Technical White Paper

HUGIN Expert A/S
Gasvaerksvej 5
DK-9000 Aalborg
Denmark

Phone: +45 9655 0790
Fax: +45 9655 0799

info@hugin.com
www.hugin.com

Contents

1 What is Artificial Intelligence ?

Artificial intelligence is hard to define precisely. Instead of giving a precise definition of artificial intelligence, it may be better to list some of the characteristics of devices or services which we will describe as possessing artificial intelligence. For instance, a device or service made by man could be described as acting intelligently if the device or service can

- efficiently solve problems of reasoning and decision making under uncertainty,
- acquire and extract knowledge from data, experience, and experts, and
- adjust the behavior to changes in the surrounding environment and efficiently respond to new situations.

Our goal is to develop model-based decision support systems. Decision support systems can be applied in many different areas of reasoning and decision making. Here we focus on reasoning and decision making under uncertainty. Thus, we want to solve problems or assist people in solving problems of reasoning or decision making under uncertainty using an explicit representation of knowledge and reasoning methods employing that knowledge.

We will base our handling of uncertainty on probability theory and use classical methods of probability theory to reason about the state of a system. Decision theory offers an extension of classical probability theory to a precise mathematical framework for rational decision making. Rational decision making is based on the assumption that the decision maker acts to optimize her or his expected gain or utility. Experience has shown, however, that, in general, people do not adhere to this principle, possibly because of the complicated nature of decision making under uncertainty. For this reason, computer assisted decision making is required.

1.1 Expert Systems

The first expert systems were constructed in the late 1960s. These expert systems were constructed as computer models of experts, e.g. a computer model of a medical doctor. The basic idea was to replace the domain expert with a computer system which modeled the best domain experts and to use the system for repeated decision making in a specific problem domain such as diagnosis of mechanical failures or medical diagnosis, for instance.

Many different paradigms for constructing expert systems exist, e.g. rule-based systems, neural networks, case-based reasoning systems, etc. Rule-based expert systems are built on a knowledge base of production rules and an inference method. Production rules have the form: *if <condition>, then <fact>* or *if <condition>, then <action>*. The production rules are used to perform reasoning from conditions to facts or actions, or in the opposite direction. An expert system for automatic regulation of a thermostat could, for instance, contain a rule of the form *if temperature* > 68 *and temperature* < 72*, then stop.* Production rules are deterministic by nature. In most expert systems there are, however, a need for reasoning under uncertainty, as we often experience in our everyday life. The uncertainty we experience can be due to lack of knowledge of the state of the world, an inherent part of the world, or for some other reason. One way to represent uncertainty in production rules is to include certainty factors *if <condition> with certainty* x*, then <fact> with certainty* $f(x)$. The introduction of certainty factors complicates reasoning and can in the worst case lead to incorrect conclusions. This is a clear weakness of rule-based expert systems.

Production rules is just one out of a number of different paradigms for building expert systems.

1.2 Normative Expert Systems

The framework of normative expert systems offers an alternative to traditional expert systems. Normative expert systems are - just like traditional expert systems - constructed for repeated

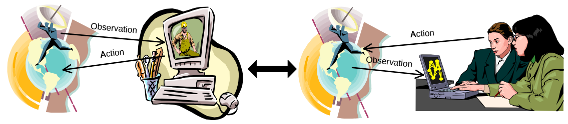

Figure 1: An illustration of the difference between traditional expert systems and normative expert systems.

decision making in a specific problem domain, but normative expert systems differ from (rule-based) expert systems on three fundamental design issues:

- model the problem domain, not the expert,
- support the expert, do not substitute the expert,
- use classical probability calculus and decision theory to handle uncertainty, not a non-coherent calculus developed for rules.

The basic difference between (rule-based) expert systems and normative expert systems is illustrated in figure 1. The leftmost part of figure 1 illustrates an (rule-based) expert system whereas the rightmost part illustrates a normative expert system. A normative system is intended to support the user of the system in her or his reasoning or decision making.

