9

Dynamic data feed to Bayesian network model and SMILE web application

Nipat Jongsawat, Anucha Tungkasthan and Wichian Premchaiswadi
Graduate School of Information Technology in Business, Siam University
Thailand

1. Introduction

Constructing Bayesian network models is a complex and time consuming task. It is difficult to obtain complete and consistent models but to get the correct and reliable probability data for the designed models is much more difficult. Normally, there are two methods to enter the probability values into the chance node of a Bayesian network model. The first method is to consult an expert for the probability values and enter them into the models. The second method is to obtain probability values from statistical or learned data (Druzdzel et al., 2001). Both methods use static data, not dynamic data. The second method acts like dynamic data but it is actually not. The statistical data from a database need to be loaded and processed each time to get the probability values. This works similar to batch processing. Finally, users still need to enter probability values into the model by manual feeding the data by hand. It is not possible to have real-time processing. The probability values are fed to every node of the model and the joint probability distribution is computed at the final stage when the model is performing Bayesian updates. The disadvantage of using manually fed data or static data is that it cannot be performed in using real-time processing, monitoring, and updating.

In this article, we propose a technique for feeding data into the Bayesian network model dynamically. A case study of several factors that have an impact on students for making a decision in enrollment is selected as the case for an application implementation of a Bayesian network model. The probability values for each node are calculated from student's data and then transferred into the model dynamically. A SMILE web-based application provides a user friendly web interface for Bayesian inference. It provides the feature set of Bayesian diagnosis for the user. The SMILE web-based application was developed based on SMILE (Structural Modeling, Inference, and Learning Engine) and SMILE.NET. SMILE is a reasoning engine that is used for graphical probabilistic models and provides functionality to perform diagnosis. SMILE.NET is used for accessing the SMILE library from the web-based interface. Using SMILE application, users can also perform Bayesian inference in the model and they can compute the impact of observing values of a subset of the model variables on the probability distribution over the remaining variables based on real-time data. Using the other BN software tools for constructing a Bayesian network model, there are some limitations such as dependent platform and is unusable on a global basis. Fig. 1

shows a generic implementation for dynamic data feed to Bayesian network model and SMILE web application.

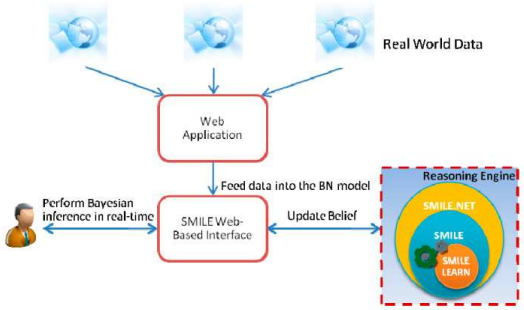


Fig. 1. A Generic implementation for dynamic data feed to BN model and SMILE web application

2. Fundamentals

This section is intended to describe the fundamentals and techniques for implementing a Bayesian network model in general. They are the followings:

2.1 Bayesian Network

Bayesian networks (also called belief networks, Bayesian belief networks, causal probabilistic networks, or causal networks) (Pearl, 1988) are acyclic directed graphs in which nodes represent random variables and arcs represent direct probabilistic dependencies among them. The structure of a Bayesian network is a graphical, qualitative illustration of the interactions among the set of variables that it models. The structure of the directed graph can mimic the causal structure of the modeled domain, although this is not necessary. When the structure is causal, it gives a useful, modular insight into the interactions among the variables and allows for prediction of the effects of external manipulation.

Nodes of a Bayesian network are usually drawn as circles or ovals. The following simple Bayesian network, shown in Fig. 2, represents two variables, Curriculum and Enrollment, and expresses the fact that they are directly dependent on each other.

