Graphical Decision-Theoretic Models on the Web

Pittaya Poompuang, Anuchar Tungkasthan, Phaisarn Sutheebanjard, Nipat Jongsawat
Siam University
Bangkok, Thailand
pitayapp@cgd.go.th, aimdala_ohm@hotmail.com, mr.phaisarn@gmail.com, nipatj@yahoo.com

Abstract: The principles of decision-analytic decision support, implemented in GeNIe (Graphical Network Interface) and SMILE (Structural Modeling, Inference, and Learning Engine) can be applied in practical decision support systems (DSSs). GeNIe plays the role of a developer's environment and SMILE plays the role of the reasoning engine. A decision support system based on SMILE can be equipped with a dedicated user interface. 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. GeNIe is an outer shell to SMILE. GeNIe is implemented in Visual C++ and draws heavily on the MFC (Microsoft Foundation Classes). GeNIe runs under one of the most popular computing platforms: Windows operating systems so that this makes it not easily portable. GeNIe is platform dependent and runs only on Windows computers. This is one disadvantage of using GeNIe. This paper presents a development environment for building graphical decision-theoretic models working on the website by using an original engine called “SMILE”. Finally, this paper also presents the prototype for using SMILE on the web.

Introduction

Decision analysis is the art and practice of decision theory, an axiomatic theory prescribing how decisions should be made. Decision analysis is based on the premise that humans are reasonably capable of framing a decision problem, listing possible decision options, determining relevant factors, and quantifying uncertainty and preferences, but are rather weak in combining this information into a rational decision.

Decision analysis comes with a set of empirically tested tools for framing decisions, structuring decision problems, quantifying uncertainty and preferences, discovering those factors in a decision model that are critical for the decision, and computing the value of information that reduces uncertainty. Probability theory and decision theory supply tools for combining observations and optimizing decisions. While decision analysis is based on two quantitative theories, probability theory and decision theory, its foundations are qualitative and based on axioms of rational choice. The purpose of decision analysis is to gain insight into a decision and not to obtain a recommendation.

GeNIe (Graphical Network Interface) and SMILE (Structural Modeling, Inference, and Learning Engine) are designed and used as a tool for framing decisions, structuring decision problems, quantifying uncertainty and preferences, discovering those factors in a decision model that are critical for the decision, and computing the value of information that reduces uncertainty.

GeNIe is a development environment for building graphical decision-theoretic models. 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. GeNIe is an outer shell to SMILE. GeNIe allows for building models of any size and complexity, limited only by the capacity of the operating memory of the computer. GeNIe is implemented in Visual C++ and draws heavily on the MFC (Microsoft Foundation Classes). This makes it not easily portable, although it runs under one of the most popular computing platforms: Windows operating systems. GeNIe is platform dependent and runs only on Windows computers.

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, and structural equation models. Its individual functions, defined in SMILE Applications Programmer Interface, allow creating, editing, saving, and loading graphical models, and using them for probabilistic reasoning and decision making under uncertainty. SMILE is implemented in C++ in a platform independent fashion. 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 suits the user of the resulting application most.

One disadvantage and limitation of using GeNIe is platform dependent. It runs only on Windows computers. This paper presents the idea of building and developing graphical decision-theoretic models on the web page in order to overcome such a limitation of GeNIe. On web page, every standard web browser can load decision-theoretic models and user or decision modeler can play with the model and compute the value of information. This paper also presents the prototype for using SMILE -Structural Modeling, Inference, and Learning Engine- on the web.

The Study

Bayesian networks

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 dependences 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 effects of external manipulation.

Nodes of a Bayesian network are usually drawn as circles or ovals. The simple Bayesian network shown in **Figure 1** represents two variables, *Success* and *Forecast,* and expresses the fact that they are directly dependent on each other.

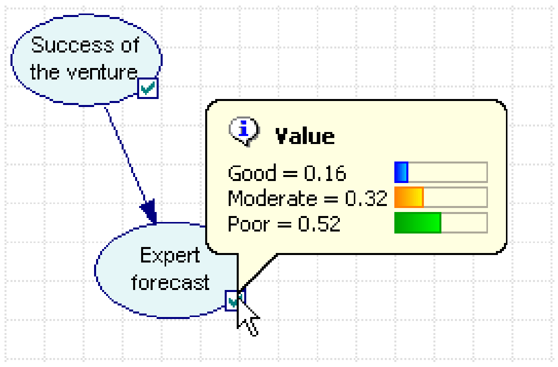

Figure 1 Bayesian Network in GeNIe

Influence diagrams

Influence diagrams (Howard & Matheson 1984), also called relevance diagrams, are acyclic directed graphs representing decision problems. The goal of influence diagrams is to choose a decision alternative that has the highest expected gain (utility).