Normative expert systems are based on the use of classical probability calculus and decision theory for reasoning and decision making under uncertainty. Thus, the inherent uncertainty of reasoning problems is handled using probabilities, and the solution of decision problems is based on the assumption that a decision maker acts to optimize her or his expected gain or utility.

The model used in a normative expert system will describe properties of the problem domain under consideration, e.g. dependence relations between causes, diseases, and symptoms in a medical diagnosis situation. The model of a problem domain in a normative system can, for instance, be represented as a Bayesian network. Since the model describes properties of the problem domain under consideration a single model can be used for both diagnostic and causal reasoning. That is, the same model can be used to reason from effects to causes and from causes to effects.

2 Bayesian Networks and Influence Diagrams

During an ESPRIT project in the 1980s on diagnosing neuromuscular diseases, the Bayesian network MUNIN was constructed. A research group at Aalborg University worked on developing correct and efficient calculation methods for the diagnosis problem. Some results had at that time been obtained by American researchers, but a very obstinate problem still remained which prevented Bayesian networks from being used in the construction of expert systems. The problem was know as the rumour problem: you may hear the same story through several different channels; but still the story may originate from the same source. Without knowing whether or not your channels are independent, you cannot combine the statements correctly. This problem was solved by Danish researchers. These researchers are the founders of Hugin Expert A/S.

2.1 What is a Bayesian Network ?

A Bayesian network (a.k.a. Bayes net, causal probabilistic network, Bayesian belief network, or simply belief network) is a compact model representation for reasoning under uncertainty. A problem domain — diagnosis of mechanical failures, for instance — consists of a number of entities or events. These entities or events are in a Bayesian network represented as random variables. One random variable can, for instance, represent the event that a piece of mechanical hardware in a production facility has failed. The random variables representing different events are connected by directed edges to describe relations between events. An edge between two random variables X and Y represents a possible dependence relation between the events or entities represented by X and Y. An edge could, for instance, describe a dependence relation between a disease and a symptom — diseases cause symptoms. Thus, edges can be used to represent cause-effect relations.

The dependence relations between entities of the problem domain are organized as a graphical structure. This graphical structure describes the possible dependence relations between the entities of the problem domain, e.g. a Bayesian network model for diagnosing lung cancer, tuberculosis, and bronchitis would describe the cause-effect relations between the possible causes of these diseases, the diseases, and the possible symptoms of these diseases.

The uncertainty of the problem domain is represented through conditional probabilities. The conditional probability distributions specify our belief about the strengths of the cause-effect relations, e.g. lung cancer does not always produce a positive (bad) chest X-ray, or a mechanical failure does not always cause an alarm to sound. Thus, a Bayesian network consists of a qualitative part which describes the dependence relations of the problem domain and a quantitative part which describes our belief about the strengths of the relations.

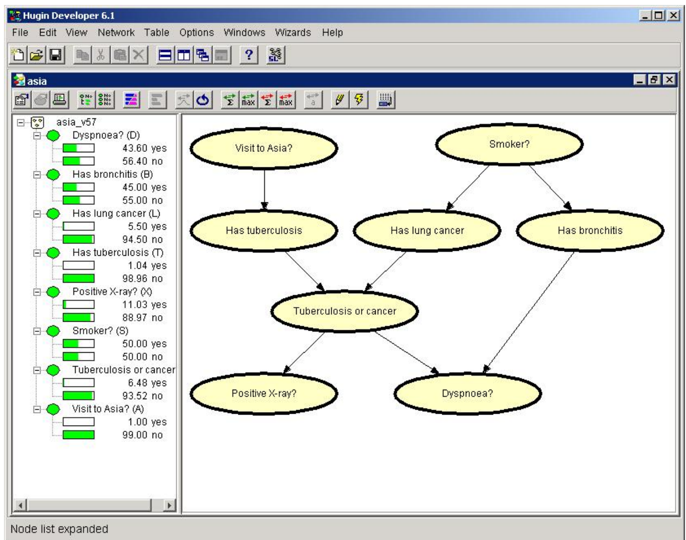

Figure 2: A graphical representation of the qualitative medical knowledge in the Chest Clinic example.