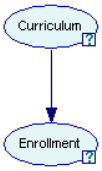


Fig. 2. An example of Bayesian network

A Bayesian network also represents the quantitative relationships among the modeled variables. Numerically, it represents the joint probability distribution among them. This distribution is described efficiently by exploring the probabilistic independence among the

modeled variables. Each node is described by a probability distribution conditional on its direct predecessors. Nodes with no predecessors are described by prior probability distributions. For example, the node Curriculum shown in Fig. 2 will be described by a prior probability distribution over its two outcomes: Impact and NoImpact. See Fig. 3 below.

	Impact	0.7
	NoImpact	0.3

Fig. 3. Prior probability distribution for a curriculum node

The enrollment node will be described by a probability distribution over its outcomes (Enroll, NotEnroll) conditional on the outcomes of its predecessor (node Curriculum outcomes, Impact and NoImpact). See Fig. 4 below.

Curriculum	CurriculumImpact	CurriculumNoImpact
Enroll	0.7	0.4
NotEnroll	0.3	0.6

Fig. 4. Conditional probability values for an enrollment node

Both the structure and the numerical parameters of a Bayesian network can be elicited from an expert. They can also be derived from data, as the structure of a Bayesian network is simply a representation of independencies in the data and the numbers are a representation of the joint probability distributions that can be inferred from the data. Finally, both the structure and the numerical probabilities can be a mixture of expert knowledge, measurements and objective frequency data.

2.2 Bayesian Updating

Bayesian updating, also referred to as belief updating, or somewhat less precisely as probabilistic inference is based on the numerical parameters captured in the model (Cooper, 1990). The structure of the model which is an explicit statement of the independencies in the domain helps in making the algorithms for Bayesian updating more efficient (Dagum & Luby, 1997). All algorithms for Bayesian updating are based on a theorem proposed by Rev. Thomas Bayes (1702-1761) and is known as Bayes Theorem.

Belief updating in Bayesian networks is computationally complex. In the worst case, belief updating algorithms are NP-hard (Cooper, 1990). There exist several efficient algorithms, however, that make belief updating in graphs consisting of tens or hundreds of variables tractable. Pearl developed a message-passing scheme that updates the probability distributions for each node in a Bayesian network in response to observations of one or more variables (Pearl, 1986). Lauritzen and Spiegelhalter, Jensen et al, and Dawid proposed an efficient algorithm that first transforms a Bayesian network into a tree where each node in the tree corresponds to a subset of variables in the original graph (Lauritzen & Spiegelhalter, 1988; Jensen et al., 1990; Dawid, 1992). The algorithm then exploits several mathematical properties of this tree to perform probabilistic inference.

Several approximate algorithms based on stochastic sampling have been developed. Of these, best known are probabilistic logic sampling (Henrion, 1988), likelihood sampling (Shachter & Peot, 1989; Fung & Chang, 1989), and backward sampling (Fung & del Favero,

1994), Adaptive Importance Sampling (AISBN) (Cheng & Druzdzel, 2000), and Approximate Posterior Importance Sampling (APIS-BN) (Yuan & Druzdzel, 2003). Approximate belief updating in Bayesian networks has also been shown to be worst case NP-hard (Dagum & Luby, 1993).

2.3 SMILE and SMILE.NET

The core reasoning engines of the SMILE web-based application development capability consist of SMILE and SMILE.NET. SMILE is a reasoning engine that is used for graphical probabilistic models and provides functionality to perform diagnosis. SMILE.NET is used for accessing the SMILE library from the web-based interface. This section provides some more detailed information about SMILE and SMILE.NET wrapper.

SMILE (Structural Modeling, Inference, and Learning Engine) is a fully platform independent library of functions implementing graphical probabilistic and decision-theoretic models, such as Bayesian networks, influence diagrams (IDs), and structural equation models (Druzdzel, 1999). Its individual functions, defined in the SMILE Application Programmer Interface (API), allow creating, editing, saving, and loading graphical models, and using them for probabilistic reasoning and decision making under uncertainty. SMILE can be embedded in programs that use graphical probabilistic models as their reasoning engines. Models developed in SMILE can be equipped with a user interface that best suits the user of the resulting application. SMILE is written in C++ in a platform-independent manner and is fully portable. Model building and the reasoning process are under full control of the application program as the SMILE library serves merely as a set of tools and structures that facilitates them. The sample source code below is the main function of SMILE that contains the core functions of the implemented model SMILE.

```
int main()
{
   CreateNetwork();
   InfereceWithBayesNet();
   UpgradeToInfluenceDiagram();
   InferenceWithInfluenceDiagram();
   ComputeValueOfInformation();
   return(DSL_OKAY);
};
```