Similarly to Bayesian networks, influence diagrams are very useful in showing the structure of the domain, i.e., the structure of the decision problem. Influence diagrams contain four types of nodes (Decision, Chance, Deterministic, and Value) and two types of arcs (influences and informational arcs). The influence diagram shown in **Figure 2** models a decision related to an investment in a risky venture.

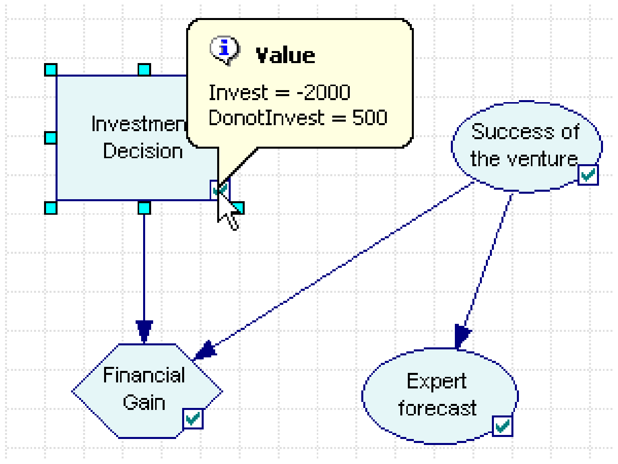

Figure 2 Influence Diagram in GeNIe

The Prototype Design for using SMILE on the web

SMILE API is used to create the graphical probabilistic and decision-theoretic models on the web page that will be helping user and modeler to develop the models more easily without installing any programs on machine. In order to display the decision-theoretic models on the web page, two outer shells and one core named “SMILE” or “SMILE.dll” are required. See **Figure 3**.

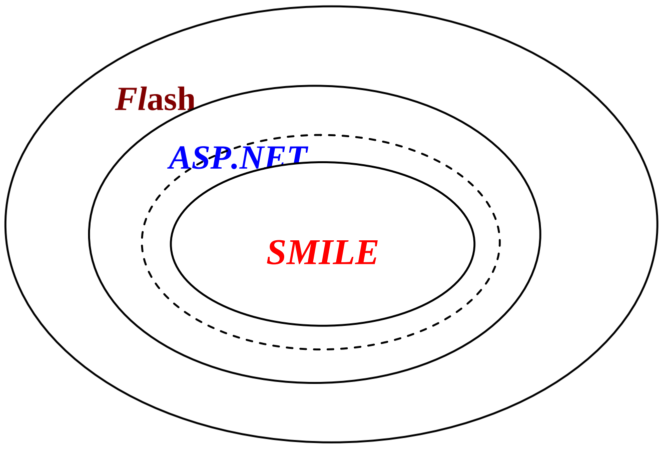

Figure 3 Decision-Theoretic Models Architecture

SMILE is the core level of the designed prototype. SMILE provides the useful functions such as Bayesian networks and influence diagrams for solving probabilistic and decision-theoretic problems. SMILE is released as a dynamic link library (DLL) implementing graphical probabilistic and decision-theoretic models.

SMILE.NET is used as a SMILE wrapper to .NET layer. Actually there are several SMILE wrappers, such as SMILE.NET (.NET interface), SMILEX (Active X), jSMILE (Java interface), etc. This project uses the SMILE.NET as a wrapper in order to interface with the .NET layer.

ASP.NET layer is classified into 2 layers, inner and outer. The inner layer of ASP.NET is developed by Visual Basic.NET to interface with SMILE.net and compile it to ClassSmileNet.dll. The outer layer of ASP.NET is developed by Visual C#.Net to interface between ClassSmileNet.dll and Flash layer.

FLASH layer is a presentation layer for this prototype. The FLASH layer composes of two main components such as FLASH Remoting and FLASH player. Flash Remoting provides a network communications channel between Flash applications and remote services (ASP.NET). FLASH player is responsible for displaying the model on web site. FLASH Remoting provides the following advantages:

• Ease of use Flash Remoting offers automatic data type conversion from native remote service code, such as Java, CFML, and C#, to ActionScript and back again. Also, Flash Remoting MX automatically performs logging, debugging, and security integration.

• Performance Flash Remoting serializes messages between Flash applications and remote services using the Action Message Format (AMF). AMF is a binary format modeled on the Simple Object Access Protocol (SOAP) format.

• Extensibility Flash Remoting is designed to integrate with established application design patterns and best practices to build well-designed Flash applications.

When compared to traditional HTML-based browser applications, Flash applications provide unique abilities to create dynamic and sophisticated user interactions, including the following:

• Flash Player runtime to execute code, transmit data, and invoke remote services

• Separation of client-side presentation logic from the server-side application logic

• Efficient use of bandwidth by removing the need to refresh the entire page and employing vector-based graphics

• Easy deployment on multiple platforms and devices on the server side, Flash Remoting runs as an assembly in .NET servers. Depending on the application server platform, Flash Remoting on the server contains a series of filters that perform logging, error handling, and security authentication, as well as automatically mapping the service function request to the appropriate server technology.