The qualitative or graphical part describes dependence and independence relations among a set of entities of the problem domain. The entities of the problem domain under consideration are represented as random variables which graphically are depicted as nodes in a graph. The quantitative part describes the strengths of the dependence relations using conditional probability distributions. The graphical structure represents a number of conditional independence assertions. A Bayesian network represents a multiplicative decomposition of a joint probability distribution where the decomposition consists of a set of conditional probability distributions. In short, a Bayesian network is a compact representation of a joint probability distribution over a set of variables.

Let $N(G, \mathcal{P})$ be a Bayesian network with graph $G = (V, E)$ and conditional probability distributions $\mathcal{P}$. Then, N encodes a joint probability distribution over the nodes V of G. This joint probability distribution is:

$$P(V) = \prod_{X \in V} P(X \mid pa(X)),$$

where $pa(X)$ are the parents of X in G.

Imagine a medical diagnosis situation where a medical doctor wants to examine her patients with respect to three diseases: tuberculosis, lung cancer, and bronchitis. From consultations with her patients, the doctor gathers information on each patient. This is information like whether or not the patient is a smoker, the result of a single chest X-ray, if the patient has recently been on a visit to Asia, and whether or not the patient is suffering from shortness-of-breath (dyspnoea). The following example describes this hypothetical medical diagnosis situation where a patient consults a chest clinic. The fictitious qualitative medical knowledge is:

> Shortness-of-breath (*dyspnoea*) may be due to *tuberculosis*, *lung cancer* or *bronchitis*, or none of them, or more than one of them. A recent *visit to Asia* increases the chances of tuberculosis, while *smoking* is known to be a risk factor for both lung cancer and bronchitis. The results of a single *chest X-ray* do not *discriminate* between lung cancer and tuberculosis, as neither does the presence or absence of dyspnoea.

The qualitative medical knowledge can be represented as a Bayesian network as shown in figure 2. In order to have a fully specified Bayesian network the strengths of the relations have to be specified, i.e. the conditional probability distributions. The leftmost part of the figure shows, the prior probability distribution of each of the variables represented in the model. For instance, the prior belief that any patient visiting the chest clinic is a smoker is 50%. This Bayesian network model can support the medical doctor of the chest clinic in her reasoning about whether or not a patient suffers from either bronchitis, lung cancer, or tuberculosis.

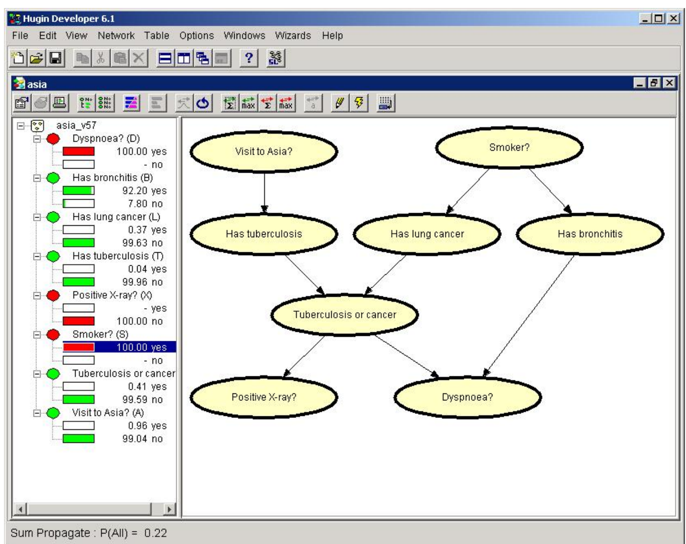

Figure 3: A particular patient is being diagnosed the chest clinic.