SMILE.NET is a library of .net classes for reasoning about graphical probabilistic models, such as Bayesian networks and influence diagrams. It can be embedded in programs that use graphical probabilistic models as a reasoning engine. It is a wrapper library that enables access to the SMILE and SMILEXML C++ libraries from .net applications. SMILE.NET is not limited to stand-alone applications. It can also be used on the back-end side of a multi-tiered application.

2.4 GeNIe

The GeNIe's name and its uncommon capitalization originate from the name Graphical Network Interface, given to the original simple interface to SMILE, the library of functions

for graphical probabilistic and decision-theoretic models (Druzdzel, 1999). GeNIe is a development environment for building graphical decision-theoretic models. It is implemented in Visual C++ and draws heavily on MFC (Microsoft Foundation Classes). It allows for building models of any size and complexity, limited only by the capacity of the available memory of the computer. The original interface was designed for SMILE which is described in a previous section. It may be seen as an outer shell to SMILE. It provides numerous tools for users such as an interface to build Bayesian network models or influence diagrams, to learn the causal relationships of a model using various algorithms, and to perform model diagnosis. In order to use GeNIe efficiently, the GeNIe software must be installed and the user should have some background knowledge about probabilistic graphical models and become familiar with the tools provided in GeNIe. Fig. 5 shows the main interface of GeNIe program.

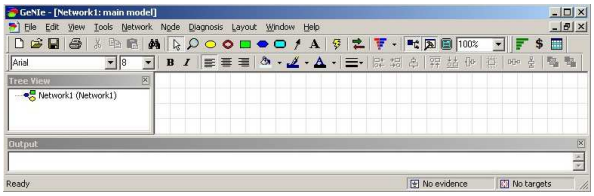


Fig. 5. The main GeNIe interface

3. Graphical Bayesian Network Model

3.1 Bayesian network model in GeNIe

In the first phase, we develop and test the graphical Bayesian network model in GeNIe as shown in Fig. 6. The students' attitude on several factors in an enrollment decision has been proposed as a case study for the model. This model contains ten variables or nodes. There are nine parent nodes thus there are no predecessor nodes and one child or predecessor node. The outcomes of each parent node are identical. It consists of impact and no impact values. There are also two outcomes for the child node (the enrollment node), enroll and not enroll values. The probability values for each parent node and the values for each state combination with an enrollment node are further defined by an expert.

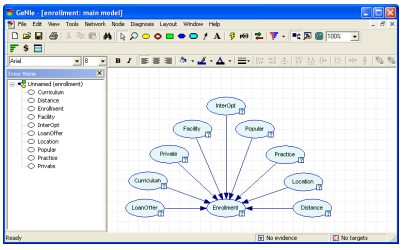


Fig. 6. Graphical Bayesian network model in GeNIe

When the specified outcome of each node and their probability values are defined, the belief updating is ready. The belief update allows for performing Bayesian inference. It is used to

compute the impact of observing values of a subset of the model variables on the probability distribution over the remaining variables. Working with this model and performing Bayesian inference, we can answer simple questions. For example, the question: "What is the chance for the impact for every parent node if the expert judges the prospects for impact to be enroll?" The evidence for the enrollment variable is set at the value of "enroll" as shown in Fig. 7. We have observed a value of the enrollment variable and ask it to update its probability distribution over all parent variables. The result is shown in Fig. 8.

