Int Conf Information Fusion 2010

Information Acquisition Strategies for Bayesian Network-based Decision Support

Ronnie Johansson[1,2]
ronnie.johansson@{his | foi}.se

[1]Informatics Research Centre
University of Skövde

Christian Mårtenson[2]
cmart@foi.se

[2]Division of Information Systems
Swedish Defence Research Agency (FOI)

Abstract – *Determining how to utilize information acquisition resources optimally is a difficult task in the intelligence domain. Nevertheless, an intelligence analyst can expect little or no support for this from software tools today. In this paper, we describe a proof of concept implementation of a resource allocation mechanism for an intelligence analysis support system. The system uses a Bayesian network to structure intelligence requests, and the goal is to minimize the uncertainty of a variable of interest. A number of allocation strategies are discussed and evaluated through simulations.*

Keywords: Bayesian networks, decision support tool, information acquisition

1 Introduction

Determining how to utilize information acquisition resources optimally is a difficult task in the intelligence domain. Nevertheless, an intelligence analyst can expect little or no support for this from software tools today. At the Swedish Defence Research Agency, we are developing an intelligence analysis support tool, `Impactorium` [5], which helps an intelligence analyst to model, structure, fuse and visualize information. In the tool, an incoming *knowledge request* (KR) can be turned into a detailed problem decomposition. Each sub-problem can then be treated separately, with its own planning, collection and analysis steps. The solution to the different sub-problems are then combined using some mathematical framework, in our case using a Bayesian Network (BN). However, even though the sub-problems can be analyzed separately, the impact on the KR of solving a sub-problem may be sub-optimal and the information acquisition resources have to be shared among the sub-problems. This means that the resource allocation must look at what is best for the KR problem as a whole.

The process of acquiring information for state estimation with a BN is akin to *sensor management* in the sense that both concern deliberation of sensing actions and their execution in order to improve the information fusion process and, ultimately, decision support. There is a growing literature on sensor management (surveyed in, e.g., [6] and [4]), but very few efforts seem so far to have been devoted to the estimation of "high-level" information, as for example the decomposable KR of an analyst. One exception is [7], which poses the problem of information acquisition for state estimation with a BN. The main difference between their work and ours is that we allow uncertain evidence of the problem components.

In general, information acquisition is a highly complex problem involving heterogeneous and fallible resources, dynamic and high-level mission objectives, and decentralized control [2, Ch. 3]. In this article, we focus on the issue of information acquisition for high-level mission objectives (such as support for KRs).

2 `Impactorium`: a Decision Support Tool

In order to understand the problem that is targeted in this paper, we need to put it into context. To do this we will picture a scenario of a typical intelligence workflow and describe a tool that is being developed at FOI to support parts of this workflow.

The sole purpose of the pursuit of intelligence is to support a commander or decision maker with information in a decision situation. The intelligence process begins when the decision maker issues a KR to the intelligence organization. The KR is received by an intelligence manager who prioritizes the request and assigns it to an analyst. The analyst is also assigned a set of information acquisition resources.

The request is often formulated as a simple yes/no question (or true/false statement). However, most often there is no information available within the intelligence organization that answers the KR directly. Moreover, the request is likely to be complex in the sense that no single observation will be able to answer it completely and with enough certainty. (Note that when we say observation, we mean all kinds of statements, no matter if it was produced by a sensor or by an analyst reading a document.) This makes it necessary to decompose the request into multiple sub-queries with narrower and more manageable scopes. The sub-queries are

further decomposed until they reach a level where it is possible to find a direct answer to them, either by consulting a knowledge base or by directing sensing capabilities (sensors or human observers) to observe some part of the area of interest. This kind of observable sub-queries are called *indicators*.

After a knowledge request has been decomposed into indicators, information is collected to determine if the indicator value, i.e., the answer to the sub-query, is true or false. The acquired information is analyzed to see if any observations relevant to the indicator can be found. An observation that either supports or rejects an indicator is called *evidence*, and will influence the analyst's estimation of the indicator value. When the indicator values have been determined, they are propagated upwards in the decomposition tree and combined to answer the queries of the higher level. This propagation and combination is repeated until the initial knowledge request is answered.