Usually, we do not have complete knowledge about the state of the world, i.e. there are some things we do not know for certain. An observation is a piece of knowledge about the exact state of the world. When we make observations or in some other way obtain additional knowledge about the state of the world, we use this knowledge to update our belief about the state of the world. If the medical doctor, for instance, makes the observation that a patient is suffering from dyspnoea, then she has a higher belief that the patient is suffering from lung cancer or bronchitis than had the patient not suffered from dyspnoea. This is a typical example of reasoning under uncertainty:

> A smoking patient with dyspnoea has visited the chest clinic. The result of a single X-ray is negative. Therefore, the medical doctor has a high posterior belief that the patient is suffering from bronchitis.

The result of inference with these observations is shown in figure 3. The figure shows that the observations changed our beliefs about the state of the world. In light of the observations the probability of the patient suffering from bronchitis is very high.

A Bayesian network can be used to compute the probability of different events or hypotheses given a number of observations, e.g. how likely is it that the patient is suffering from lung cancer given that we know she has recently been on a visit to Asia and that the result of a single X-ray was positive ? This kind of query can be solved efficiently using a Bayesian network.

Our updated belief about the state of the world is represented as a set of posterior probability distributions. The conditional probability distribution $P(X \mid \epsilon)$ where X is a variable and ϵ is the set of evidence is computed for each variable in the model.

The main reason for reasoning under uncertainty is to make better decisions. Influence diagrams are Bayesian networks augmented with decision variables and utility functions. Decision variables specify points in time where a decision is to be made and utility functions specify the preferences of the decision maker over different states of the world. A decision variable can, for instance, represent a decision on a treatment of a mildew attack of a wheat field. The options available at the time of decision will be the different treatments with fungicides available. One utility function can specify the cost of the different treatments and another utility function can specify the influence of the quality of the wheat field on the outcome of the harvest.

Let $N = (G, \mathcal{P}, \mathcal{U})$ be an influence diagram with graph $G = (V, E)$, conditional probability distributions $\mathcal{P}$, and utility functions $\mathcal{U}$. Then, N encodes a joint expected utility function over the decision nodes V_D and chance nodes V_C of G. This joint expected utility function over $V_D \cup V_C$ is:

$$EU(V_C, V_D) = \prod_{X \in V_C} P(X \mid pa(X)) \sum_{U \in V_U} f(pa(U)),$$

where $pa(X)$ are the parents of X, $pa(U)$ are the parents of U, and V_U are the utility nodes of G such that $V = V_C \cup V_D \cup V_U$. The following example describes a common type of decision problem which can be represented as an influence diagram:

> An oil wildcatter must decide either to *drill* or *not to drill*. He is uncertain whether the *hole* is dry, wet, or soaking with oil. The wildcatter could *take seismic soundings* that will help determine the *geological structure* of the site. The *soundings* will give a closed reflection pattern (indication of much oil), an open pattern (indication of some oil), or a diffuse pattern (almost no hope of oil).

The oil wildcatter has two decisions. A decision on whether or not to perform a seismic soundings test to determine the geological structure of the site, and a decision on whether or not to drill for oil. The last decision will be based on the outcome of the first decision. Figure 4 shows an influence diagram representation of the oil wildcatter decision problem.

The influence diagram contains two chance nodes (the ovals), two decision nodes (the rectangles), and two utility nodes (the diamonds). The influence diagram shows that the decision maker should drill for oil if the outcome of the seismic soundings test is an open pattern as the decision option *drill* has the highest expected utility (22.86) for the drill decision.

Influence diagrams are compact and intuitive graphical models for reasoning about decision making under uncertainty. In essence an influence diagram is a Bayesian network with explicit representations of decisions and utilities. An influence diagram can be used to efficiently compute the optimal strategy consisting of a policy for each decision in the decision problem. Each decision policy $\delta(D \mid rl(D))$ where D is a decision and $rl(D)$ is the relevant past of D specifies the optimal decision at D for each possible instantiation of $rl(D)$.