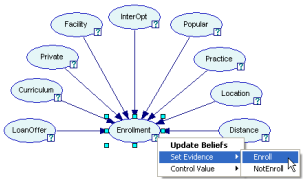


Fig. 7. Setting evidence at enroll outcome for an enrolment node

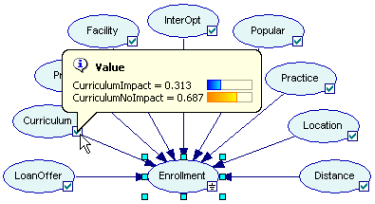


Fig. 8. The posterior probability distribution over a curriculum node

Constructing a Bayesian network model in GeNIe is simply done. There are a lot of tools provided in GeNIe for working and implementing a model but GeNIe has some limitations. Firstly, GeNIe only runs under the Windows operating systems. GeNIe is implemented in Visual C++ and draws heavily on the MFC (Microsoft Foundation Classes), which runs only on a Windows platform. It does not support cross-platform, web or an Internet-based application environment so that there are some limitations for its use on a worldwide basis. Secondly, the probability value of each variable node must be entered manually. This means that the probability determination method must be done before using GeNIe. The probability values can be obtained by asking the experts, statistical methods, or learned data from a database. However, the probability values are still put into the model by hand because GeNIe itself cannot support real-time or dynamic data. Thirdly, a graphical presentation such as pie chart or bar chart in GeNIe is intentionally designed for displaying an individual node. It does not present an overview or comparison for similar outcomes of all nodes. Lastly, the model in GeNIe is static, not dynamic. The model needs to be loaded, have some values changed, and observe the results after updating beliefs one at a time.

3.2 Client/server architecture for SMILE web application

To overcome these limitations of GeNIe mentioned in 3.1. We designed the SMILE web application that works similar to GeNIe. GeNIe is the interface to SMILE for a windows platform. The SMILE web application is the interface of SMILE on the web or an Internet-

based platform. It means that the SMILE web application can support real-time data processing that GeNIe cannot. It also supports a dynamic data feed into the model. See Client/Server Architecture of the SMILE web application in Fig. 9.

Fig. 9. Client/server architecture of SMILE web

In the client/ server architecture of the SMILE web application, the client web application is designed in order to collect data from students through an online questionnaire. The data from the client is sent over the Internet to the server. The server web application or SMILE web is designed to handle incoming data, calculate probability values and put them into each chance node, construct the Bayesian network model in .xdsl file format, feed the calculated probability values into the model, call the core functions of SMILE, read and update probability values for each node in database, send all parameters to SMILE, receive values from SMILE and visualize the results. Both the client and server web application are implemented in the ".NET" environment. Web pages are created by ASP.NET and the code behind is developed in visual C#.net. The code behind the web server application contains the core functions of SMILE such as CreateNetwork(), InfereceWithBayesNet(), and ComputeValueOf Information(). A CreateNetwork function is mainly used for creating the Bayesian network model. This function creates chance nodes, adds arcs from one node to other nodes, and fills in the conditional probability distribution for all nodes in the model. An InfereceWithBayesNet function is used to read the .xdsl file or model, specify the clustering algorithm, update the network or update beliefs, set an evidence for each node and obtain the returned result values. The clustering algorithm in the second function works in two phases: (1) compilation of a directed graph into a junction tree, and (2) probability updating in the junction tree. It has been a common practice to compile a network and then perform all operations in the compiled version. The clustering algorithm is the fastest known exact algorithm for belief updating in Bayesian networks. The clustering algorithm is the SMILE web default algorithm and should be sufficient for most applications. When networks become very large and complex, the clustering algorithm may not be fast enough. In that case, it is suggested that the user choose an approximate algorithm, such as one of the stochastic sampling algorithms. The "ComputeValueOf Information" function is used to compute an expected value of information for the model.

4. Implementation

According to the Client/Server Architecture of SMILE Web mentioned in section 3, SMILE web is designed to work in a more flexible manner for analyzing and diagnosing reasoning. It is designed for worldwide users, who can access the Internet for diagnosing the model. It

overcomes platform dependent, limitations on graphical presentation, and the manual data entry for a Bayesian network model found in GeNIe. To implement SMILE web, there are four main components according to the client/server architecture as follows: 1) Client Web Application, 2) SMILE Server Web Application, 3) Probability Calculation Process, and 4) SMILE Engine.