Using Flash Remoting, we can build sophisticated Flash applications, such as a message board, shopping cart, or product catalog. The following figure depicts a simplified representation of the Flash Remoting architecture:

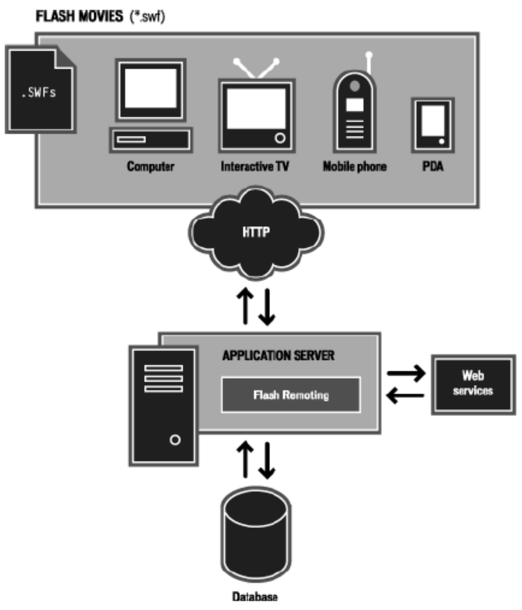

Figure 4 Flash Remoting Architecture

Graphical Decision-Theoretic Model on Web: The Case Study

The graphical decision model, shown in **Figure 5**, presents an Influence Diagrams for investment decision in a risky venture. It is acyclic directed graphs representing decision problem. The goal of influence diagrams is to choose a decision alternative that has the highest expected gain (utility). Influence diagrams contain four types of nodes (Decision, Chance, Deterministic, and Value) and two types of arcs (influences and informational arcs).

Decision nodes, usually drawn as rectangles (such as node Investment Decision below), represents variables that are under control of the decision maker and model the decision alternatives available to the decision maker. Decision nodes include a specification of the available decision options. Node Investment Decision below has two alternatives Invest and DoNotInvest.

Chance nodes, usually drawn as circles or ovals (such as nodes Expert Forecast and Success of the venture below), are random variables and they represent uncertain quantities that are relevant to the decision problem. They are quantified by conditional probability distributions.

Deterministic nodes, usually drawn as double-circles or double-ovals, represent either constant values or values that are algebraically determined from the states of their parents. In other words, if the values of their parents are known, then the value of a deterministic node is also known with certainty. Deterministic nodes are quantified similarly to Chance nodes. The only difference is that their probability tables contain all zeros or ones (note that there is no uncertainty about the outcome of a deterministic node once all its parents are known).

Value nodes, usually drawn as diamonds (such as node Financial Gain above), represent utility, i.e., a measure of desirability of the outcomes of the decision process. They are quantified by the utility of each of the possible combinations of outcomes of the parent nodes.

Normally, **an arc** in an influence diagram denotes an influence, i.e., the fact that the node at the tail of the arc influences the value (or the probability distribution over the possible values) of the node at the head of the arc. Some arcs in influence diagrams have clearly a causal meaning. In particular, a directed path from a decision node to a chance node means that the decision (i.e., a manipulation of the graph) will impact that chance node in the sense of changing its probability distribution.