2.2 What is the Foundation of Bayesian Networks ?

The foundation of Bayesian networks is the following theorem known as Bayes' Theorem:

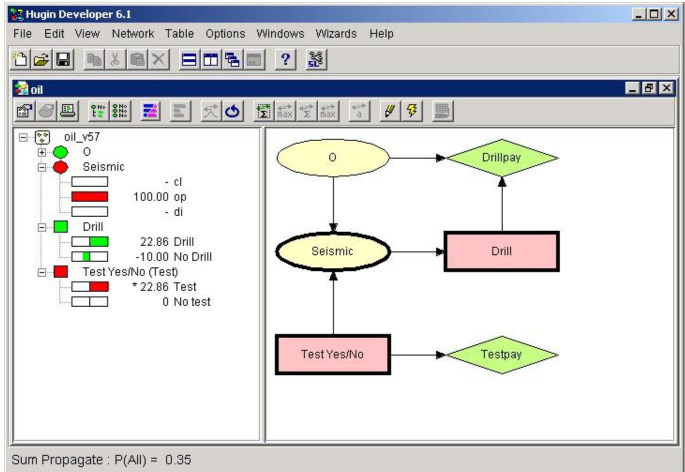

Figure 4: A graphical representation of the decision problem faced by the oil wildcatter.

$$P(H \mid E, c) = \frac{P(H \mid c)P(E \mid H, c)}{P(E \mid c)}.$$

It is named after Reverend Thomas Bayes (1702-1761) an 18th century Nonconformatist minister from England who derived a special case of this theorem, see figure 5.

Figure 5: Reverend Thomas Bayes.

The derivations made by Bayes were published in 1763, two years after his death. Exactly what Bayes intended to do with the calculation, if anything, still remains a mystery. However, the theorem, as generalized by Laplace, is the basic starting point for inference problems using probability theory as logic. The theorem describes how to update our beliefs about the state of the world in the light of observations.

2.3 Why use Bayesian Networks ?

The framework of Bayesian networks offers a compact, intuitive, and efficient graphical representation of dependence relations between entities of a problem domain. The graphical structure reflects properties of the problem domain in an intuitive way which makes it easy for non-experts of Bayesian networks to understand and express knowledge in this kind of knowledge representation.

Inference in Bayesian networks is based on a coherent and mathematically sound handling of uncertainty and decisions. The foundation for reasoning is Bayes' Theorem, which was derived many years prior to the development of graphical models.

The construction of a Bayesian network consists of two phases, namely the identification or specification of the cause-effect relations of the domain considered and a specification of the strengths of these relations. Bayesian networks can either be constructed manually (e.g. by or in cooperation with domain experts), automatically from a database of observations, or via a combination of the two.

It is often criticized that in order to construct a Bayesian network you have to *know* too many probabilities. The number of probabilities to be specified depends on the connectivity of the graph of the Bayesian network. The necessary probabilities can either be assessed from experts with domain knowledge or they can be estimated from observations on entities of the problem domain, i.e. learned from a database of cases where some values may be missing. Thus, when constructing a Bayesian network, it is possible to utilize both background knowledge such as expert knowledge and knowledge stored in databases — data and knowledge can be fused.

Sequential updating makes it possible to update and improve the conditional probability distribution of the Bayesian network as observations become available. That is, the model can be updated as it is used. This is important if the model is incomplete, the modeled domain is drifting over time, or if the model quite simply does not reflect the domain properly. For instance, a model is deployed in similar, but different settings or environments. Using sequential updating the model can adjust to the particular setting in which the model is used. This is a very powerful feature.

Continuing the chest clinic example: After completing the consultation with the patient, the medical doctor has gained more insight into the strengths of the relations between the causes, diseases, and symptoms of the problem domain. The medical doctor would like to incorporate this newly acquired knowledge into the model of the problem domain. This is performed through sequential learning (also known as adaptation). The result of adapting the model in figure 2 is shown in figure 6. For instance, we can see that after having seen a smoking patient our belief that an average patient is smoking has increased.