The combination of evidence can be made either manually, based on the experience of the analyst, or automatically utilizing a supporting mathematical framework. The simplest propagation model would be to just use a mean value calculation to merge the answers from all sub-queries. To make the model a little more elaborate, a weighted mean can be used to capture the varying importance of the different sub-queries. A yet more advanced approach would be to use a BN. This allows a more detailed description of the dependencies as well as the causal relations between the sub-queries.

As mentioned above, when the decomposition is complete and the propagation models have been decided, the analyst starts to search for information matching the indicators to answer the lowest level sub-queries. To do this she exploits the information acquisition resources that she was assigned by the intelligence manager. The resources can be heterogeneous, which means that different resources can be more or less suitable for acquiring information on a particular indicator. To handle this, as suggested in this paper, the analyst could formulate a specific acquisition task for each indicator-resource pair. Then she could estimate the expected value of performing each task. This would form the basis for a task assignment, which could be optimized to minimize the uncertainty of the initial knowledge request.

`Impactorium` is a software support tool for intelligence analysis under development at the Swedish Defence Research Agency (FOI). The tool helps an analyst with modeling, structuring, fusing and analyzing intelligence information, according to the workflow described above. The tool allows an analyst to construct intelligence models, by for instance decomposing a knowledge request into sub-queries and indicators, and then use the model as a basis for a systematic search for and combination of evidence. The models are graphical, and hence built from nodes and links, where the links describe how the nodes influence each other. Each node consists of a variable and a probability distribution. The variable can for example represent a claim that can take values True or False, and the probability distribution

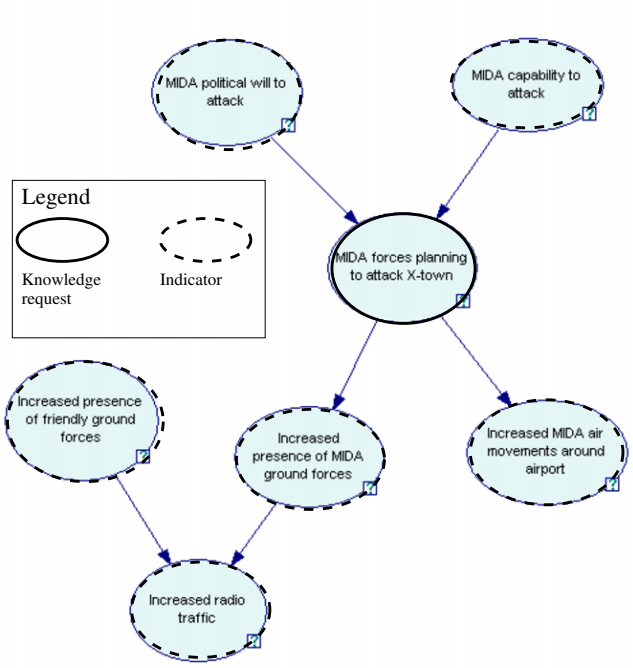

Figure 1:
An example of an `Impactorium` intelligence model represented as a Bayesian network. The network was drawn with the GeNIe tool.

describes how likely it is that the claim is True or False. The probability that a particular node is True depends on the values of its related nodes, and to describe this influence a number of different mathematical models can be used, e.g., a mean value function or a BN.

Each node in the model is tied to a set of evidence that forms the basis for the node's probability distribution. The evidence set ensures traceability since it is possible at anytime to obtain all the information that served as a basis for the assigned value.

`Impactorium` currently does not provide any advice for what information to acquire. That issue is addressed in this paper.

2.1 Example

The following example refers to a training scenario developed by the Swedish Armed Forces. It takes place in Bogaland, a fictive nation with economic, ethnic and religious conflicts. In order to stabilize the situation, the UN has deployed peace enforcing troops under the name BFOR. In this particular example, the BFOR commander in charge of stabilizing the capital X-town has the following knowledge request: Are (the hostile) Mida forces planning to attack X-town?

The intelligence analyst assigned to investigate the case makes a query decomposition using the `Impactorium` tool, and decides to use a BN to model the dependencies. The nodes and their interdependencies can be seen in Fig. 1.