The first part, client web application, is an online questionnaire designed for prospective students. They are asked to fill out the questionnaire before downloading an application form from the university website. See Fig. 10.

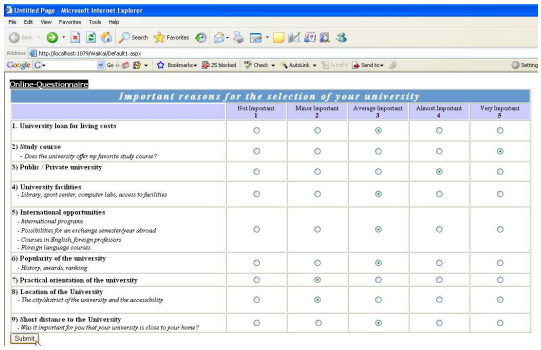


Fig. 10. Online questionnaire for prospected students

The second part, SMILE Server Web Application, is designed for the reasoning aspect of the web user interface for SMILE. Users can update beliefs and perform diagnosis through the SMILE web application as GeNIe did, See Fig. 11 and Fig. 12. The third part, Probability Calculation Process, is actually a probability calculation function in the SMILE web application. It receives the data from client web application (online questionnaire) and processes the probability values in real-time. Moreover, it is responsible for feeding the probability values into the model dynamically. The advantage of this function is that we can get real-time data and probability values for the model that GeNIe could not do. The last part, the SMILE Engine, receives data from the SMILE web application. SMILE's functions such as CreateNetwork(), InfereceWithBayesNet(), and ComputeValueOfInformation () are called to perform according to its operation. The resulting values are sent back to the SMILE web application. The SMILE engine is written in C++ in a platform-independent fashion and is fully portable. The web application's interface is defined in terms of a collection of C++ classes that form the "body" of the library and can be used within an application program. These classes allow building graphical models, editing, saving and loading them, and using them for probabilistic reasoning and decision making under uncertainty.

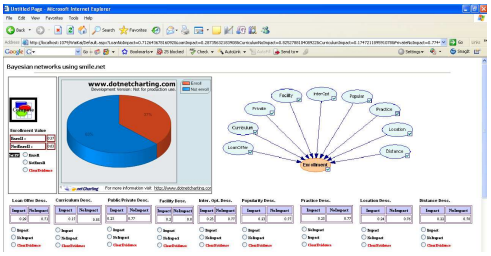


Fig. 11. SMILE web application

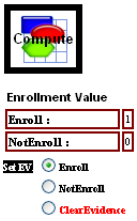


Fig. 12. Setting evidence at enroll outcome for an enrolment node

Users are allowed to perform diagnosis by setting evidence at one variable or node and exploring the probabilistic independencies among the modeled variables. See the sample variables, Public/Private University, Facilities, and International Opportunity, in Fig. 13.

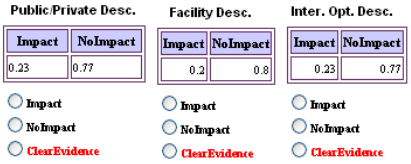


Fig. 13. Three sample nodes for observing values.

The Clear Evidence option is also provided for canceling the diagnosis and going back to use the original values in the calculation. Users can set and clear the evidence at every node in the model in order to perform diagnosis. The graphical representation of SMILE web is shown in Fig. 14, 15, and 16.