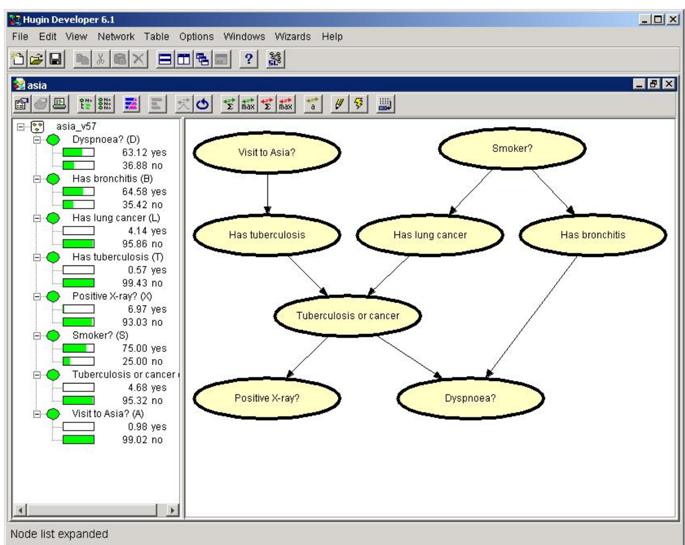

Figure 6: After the consultation the Bayesian network is updated.

The compactness and efficiency of Bayesian network models have been exploited to develop efficient algorithms for solving queries against a model. Queries like *what is the probability that device X is the cause of the mechanical failure of the production line ?* and *what is the probability that a person applying for a loan will repay this loan given that we know the age, gender, income, financial status of the person ?* can be answered efficiently. Queries against a Bayesian network model are solved efficiently by local computation. The task of solving a query is to compute the posterior probability distribution of each unobserved variable given a set of observations. The effects of the observations are entered and propagated throughout the Bayesian network by local computations. Influence diagrams are solved using a minor extension of the algorithm for solving queries against a Bayesian network model.

We may not be satisfied with having computed the answer to a query. In some case we may want to analyze the results of a query further. For instance, in medical diagnosis situations where a patient has been assigned a dangerous or high risk treatment, the patient would like to have an explanation of why she or he needs this treatment. This is supported by Bayesian networks. An explanation of the answer to a query against a Bayesian network model can be formulated using the graphical structure of the model. The graphical structure of the model represents dependence and independence relations between the entities of the problem domain, but these dependence relations change as observations are made. This is exploited to generate explanations of the reasoning performed by the system.

Similarly, some of the observations we make about the state of the world may be conflicting. The results of two different tests may be conflicting such that one result indicate that the patient is not suffering from a disease whereas the other result does. Data conflict analysis can be used to identify, trace, and resolve possible conflicts in the observations made.

During the interview of a person applying for a loan, the banker may be concerned with whether or not the person is actually going to repay the loan. During this interview the banker gathers information about the applicant. If, based on this information, the banker has a high belief that the person is going to repay the loan, then the banker may wonder how sensitive her conclusion is to the answers supplied by the applicant — *what if the applicant had answered differently to some question ?* This sensitivity analysis can be performed using Bayesian networks. Sensitivity analysis can be performed both with respect to the observations, but also with respect to the conditional probabilities of the model.

In a decision making scenario it may be beneficial for the decision maker to acquire additional information before a decision is made. An example is a decision on whether or not to drill for oil at specific site. The result of an additional test may change the decision, but is it worth the cost to perform the test ? This kind of value of information analysis is also supported. In fact a large number of different techniques can be applied to analyze the results obtained from queries against a Bayesian network model.

2.4 Bayesian Network Terminology

As mentioned above, Bayesian network is a compact and intuitive graphical model for reasoning under uncertainty. The graph of a Bayesian network describes possible causal or influential relations (e.g. causal relations between diseases and symptoms) and independence relations (e.g. a certain disease does not cause all symptoms) between entities of the modeled problem domain. Entities are represented using random variables. The strengths of the edges are specified using conditional probabilities (e.g. it is not always the case that a disease causes a particular symptom).

