more analytic methods of incorporating the uncertainty directly into the Bayesian analysis and move away from simulation. However, due to the speed of Netica, we find that simulation performs quite well on fairly large models of several hundred nodes. We are looking for a multi-variate random distribution that is a bit more robust than our Dirichlet Mixture. Although uncertainty is considered in evidence marshalling, we would like marshalling to actually drive down uncertainty and drive discrimination between competing hypotheses. Remaining influence analysis is a relatively new capability that can certainly be improved.

Currently, IKE is a monolithic application. We are currently developing a web version of IKE so that teams of analysts can collaborate over the web, each entering evidence in a joint analysis.

IKE is a National Lab capability. If it can help your organization solve a problem, please contact one of the authors.

10. References

[1] Pearl, Judea (2000). *Causality: Models, Reasoning, and Inference*. Cambridge University Press. ISBN 0-521-77362-8.

[2] Finn V. Jensen, "Introduction to Bayesian Networks", Springer; 1 edition (August 15, 1997).

[3] C. Huang and A. Darwiche, "Inference in belief networks: A procedural guide", International Journal of Approximate Reasoning, volume 15, number 3, pp. 225-263 (1996)

[4] N. Metropolis and S. Ulam, "The Monte Carlo Method", Journal of the American Statistical Association, volume 44, number 247, pp. 335–341 (1949) (doi:10.2307/2280232)

[5] D. Izraelevitz, H. F. Martz, D. A. Leishman, and W. L. Gibson, "On representing second-order uncertainty in multi-state systems via moments of mixtures of Dirichlet distributions", Journal of Statistical Computation and Simulation (2008) (to appear).