Indicator	Resource	Certainty
Mida pol. will to attack	Int assist	0.80
Mida pol. will to attack	Liaison officer	0.90
Mida pol. will to attack	MOT	0.55
Mida cap. to attack	Int assist	0.70
Mida cap. to attack	Liaison officer	0.90

Table 1: Examples of sensing resources and their expected performance

The next step is to look for evidence to the indicators. In the example, all nodes except the original query (Mida forces planning to attack X-town) and the Increased presence of Mida ground forces-node are indicators. Along with the knowledge request, the analyst is assigned a set of resources for intelligence gathering, e.g.,

- Mechanized platoon 1 (Mech Plt 1)
- Mechanized platoon 2 (Mech Plt 2)
- Intelligence platoon (Int Plt)
- Military Observation Team (MOT)
- Liaison officer
- Intelligence Assistant (Int Assist)

The resources offer different gathering services which can be more or less suitable for capturing evidence to a particular indicator. In order to make a reasonable collection plan, the analyst lists all indicator-resource pairs and for each pair makes a tentative judgment of the expected impact of assigning that particular resource for gathering intelligence to that particular indicator.

An ordinary BN requires that nodes are observed with certainty, i.e., observed nodes are assigned an absolute value. However, in our case, sensor imperfection requires uncertain evidence. In the case of a Boolean node, this imperfection will correspond to a certainty factor between 0.5 and 1, where 0.5 is an observation which won't add any information (equal to a non-observation) and where 1 is a completely certain observation. Table 1 lists an extract of the indicator-resource judgments. In a real situation the cost of each assignment would also be a factor to consider. However, for the sake of simplicity we leave it out (see [3] for a description on different cost aspects of information supply for information fusion services).

3 General Problem Description

In the previous section, we described the `Impactorium` intelligence model consisting of knowledge requests, indicators and their interdependencies and that this can be represented by a Bayesian network. In general, a BN represents a joint probability function (JPF) over all included variables, typically representing *belief* of joint states based on observations and prior probabilities. The probability function of

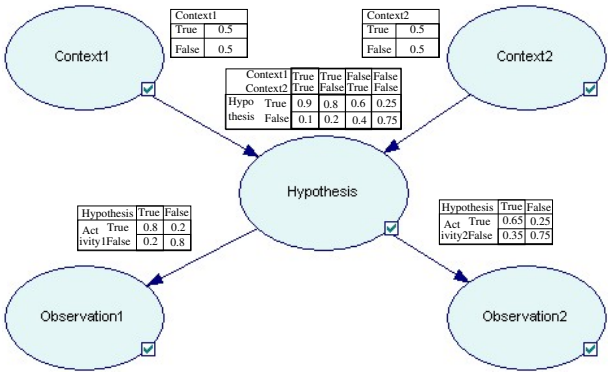

Figure 2:
A generic description of an `Impactorium` intelligence model as a Bayesian network including a hypothesis variable (corresponding to a knowledge request and not directly observable) and information variables (corresponding to indicators). The network was drawn with the GeNIe tool.

a specific variable in the network (representing the belief in the states of the variable) can be found by marginalization. Rather than specifying a single (and potentially enormous) probability table for the JPF, the BN facilitates the specification of a number of smaller conditional probability tables (CPTs), representing the direct dependence between a subset of nodes.

The network consists foremost of *hypothesis* and *information* variables (corresponding to knowledge requests and indicators in `Impactorium`, respectively). The former are variables of primary interest for a decision maker and the latter are variables that can be observed by some means (e.g., by sensors). Fig. 2 depicts a generalized but simple intelligence model as a BN. Indicators are here context variables (i.e., circumstances that may support or reject a hypothesis, such as weather conditions or political climate) and activity variables (i.e., possible consequences of an event), both of which can be observed.

While the hypothesis variable can not be directly observed, the indicators can. Observations of the indicators will affect (through the dependencies of the BN) the belief of the hypothesis variable.

By necessity, we relax the assumption that the indicators can be observed with certainty. Instead, as explained in Section 2, sets of indicator evidence are maintained for each information node in the BN. For simplicity, the evidence are binary statements (either supporting or rejecting the concerned indicator). The sets of evidence are interpreted as uncertain evidence for the node in question. As BNs by design require certain observations, the transformation from sets of evidence to uncertain evidence for the BN must be handled. How we deal with this issue is explained in Section 4.