A Bayesian network supports causal and diagnostic reasoning and it is an efficient model for complex chains of reasoning under uncertainty. A Bayesian network can be used to efficiently compute the posterior probability of any event given any subset of other events.

Structural learning is the task of identifying the dependence and independence relations of the Bayesian network from data.

- There may be missing values in data,
- domain knowledge can be exploited to constrain the learning, and
- the user can interact with the learning process.

Background knowledge or domain expert knowledge may be available in the form of known structural relations between entities of the problem domain such as

- knowledge about the existence/non-existence and direction of edges.

Parameter learning is the task of estimating the strengths of the relations from data, i.e. the conditional probability distributions.

- There may be missing values in data,
- domain knowledge can be exploited to guide and speed-up the learning, and
- learning can be turned off for any variable.

Inference in a Bayesian network is performed according to Bayes' Theorem. That is, probability values are updated by applying Bayes' Theorem and propagating the results throughout the model.

- The probability of any event given any subset of events (i.e. no notion of input/output),
- the probability of the observations,
- the configuration with maximum probability, and
- the probability of any subset of events can be computed.

Adaptation is the task of sequentially updating the strengths of the dependence relations of the model as the model is being used, i.e. updating the conditional probability distributions. This allows for the model to adjust to the local settings in which it is used.

- There are often only a few observations in a case,
- the belief in prior distributions can be specified,
- the impact of the past can be faded away, and
- adaptation can be turned off for any variable.

Analysis of the results obtained using a Bayesian network can be performed efficiently. Many different types of questions can be answered using different kinds of analyses.

- Why the system arrive at the conclusion it did (explanation of results) ?
- are any of the observations which have been made in conflict (conflict analysis) ?
- which observations support the hypothesis, which do not, and how sensitive are the results to the model parameters (sensitivity analysis) ? and
- if I could get more information, then what information should I get (value of information analysis) ?

2.5 How are Bayesian Networks Used ?

Bayesian networks can and have been used as components for reasoning under uncertainty in large and complex systems. Consider, for instance, a large medical diagnosis system available to medical doctors through the Internet. Such a system could consist of a large number of components where each component can be used to diagnose a set of different but related diseases. Each medical doctor could through a computer interact with the system when diagnosing patients in order to make better diagnoses or to confirm a diagnosis. One of the components of such a large and complicated system could, for instance, be a component for diagnosing lung cancer, bronchitis, and tuberculosis like the Chest-Clinic example, see figure 7. Another component could support diagnosis of different diseases.

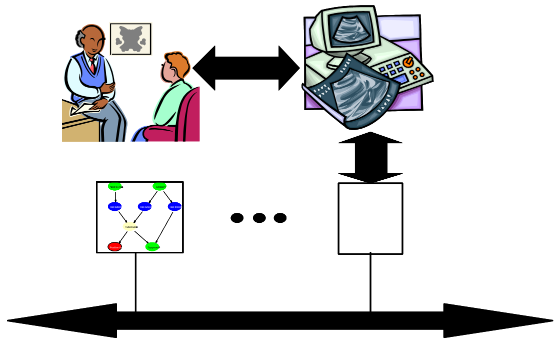

Figure 7: Bayesian networks are most often used as components for reasoning under uncertainty in complex systems or applications.

2.6 What have Bayesian Networks been Used for ?

Bayesian networks have been applied for reasoning and decision making under uncertainty in a large number of different settings. A few of the applications are indicated in the list below:

Medicine diagnosis of muscle and nerve diseases, antibiotic treatment, diabetes advisory system, medical triage.

Software software debugging, printer troubleshooting, safety and risk evaluation of complex systems, help facilities in Microsoft Office products.

Information Processing information filtering, display of information for time-critical decisions, fault analysis in aircraft control.

Industry diagnosis and repair on-board unmanned underwater vehicles, control of centrifugal pumps, process control in waste water purification, forensic identification.

Economy credit application evaluation, port folio risk and return analysis.

Military NATO airborne early warning & control program, situation assessment.

