An MDP Approach for Explanation Generation

Francisco Elizalde[1,3]
[1] Tec de Monterrey, C. Cuernavaca
P. de la Reforma 182-A, Col. L. Cuernavaca
Temixco, Morelos, 62589, México
fef@iie.org.mx

Enrique Sucar[1,2]
[2] INAOE
Luis Enrique Erro No. 1
Tonantzintla, Puebla, 72000, México
esucar@inaoep.mx

Alberto Reyes and Pablo deBuen
[3] Instituto de Investigaciones Eléctricas
Reforma 113, Col. Palmira
Cuernavaca, Morelos, 62490, México
areyes; debuen@iie.org.mx

Abstract

In order to assist a power plant operator to face unusual situations, we have developed an intelligent assistant that explains the suggested commands generated by an MDP-based planning system. This assistant provides the trainee a better understanding of the recommended actions to later generalize them to similar situations. In a first stage, built-in explanations are predefined by a domain expert and encapsulated within explanation units. When the operator takes an incorrect action, an explanation is automatically generated. A controlled user study in this stage showed that explanations have a positive impact on learning. In a second stage, we are developing an automatic explanation generation mechanism based on a factored representation of the decision model used by the planning system. As part of this stage, we describe an algorithm to select a relevant variable, which is a key component of the explanations defined by the expert.

Introduction

Under emergency conditions, a power plant operator has to assimilate a great amount of information to promptly analyze the source of the problem and take the corrective actions. In such situations, in order to define the corrective actions, He has to be able to discriminate between erroneous inputs and the source of the problem. To assist the operator to face these situations, we have developed an intelligent assistant system (IAS) (Elizalde, Sucar, & deBuen 2005). An important requirement for intelligent assistants is to have an explanation generation mechanism, so that the trainee has a better understanding of the recommended actions and can generalize them to similar situations (Herrmann, Kloth, & Feldkamp 1998).

Although there is a lot of work on explanation generation for some representations, such as rules and qualitative models, there is very little work on explanation using probabilistic representations, in particular graphical models. The work on explanations based on graphical models (GM) can be divided according to the classes of models considered, basically Bayesian networks (BN's) and decision networks. BN's (Pearl 1988) graphically represent the dependencies of a set of random variables, and are usually used for estimating the posterior probability of some variables given another. So the main goal of explanations is to try to understand this inference process, and how it propagates through the network. Two main strategies have been proposed for explanation with BN's. One strategy is based on transforming the network to a qualitative representation, and using this more abstract model to explain the relations between variables and the inference process (Druzdzel 1991), (Renooij & Van-DerGaa 1998). The other strategy is based on the graphical representation of the model, using visual attributes (such as colors, line widths, etc.) to explain relations between nodes (variables) as well as the the inference process (Lacave, Atienza, & Diez 2000). The explanation of links represents qualitative influences (Wellman 1990) by coloring the links depending on the kind of influence transmitted from its tail to its head. Another possibility for static explanation consists of explaining the whole network.

Decision networks or influence diagrams extend BN's incorporating decision nodes and utility nodes. The main objective of these models is to help in the decision making process, by obtaining the decisions that maximize the expected utility. So explanation in this case has to do with understanding why some decision (or sequence of decisions) is the optimal one given the current evidence. There is very little work on explanations for decision networks. Bielza (Bielza, del Pozo, & Lucas 2003) proposes an explanation method for medical expert systems based on influence diagrams. It is based on reducing the table of optimal decisions obtained from an influence diagram, building a list that clusters sets of variable instances with the same decision.They propose to use this compact representation of the decision table as a form of explanation, showing the variables that are fixed as a rule for certain case. It seems like a very limited form of explanation, difficult to apply to other domains. Although, an approach based on templates combined with expert knowledge could be seen as a new alternative to solve some limitations in the field.

Markov decision processes can be seen as an extension of decision networks, that consider a series of decisions in time (dynamic decision network). Thus, the work in explanation for Bayesian and decision nets is relevant, although not directly applicable.

The following section describes a Markov decision

Copyright © 2007, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

processes fundamentals. Next we describe the IAS, their components and a brief of the application domain. After this section we describe the Explanation Generation mechanism proposed in two stages: first, as a Built-in Explanations, and; second, as an Automatic Explanation approach. Finally, we give conclusions and future work.