In Section 2.1, we give examples of resources that can be used to acquire information for an indicator for a specific mission. For our experiments (Section 6), we will use a generalized form of resource. The resources in our ex-

periments are assumed to be homogeneous, i.e., having the same performance. Each resource generates one observation (in terms of a likelihood) of the indicator, with known certainty k, i.e.,

$$p(O_m = i_{m,n} | I_m = i_{m,n}) = k \quad \forall n, \tag{1}$$

where $0.5 < k \leq 1$, I_m is indicator m, O_m is an observation of indicator I_m, $i_{m,n}$ is state n of the binary indicator I_m (in our experiments there are only two states: True and False). Hence, if an observation claims that the state of indicator I_m is $i_{m,n}$ then this information is only certain to degree k. If multiple resources are at the system's disposal, more than one resource may be used for one indicator (hence resulting in multiple observations on the same indicator). We define a *sensing action* to be a full assignment of available resources to observe indicators.

As mentioned, applying a number of resources to observe the indicators will change the understanding of the Hypothesis variable, but it can in general not be guaranteed that the true state of the Hypothesis variable can be determined with certainty. For instance, in the case of the BN in Fig. 2, even if the information variables are known with certainty to be, say, True, the probability of the true state for the Hypothesis variable is less than 1 (0.989 in this case). Hence, to estimate the result of a sensing action it is not appropriate to compare the resulting marginal probability over the Hypothesis variable, $p_O(H|\mathbf{O})$ (where $\mathbf{O}$ is the set of all observations), to its true state (which corresponds to a probability function with probability 1 assigned to the true state) as that is, by the design of the BN, unattainable. Instead, the performance metric we use in our experiments, $Q\left(p_O(H|\mathbf{O}), p_I(H|\mathbf{I})\right)$, is a distance metric between the estimated probability $p_O(H|\mathbf{O})$ and the best possible one $p_I(H|\mathbf{I})$ (where the true state of all information variables $\mathbf{I}$ are known)

$$Q\left(p_O(H|\mathbf{O}), p_I(H|\mathbf{I})\right) = \sum_{n=1}^{N} \left(p_{O,n} - p_{I,n}\right)^2, \tag{2}$$

where N is the number of states of the Hypothesis variable H (in our case just two, True and False) and $p_{0,n}$ and $p_{I,n}$ are the probability values of the nth state for the probability functions. A low value of $Q(\cdot)$ is desirable.

4 Uncertain Evidence

As mentioned in the previous section, observations of indicator states are collected into sets of evidence. These sets are further transformed into uncertain evidence for the indicator variables in the BN. Each evidence can be described as a likelihood (Eq. (1)) and the members of a set of evidence $\mathbf{O}$ can be combined using Bayesian inference in the following way:

$$p(I|\mathbf{O}) \propto p(I) \prod_{o \in \mathbf{O}} p(O = o|I), \tag{3}$$

where $p(I)$ is a common prior (in our experiments assumed to be uniform). For instance, if $\mathbf{O} = \{T, T, F\}$, i.e., contains two observations supporting state True of an information variable and one observation supporting False, and $k = 0.9$, then the fused uncertain evidence would become (the uniform prior can be ignored)

$$\begin{cases} p(I = True|\mathbf{O}) = \dfrac{0.9 \cdot 0.9 \cdot 0.1}{0.9 \cdot 0.9 \cdot 0.1 + 0.1 \cdot 0.1 \cdot 0.9} = 0.9 \\ p(I = False|\mathbf{O}) = \dfrac{0.1 \cdot 0.1 \cdot 0.9}{0.9 \cdot 0.9 \cdot 0.1 + 0.1 \cdot 0.1 \cdot 0.9} = 0.1 \end{cases}$$

Once the uncertain evidence $p(I|\mathbf{O})$ for indicator I has been calculated, it has to be enforced in the BN of the intelligence model.