Agriculture blood typing and parentage verification of cattle, replacement of milk cattle, mildew management in winter wheat.

For more information on applications you may wish to visit our web-site:

`http://www.hugin.com`

3 The Hugin Tools

Two different tools for building, maintaining, and manipulating graphical knowledge bases are available from Hugin Expert A/S. One tool is the Hugin Graphical User Interface which has been built on top of the Hugin Decision Engine. Both tools are tools for construction, usage, revision, analysis, and documentation of Bayesian networks and influence diagrams.

Construction refers to all aspects of the construction of a Bayesian network or an influence diagram.

Usage refers to solving queries against the model, e.g. performing probabilistic inference or solving an influence diagram.

Revision refers to changing the model as it is used, e.g. sequential learning of probabilities.

Analysis refers to analysis of the solution to a query as generated by a model, e.g. conflict analysis, sensitivity analysis, and value of information analysis.

Documentation refers to the documentation of a model, e.g. grouping and coloring of nodes.

3.1 The Hugin Decision Engine

The Hugin Decision Engine is a library of functions. These functions support construction, usage, analysis, documentation, and revision of Bayesian networks. The Hugin Decision Engine contains a compiler and an inference engine. The inference engine supports reasoning in graphical models, e.g. Bayesian networks and influence diagrams. The Hugin Decision Engine is available for the C, C++, and Java programming languages and as an ActiveX-server.

The Hugin Decision Engine is provided in two versions: a version using single-precision floating point arithmetic and a version using double-precision floating point arithmetic. The double-precision version may prove useful in computations with continuous random variables (at the cost of a larger space requirement). Models saved in the Hugin knowledge base-format using a single precision version can be loaded by a double-precision version of the Hugin Decision Engine, and vice versa.

The Hugin Decision Engine has an efficient error handling mechanism. Whenever a Hugin Decision Engine operation fails, an error indicator (an error code) is returned — it is then up to the application to decide what action to take. Associated with each error code is an error description. The specifics of the error handling mechanism depends on the programming interface.

An extensive reference manual is available for the Hugin Decision Engine. Furthermore, for each of the C++, Java, and ActiveX-server programming environments additional html-documentation is available.

A demo version of the Hugin Decision Engine be downloaded from our web-site at:

`http://www.hugin.com/Products_Services/Products/Demo/`

A demo version for each of the platforms mentioned above is available. For examples of the usage of the Hugin Decision Engines you may wish to visit our developer web-site:

`http://developer.hugin.dk`

A number of tutorials is also available on our web-site.

3.2 The Hugin Graphical User Interface

The Hugin Graphical User Interface is basically a graphical interface to the Hugin Decision Engine. Examples of the Hugin Graphical User Interface has been shown in a number of figures above. The graphical user interface is currently available for Microsoft Windows, Sun Solaris, and Linux Red Hat platforms. Through the Hugin Graphical User Interface all functionality (except very few functions) of the Hugin Decision Engine can be accessed.

A lot of help facilities in the form of html pages are available for the Hugin Graphical User Interface. These help facilities are included in the installation of the Hugin Graphical User Interface. The Hugin Graphical User Interface can be customized to meet the preferences of the user.

A demo version — Hugin Lite — of the Hugin Graphical User Interface can be downloaded from our web-site at:

`http://www.hugin.com/Products_Services/Products/Demo/`

The Hugin Lite includes demo versions of all interfaces to the Hugin Decision Engine.

A number of tutorials are also available at our web-site.

3.3 The Hugin Advisor

The Hugin Advisor is a unique technology particularly suitable for troubleshooting applications. Hugin Advisor provides the ideal set of tools for constructing and deploying decision support systems, including, but not limited to, the following application areas:

- Call center services.
- Process line diagnostics.
- Help desk services.
- Self-service.
- On-line consulting.
- Tutoring.

A demo version — Hugin Advisor Evaluation — of the Hugin Advisor application can be downloaded from our web-site at:

`http://www.hugin.com/Products_Services/Products/Demo/`