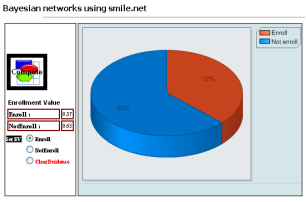


Fig. 14. Pie chart for enrollment node

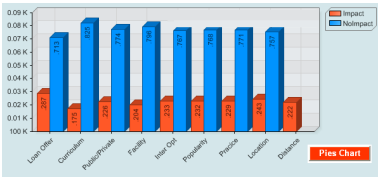


Fig. 15. Bar chart for parent nodes

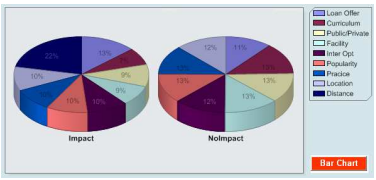


Fig. 16. Pie chart for parent nodes

5. Conclusion

GeNIe, Graphical Network Interface, is designed for a windows environment. It works well on a windows platform. It cannot be run on a web or Internet-based platform. That is why there is some limitation for its use on a worldwide basis. Another thing is that it does not support is real-time data processing. To overcome the limitations of GeNIe, the SMILE web application was designed and implemented on a client/server architecture mentioned in section 3. GeNIe is an outer shell of SMILE. SMILE web is also the outer shell of SMILE. The difference is that the SMILE web application is basically constructed in a web-based environment. SMILE web calls and submits parameters to the core functions of SMILE directly. After processing, SMILE returns all computed values back to SMILE web. SMILE web represents the Bayesian network model on a website. It is the model that users, who access the Internet, can utilize to perform diagnosis. They can update the probability distributions for each variable in a Bayesian networks in response to observations of one or more variables. SMILE web also provides a function to handle dynamic data, compute probability values in real-time, and enter them into the model. This article presents the first step for developing SMILE web application. The next step is to enhance the efficiency of SMILE web by improving the SMILE web interface, including more functions, and increasing the flexibility for model creation. The final phase for SMILE web development will be to enable it to handle influence diagrams and structural equation models. Users can use SMILE web for choosing a decision alternative that has the highest expected gain or utility.

6. Acknowledgement

The authors would like to thank the Decision Systems Laboratory, University of Pittsburgh for supporting documents, and source file of the engines: Structural Modeling, Inference, and Learning Engine (SMILE), SMILEarn, and SMILE.NET wrapper. All necessary files and

documentations have been obtained from the Decision Systems Laboratory's web site. It is available at http://genie.sis.pitt.edu.

7. References

Agniezka O., Druzdzel, M. J., Hanna W., & Warsaw. (2001).Learning Bayesian Network parameters from Small Data Sets", *International Journal of approximate Reasoning*, 27(2), p. 165-182.

Cheng, J. & Druzdzel, M. J. (2000). AIS-BN: An Adaptive Importance Sampling Algorithm for Evidential Reasoning in Large Bayesian Networks. *Journal of Artificial Intelligence Research (JAIR)*, Vol. 13, p. 155-188.

Cooper, G. F. (1990). The Computational Complexity of Probabilistic Inference using Bayesian Belief Networks, *Artificial Intelligent*, Vol. 42, No. 2-3, p. 393-405.

Dagum, P. & Luby, M. (1997). An Optimal Approximation Algorithm for Bayesian Inference, *Artificial Intelligence*, Vol.93, p.1-27.

Dagum, P., & Luby, M. (1993). Approximate probabilistic reasoning in Bayesian belief network is NP-Hard. *Artificial Intelligence*, Vol. 60, p. 141-153.

Druzdzel, M. J. (1999). SMILE: Structural Modeling, Inference, and Learning Engine and GeNIe: A Development Environment for Graphical Decision-Theoretic Models. *In Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI–99)*, p. 902-903, Orlando, FL.

Druzdzel M. J., & Roger R. F. (2002). Decision Support Systems. *Encyclopedia of Library and Information Science*, Second Edition.

Henrion, M. (1989). Some practical issues in constructing belief networks. *In L. N. Kanal, T. S. Levitt, and J. F. Lemmer, editors, Uncertainty in Artificial Intelligence*, 3, p. 161–173.

Gregory, F. C. (1990). The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, 42(2–3), p. 393–405.