We use Pearl's method [1] to achieve this. It is suitable for Bayesian networks (simply adding an additional temporary node Z with an arc extending from the node that the evidence concerns to the temporary node), but requires that evidence is specified as likelihoods in the CPT of Z. If, on the other hand, the evidence is expressed as a probability function (as in Eq. (3)), the required likelihoods can still easily be obtained from the desired probability function. Remember that we wish to set the probability function of indicator node I to $p(I|\mathbf{O})$. Now, add a temporary variable Z with states $\{z_i\}_{i=1}^2$. Z by design only has two states, namely $z_1 =$ True and $z_2 =$ False. The likelihoods of Z's CPT are then:

$$\begin{array}{l} p(Z = True|I = i_n) = c \cdot \lambda_n \quad \forall n \\ p(Z = False|I = i_n) = 1 - c \cdot \lambda_n \quad \forall n, \end{array} \tag{4}$$

where λ_n is the ratio of the desired updated probability $p(I = i_n|\mathbf{O})$ and the original probability $p(I = i_n)$:

$$\lambda_n = \frac{p(I = i_n|\mathbf{O})}{p(I = i_n)}. \tag{5}$$

The c in Eq. (6) is a constant that normalizes the λ_ns to make them valid probabilities, e.g,

$$c^{-1} = \max_{n=1}^{N} \lambda_n. \tag{6}$$

5 Information Acquisition

We compare a few different strategies to select sensing actions for information acquisition:

Rand (random): Assigns the available resources randomly to the indicators

Egal (egalitarian): Assigns the resources equally to all indicators (only used when the number of resources equals the number of indicators)

Ex-Greedy (exclusive greedy): Assigns all resources to the indicator with the highest uncertainty (in terms of highest entropy)

Nx-Greedy (non-exclusive greedy): Assigns most resources to the node with the highest uncertainty (in terms of highest entropy), but divides also some of the resources to the other nodes according to the

entropy. Our somewhat ad-hoc approach to distribute the resources is based on the principle that twice as high entropy receives twice as much resources.

Hypent (hypothesis entropy): Assigns the resources in such a way that the expected decrease in entropy of the Hypothesis variable is maximized

While the two first are self-explanatory, Ex-Greedy, Nx-Greedy and Hypent require some further explanation. All three strategies use the concept of *entropy* $\mathcal{H}(I)$ which can be used to quantify the uncertainty of a variable I.

$$\mathcal{H}(I) = -\sum_{n=1}^{N} p(i_n) \log p(i_n) \tag{7}$$

5.1 Ex-Greedy and Nx-Greedy Strategies

The Ex-Greedy strategy executes the following steps:

1: $highest\_entropy \leftarrow 0$
2: **for all** information variables I **do**
3: $E \leftarrow \mathcal{H}(I)$ {from Eq. (7)}
4: **if** $E > highest\_entropy$ **then**
5: $highest\_entropy \leftarrow E$
6: $highest\_indicator \leftarrow I$
7: **end if**
8: **end for**
9: **return** the sensing action that assigns all resources to indicator $highest\_indicator$

A modification is made to the algorithm above so that it can handle the case where the entropy of multiple nodes is maximal. In that case, the node which receives all resources is selected randomly from the set of nodes with maximal entropy.

Nx-Greedy works similarly but stores the entropy of all indicators (not just the one with the highest entropy) and distributes the resources proportionally to the indicators.

5.2 Hypothesis Entropy Strategy

The Hypothesis entropy strategy focuses on the knowledge request, i.e., the hypothesis variable of the BN. The objective is to select the sensing action that minimizes the expected entropy. The strategy executes the following steps:

1: $lowest\_entropy \leftarrow MAXINT$
2: **for all** possible actions a **do**
3: $virtual\_reports \leftarrow sample\_reports(a)$
4: $action\_entropy \leftarrow exp\_imp(virtual\_reports)$
5: **if** $action\_entropy < lowest\_entropy$ **then**
6: $lowest\_entropy \leftarrow action\_entropy$
7: $best\_action \leftarrow a$
8: **end if**
9: **end for**
10: **return** sensing action $best\_action$

Function $sample\_reports(a)$ in line 3 generates the most likely reports given the current understanding of the indicators (as represented by the BN) and action a, e.g., if one sensing resource is assigned to indicator I_m and $p(I_m = True) > p(I_m = False)$ then a virtual observation for $I_m = True$ will be generated. Function $exp\_imp(virtual\_reports)$ in line 4 calculates the entropy of the Hypothesis variable after incorporating the virtual observations for action a. In the case that several actions receive the highest expected entropy decrease, one of the best actions is selected randomly.