Markov decision processes

A Markov decision process (MDP) (Puterman 1994) models a sequential decision problem, in which a system evolves in time and is controlled by an agent. The system dynamics is governed by a probabilistic transition function that maps states and actions to states. At each time, an agent receives a reward that depends on the current state and the applied action. Thus, the main problem is to find a control strategy or *policy* that maximizes the expected reward over time. The Markov property means that the next state s_{t+1} and the immediate reward r_{t+1} depend only of the current state and the action (s_t,a_t).

A solution to an MDP is a policy that maximizes its expected value. For the discounted infinite-horizon case with any given discount factor β, there is a policy V^* that is optimal regardless of the starting state that satisfies the *Bellman* equation (Bellman 1957):

$$V^*(s) = max_a\{R(s,a) + \gamma\beta_{u\in S}\Phi(a,s,u)V^*(u)\}$$

Two popular methods for solving this equation and finding an optimal policy for an MDP are: (a) value iteration and (2) policy iteration (Puterman 1994).

In value iteration, optimal policies are produced for successively longer finite horizons until they converge. In policy iteration, the current policy is repeatedly improved by finding some action in each state that has a higher value than the action chosen by the current policy for the state. The policy is initially chosen at random, and the process terminates when no improvement can be found. This process converges to an optimal policy.

Factored MDPs

A common problem with the MDP formalism is that the state space grows exponentially with the number of domain variables, and its inference methods grow with the number of actions. Thus, in a large problems, MDPs becomes impractical and inefficient. Factored representations avoid enumerating the problem state space, producing a more concise representation that makes solving more complex problems tractable.

In a factored MDP, the set of states is described via a set of random variables **X** = $\{X_1, ..., X_n\}$, where each X_i takes on values in some finite domain *Dom*(X_i). A state **x** defines a value $x_i \in$ *Dom*(X_i) for each variable X_i. Thus, the set of states *S* = *Dom*(X_i) is exponentially large, making it impractical to represent the transition model explicitly as matrices. Fortunately, the framework of dynamic Bayesian networks (DBN) (Dean & Kanasawa 1989) gives us the tools to describe the transition model concisely. In these representations, the post-action nodes (at the time *t+1*) contain matrices with the probabilities of their values given their parents' values under the effects of an action.

Intelligent Assistant System for Operator's Training

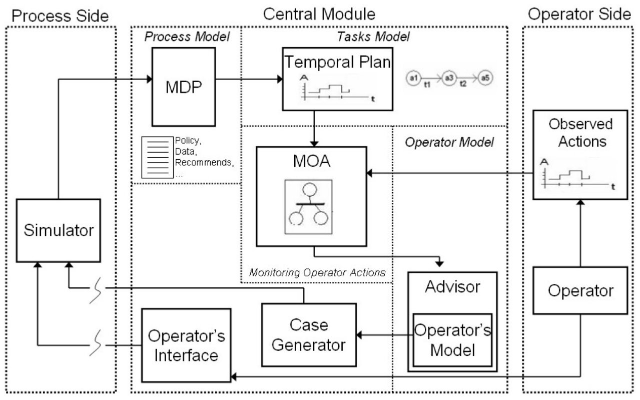


Figure 1: IAOT description (Elizalde, Sucar, & deBuen 2005).

We have developed an Intelligent Assistant for Operator's Training (IAOT), see Fig. 1. The input to the IAOT is a recommended-plan generated by a decision-theoretic planner, which establishes the sequence of actions that will allow to reach the optimal operation of a steam generator (Reyes *et al.* 2006). Operator actions are monitored and discrepancies are detected regarding the operator's expected behavior. This behavior is followed-up using a model (obtained from the optimal plan) represented as a Petri net (Petri 1962). IAOT's architectural components are described as follows. Fig. 1 shows the following three main blocks: *process side*, *central module*, *operator side*.

The simulator that represents the plant is on the process side. The operator side includes the operator (human-machine interface) and the observed actions. The central module has 5 submodules: *process model*, *task model*, *operator model*, *Monitoring Operator Actions* and *interface*. The process model contains the MDP which generates the optimal policy. In the task model, a temporal plan is generated. The operator's model includes the advisor module. The monitor of the operator's actions contains a Petri net model just for the actions sequence.