Jensen, F. V.; Olesen, K. G. and Andersen, S. K. (1990). An Algebra of Bayesian Belief Universes for Knowledge-Based Systems. *Networks: Special Issue on Influence Diagrams*, Vol.20, No. 5, August 1990, p.637-659.

Lauritzen, S. L. & Spiegelhalter, D. J. (1988). Local Computations with Probabilities on Graphical Structures and their Application to Expert Systems (With Discussion). *Journal of the Royal Statistical Society Series B*, Vol. 50, No 2, p.157-224.

Pearl, J. (1986). Fusion, Propagation, and Structuring in Belief Networks. *Artificial Intelligence*, Vol. 29, No. 3, p. 241-288.

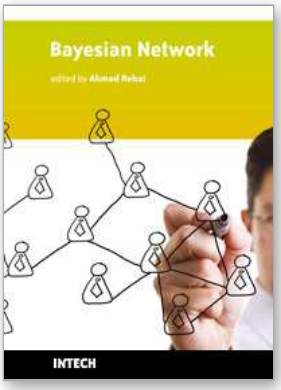
Pearl, J. (1988). Probabilistic Reasoning in Intelligent Systems. *Networks of Plausible Inference,* San Mateo, CA, Morgan Kaufmann Publishers.

Pieter, C., Kraaijeveld, & Druzdzel, M. J. (2005). GeNIeRate: An Interactive Generator of Diagnostic Bayesian Network Models. Air Force Office of Scientific Research under grant F49620-03-1-0187 and by Intel Research.

Shachter, R. D. & Peot, M. A. (1989). Simulation Approaches to General Probabilistic Inference on Belief Networks. *In Uncertainty in Artificial Intelligence*, p. 221-231, New York, N.Y., Elsevier Science Publishing Company, Inc.

Yuan, C. & Druzdzel, M. J. (2003). An Importance Sampling Algorithm Based on Evidence Pre-propagation. *Nineteenth International Conference on Uncertainty in Artificial Intelligence*, Acapulco, Mexico, p. 624-631.

http://genie.sis.pitt.edu
http://genie.sis.pitt.edu/wiki/Probabilistic_Decision_Support_System:_Bayesian_Networs
http://genie.sis.pitt.edu/wiki/SMILE:_Probabilistic_Inference_in_Bayesian_Networks
http://genie.sis.pitt.edu/wiki/Appendices:_XDSL_File_Format__XML_Schema_Definitios



Bayesian Network

Edited by Ahmed Rebai

ISBN 978-953-307-124-4
Hard cover, 432 pages
Publisher Sciyo
Published online 18, August, 2010
Published in print edition August, 2010

Bayesian networks are a very general and powerful tool that can be used for a large number of problems involving uncertainty: reasoning, learning, planning and perception. They provide a language that supports efficient algorithms for the automatic construction of expert systems in several different contexts. The range of applications of Bayesian networks currently extends over almost all fields including engineering, biology and medicine, information and communication technologies and finance. This book is a collection of original contributions to the methodology and applications of Bayesian networks. It contains recent developments in the field and illustrates, on a sample of applications, the power of Bayesian networks in dealing the modeling of complex systems. Readers that are not familiar with this tool, but have some technical background, will find in this book all necessary theoretical and practical information on how to use and implement Bayesian networks in their own work. There is no doubt that this book constitutes a valuable resource for engineers, researchers, students and all those who are interested in discovering and experiencing the potential of this major tool of the century.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Nipat Jongsawat, Anunucha Tungkasthan and Wichian Premchaiswadi (2010). Dynamic Data Feed to Bayesian Network Model and SMILE Web Application, Bayesian Network, Ahmed Rebai (Ed.), ISBN: 978-953-307-124-4, InTech, Available from: http://www.intechopen.com/books/bayesian-network/dynamic-data-feed-to-bayesian-network-model-and-smile-web-based-interface

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

© 2010 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution-NonCommercial-ShareAlike-3.0 License, which permits use, distribution and reproduction for non-commercial purposes, provided the original is properly cited and derivative works building on this content are distributed under the same license.