6 Experiments

In our experiments, we focus on the prototype intelligence model BN in Fig. 2, which includes a hypothesis variable (corresponding to a generic knowledge request) and some indicators of context and activity. We compare the action selection strategies of Section 5 with respect to the performance metric Q from Section 2. In Fig. 3, we compare the strategies for different observation certainties in the interval 55%-95% (see Eq. (1)) for every 5% when sensing actions involve only one resource. Each strategy is run 1000 times for each degree of observation certainty. For each simulation run, the state of the hypothesis variable is set to be True, and the true states of the indicator variables are selected randomly based on how likely they are given the state of the hypothesis variable. For each simulation run, each strategy is allowed to select its preferred sensing action, observations are randomly generated (conditioned on the true state of the simulation run) and the BN, updated with the new observations, is evaluated according to Q.

The results of the Rand strategy is shown as a red and dot-dashed line. The Ex-Greedy strategy has a dark green and dashed line, while the Nx-Greedy strategy has a purple and shorter-dashed line. Only shown for four resources (Fig. 6) is Egalitarian as a cyan-colored and dotted line.

Both the mean value of 1000 runs as well as 95%-confidence intervals are plotted. As can be seen, the different strategies have similar performance from 55% to 85%. From 85% the Hypent strategy dominates the others. As the Hypent optimizes its performance with respect to the Hypothesis variable (which is also the object of the evaluation with the performance metric Q), this could be expected, but the actual degree of observation certainty where this dominance starts is harder to guess.

For two and three resources (Figs. 4-5), with otherwise identical experiment settings, the advantage of Hypent increases even further, although the strategies are still inseparable for lower degrees of observation certainties.

In Fig. 6, we show the result of using four resources. The results are similar to those in the previous plots, but this one also includes the Egal strategy which distributes the resources equally to all indicators (i.e., they receive one resource each). Contrary to the other strategies it is poor compared to Hypent for lower observation certainties, but better for higher certainties. The reason for its exceptional performance for higher degrees of observation certainty is simply that it will be able to estimate all indicators with high certainty.