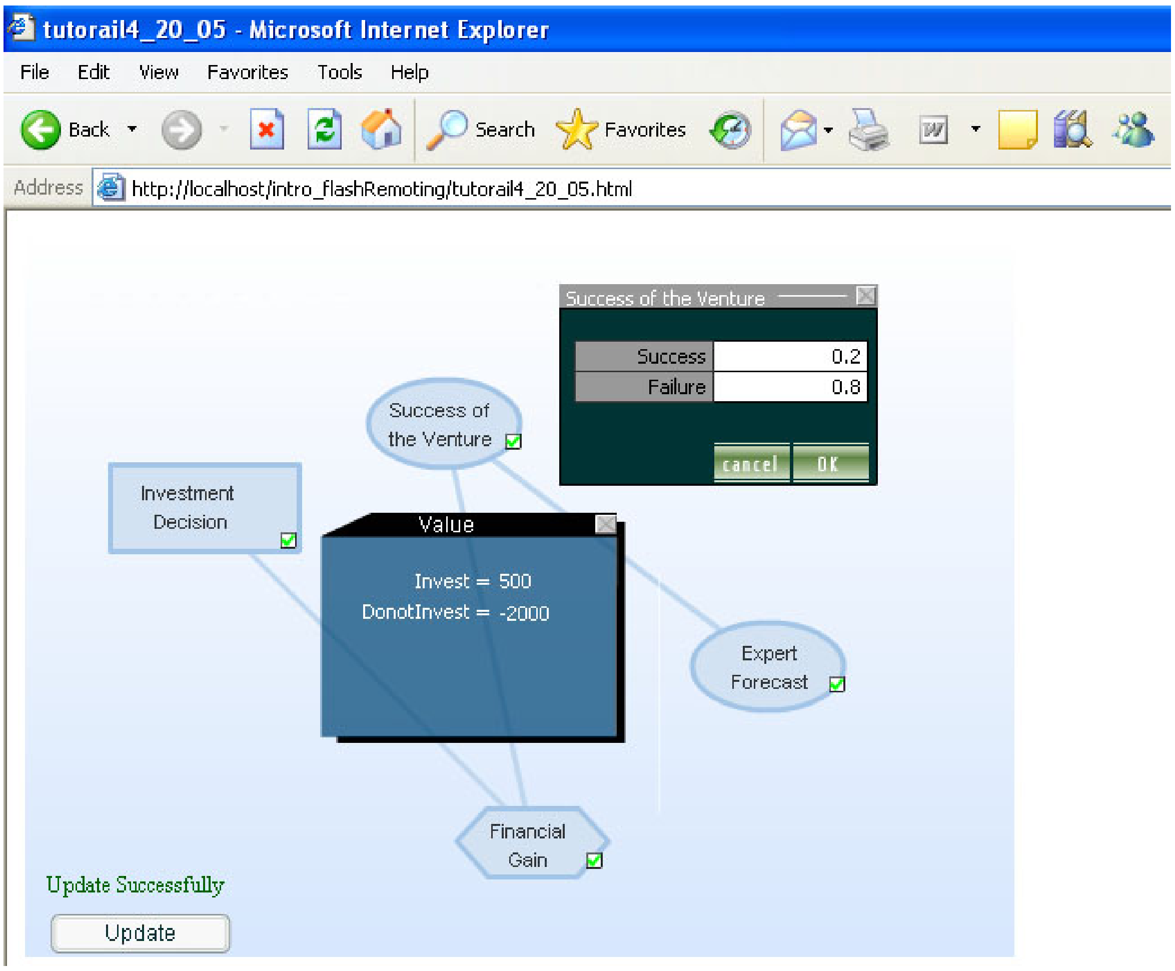

Figure 5 Graphical Decision-Theoretic Model on Web

Future Work

This prototype is a first stage of implementing Graphical Decision-Theoretic Models on the web site. Next stage, we will further develop many Graphical Decision-Theoretic Models on Web Site for specific DSS purpose such insurance, loan, and deposit models for helping decision makers. Lastly, we will develop web user interface for building decision models in general purposes as well as GeNIe working on Windows environment.

Conclusions

Graphical decision-theoretic models on the web is designed and presented in order to compensate the GeNIe (Graphical Network Interface). GeNIe is platform dependent and runs only on Windows computers. Working with GeNIe, modelers still need to install GeNIe program on machine. It is not worldwide. However, GeNIe is still work well on Windows environment. This paper presents the prototype of graphical decision-theoretic models on the web that works as GeNIe do on the Windows. It is a new option for working with graphical decision-theoretic models by using an existing SMILE (Structural Modeling, Inference, and Learning Engine) engine and SMILE.NET wrapper. This new option, building graphical decision-theoretic models on the web, will be a nice way for users, consumers, and modelers working with the models more comfortable and easily. This will be an important part for human prescribing how decisions should be made.

References

Lin, Yan & Marek J. Druzdzel (1997). Computational advantages of relevance reasoning in Bayesian belief networks. In Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence (UAI-97), Morgan Kaufmann: San Mateo, CA, pages 342-350.

Marek J. Druzdzel and Linda C. van der Gaag (2000). *Building probabilistic networks: `Where do the numbers come from?' Guest editors' introduction*. IEEE Transactions on Knowledge and Data Engineering, 12(4):481-486.

Yuan, Changhe & Marek J. Druzdzel. An Importance Sampling Algorithm Based on Evidence Pre-propagation. In Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence (UAI-03), Morgan Kaufmann: San Mateo, CA, pages 624-631.

Cheng, Jian & Marek J. Druzdzel. *AIS-BN: An adaptive importance sampling algorithm for evidential reasoning in large Bayesian networks*. Journal of Artificial Intelligence Research (JAIR), 13:155-188, 2000.

Druzdzel, Marek J. & Henri J. Suermondt (1994). *Relevance in probabilistic models: "backyards" in a "small world."* In Working notes of the AAAI-1994 Fall Symposium Series: Relevance, New Orleans, LA, pages 60-63.

http://genie.sis.pitt.edu/