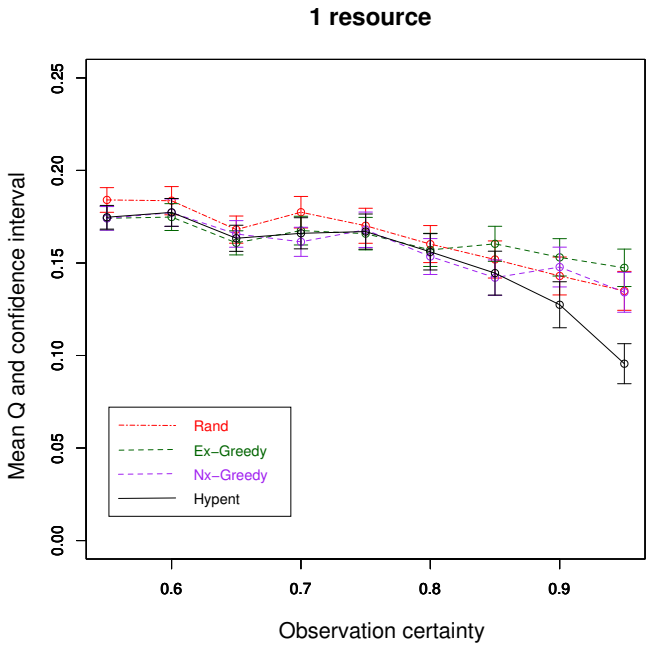

Figure 3:
The result of comparing the strategies using one resource

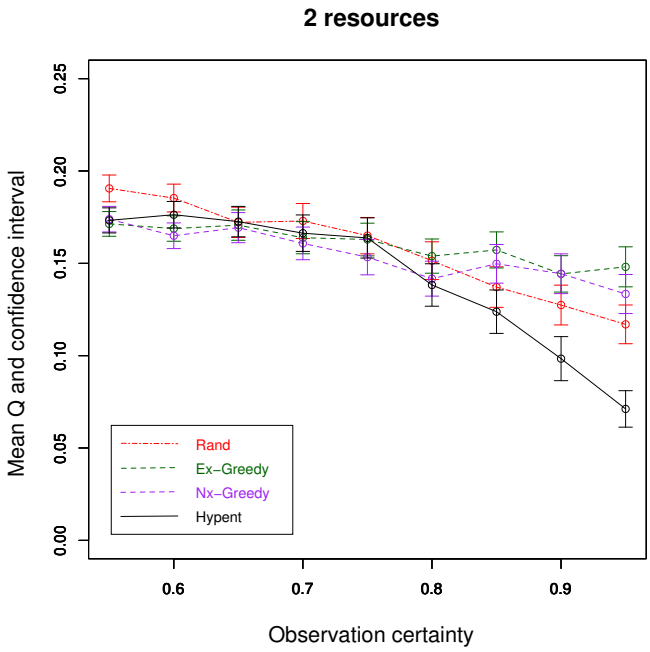

Figure 4:
The result of comparing the strategies using two resources

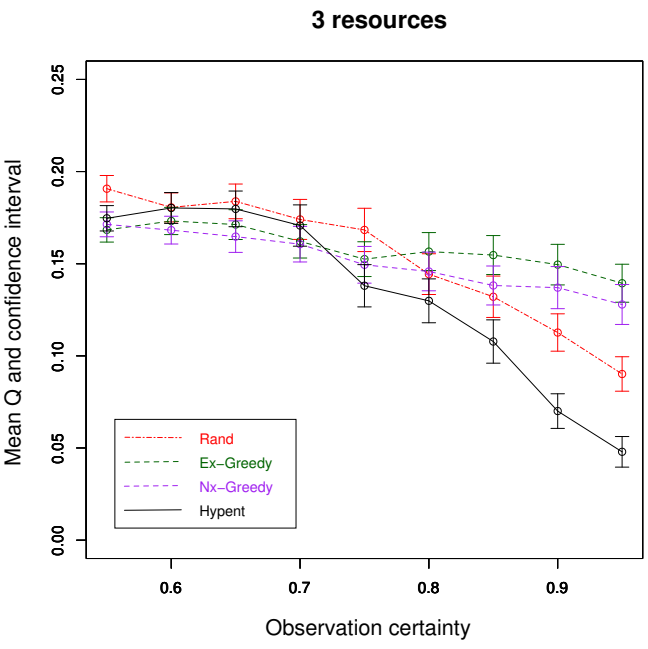

Figure 5:
The result of comparing the strategies using three resources

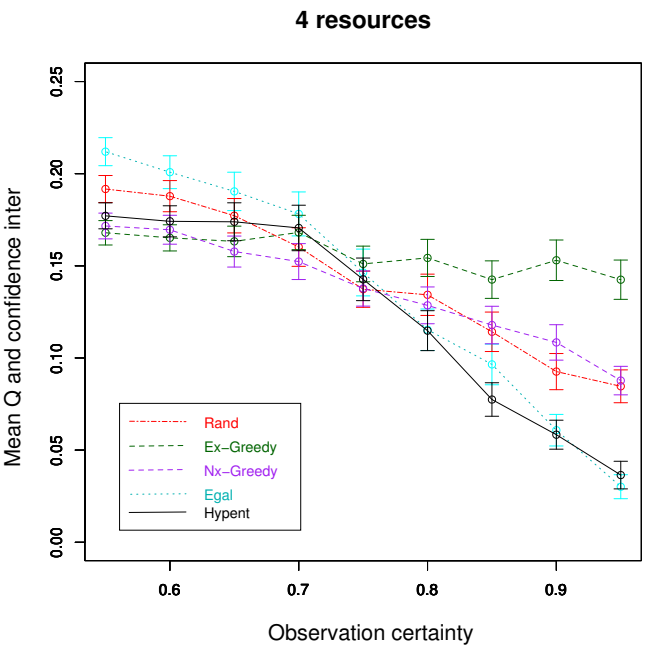

Figure 6:
The result of comparing the strategies using four resources

Strategy	#resources	Running time [ms]
Rand	1-4	< 1
Egal	1-4	< 1
Ex-Greedy	1-4	< 1
Nx-Greedy	1-4	< 1
Hypent	1	24
Hypent	2	60
Hypent	3	123
Hypent	4	218

Table 2: The average running times for the different information acquisition strategies and the number of resources to deploy.

Obviously, one disadvantage of the Hypent strategy is its running time. The Egal strategy has a time complexity linear in the number of indicators M (always assigning one resource to each indicator), i.e., $\mathcal{O}(c_e M)$, with a small constant c_e. The greedy strategies also have a linear complexity, $\mathcal{O}(c_g M)$, as they only evaluate each indicator variable, but with a slightly higher constant, c_g (calculating and comparing entropies). The current implementation of Rand selects one sensing action from the complete set of possible sensing actions. As the action set increases polynomially with the number of information variables, so does the complexity for Rand, $\mathcal{O}(c_r M^t)$ (for some exponent t). Hypent also iterates over the set of all sensing action, but has a much higher cost per sensing action, c_h, $\mathcal{O}(c_h M^t)$. The differences in running time in practice are exemplified in the Table 2 where it can be seen that the Hypent strategy is considerably more time consuming than the others. Still, for the many decision support applications, the less than one second time required for recommending sensing actions is sufficient. The experiments were run on a Intel Core2 Duo 2.66 GHz CPU and 3GB RAM.

7 Discussion and Future Work

The implementation and experiments described in Section 6 should be seen as a proof of concept. Although the simple prototype BN used in the experiments (Fig. 2) captures some relevant aspects of an intelligence model, i.e., a hypothesis variable and some context and activity information variables, it would be interesting to see if the same results are received in other situations. One way of doing this is to study the behavior of the different sensing action strategies on randomly generated BNs. Alternatively (or complementary), one could study how the strategies perform on a number of manually constructed typical cases. For instance, it is not hard to see how to put together a BN that would make the greedy strategies fail completely. Simply add an indicator with high uncertainty and low impact on the Hypothesis variable, and the greedy strategies will put their resources on this node with a poor result.

Although not proven, it is plausible that humans fall into the same pitfalls as the greedy strategies when facing a complex BN, where it is difficult to predict the impact of observing the different indicators. This motivates further work in the direction pointed out by this paper. It also gives a hint on the importance of visualizing indicator impact to an analyst of a decision support tool such as `Impactorium`. This kind of visualization should play a major role when designing an explanation function of an automatic information acquistion system.

The desirable performance of Hypent comes, of course, at price as revealed in Table 2. Another future direction is therefore to study efficient approximation algorithms to scale from the prototype BN in Fig. 2 to more complex BNs with additional information variables and states.

Acknowledgement

This work was supported by the FOI research project "Tools for information management and analysis", which is funded by the R&D programme of the Swedish Armed Forces. The experiment implementation is based on the SMILE reasoning engine for graphical probabilistic model and the visualizations of Bayesian networks by its visual front-end, GeNIe. Both tools are developed by the Decision Systems Laboratory, University of Pittsburgh (`http://dsl.sis.pitt.edu`). Plots are made with the R scientific programming language (`http://www.r-project.org/`). All supplementary programming was solved with the Python programming language (`http://www.python.org/`).

References

[1] Hei Chan and Adnan Darwiche. On the revision of probabilistic beliefs using uncertain evidence. *AI*, 163(1):67–90, 2005.

[2] Ronnie Johansson. *Large-Scale Information Acquisition for Data and Information Fusion*. PhD thesis, The Royal Institute of Technology, Stockholm, Sweden, March 2006.

[3] C. Mårtenson and P. Svenson. Information supply for high-level fusion services. In *Intelligent Sensing, Situation Management, Impact Assessment, and Cyber-Sensing, SPIE Vol 7352*, 2009.

[4] G. W. Ng and K. H. Ng. Sensor management - what, why and how. *Information Fusion*, 1:67–75, 2000.

[5] Pontus Svenson, Tomas Berg, Pontus Hörling, Mikael Malm, and Christian Mårtenson. Using the impact matrix for predictive situational awareness. In *Proceedings of the 10th International Conference on Information Fusion*, 2007.

[6] Ning Xiong and Per Svensson. Sensor management for information fusion - issues and approaches. *Information Fusion*, 3:163–186, 2002.

[7] Yongmian Zhang, Qiang Ji, and Carl G. Looney. Active information fusion for decision making under uncertainty. In *Proceedings of the 5th International Conference on Information Fusion*, pages 643–650. International Society of Information Fusion, 2002.