The process starts with an initial state of the plant, usually under an abnormal condition; so the operator should return the plant to its optimum operating condition using some controls. Considering this situation, an optimal policy is generated based on an MDP, and an a temporal plan is obtained to reach the desired state. This plan has nodes that represent actions whose temporary relationship are represented through arcs. The model into the monitor of the operator's actions is applied in order to follow the sequence of actions from the operator to compared against those actions recommended by the adviser. If an error is detected when the action performed by the operator deviates from the optimal plan, either in the type of action or its timing. Depending on the of error and the operators model (novice, intermediate or advanced), an advice message is generated. These messages give evidence of the incorrect actions completed by the operator or the critical plant conditions, and suggest the correct actions. If the error is not critical the training session continues, otherwise it is terminated.

The assistant detects the differences between the executed and recommended actions, gives advice to the trainee and if necessary stops the current case. Depending on the operator's performance, the advisor presents a new case through the case generator module. This module contains several predefined scenarios for selection, with different complexity levels. Through an operator's interface the trainee interacts with the system. This interface represents the objects, the instruments and the application domain information which the operator uses to complete the task.

Domain

We have considered as a training scenario the operation of the steam generation system in a power plant. In this process, and under certain conditions, the drum level becomes unstable and the operator has to return it to a safe state using the control valves.

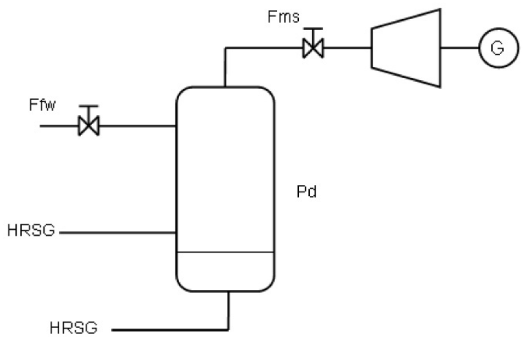


Figure 2: Schematic diagram for the steam section in a combined cycle power plant, indicating the four variables used in the explanation-framework. one of these is selected as a relevant variable for each state-action

Figure 2 depicts the main elements considered in the training scenario: steam drum, turbine, generator and control valves. The main variables in this process are:

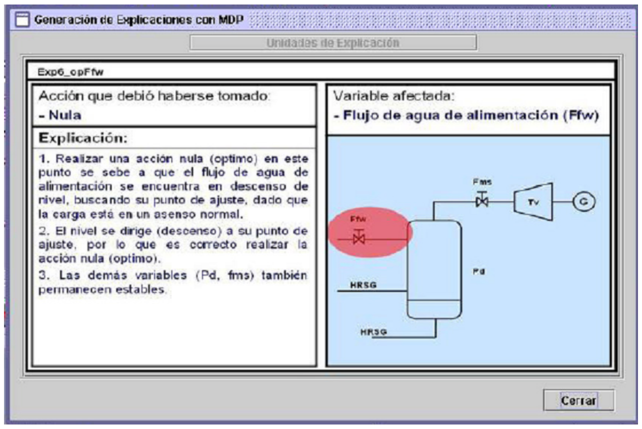
1. Drum pressure (Pd)
2. Main steam flow (Fms)
3. Feed water flow (Ffw)
4. Power Generation (G)

In this domain the operator can command the process opening or closing control valves (Fms, Ffw). The null action is also possible to be selected.

Built-in Explanations

As a first stage towards explanation generation, we started by defining a set of explanation units with the aid of a domain expert, to test their impact on operator training. This explanation units are stored in a data base, and the assistant selects the appropriate one to show to the user, according to the current state and optimal action given by the MDP.

The MDP representation establishes the optimal action given a plant state (optimal policy). Explanations are extracting from the MDP optimal policy by considering: a) the optimal action for the current state; b) the relevant variable (Var_{Rel}) for each action (determined by the expert); and c) the most adequate explanation that justifies the action. These explanations are defined by an expert and are encapsulated in explanation units ($Unid_{Exp}$), an example of an ($Unid_{Exp}$) is show in Fig. 3.



Exp6_opFfw

Acción que debió haberse tomado:
- Nula

Explicación:
1. Realizar una acción nula (optimo) en este punto se sebe a que el flujo de agua de alimentación se encuentra en descenso de nivel, buscando su punto de ajuste, dado que la carga está en un asenso normal.
2. El nivel se dirige (descenso) a su punto de ajuste, por lo que es correcto realizar la acción nula (optimo).
3. Las demás variables (Pd, fms) también permanecen estables.

Variable afectada:
- Flujo de agua de alimentación (Ffw)

Figure 3: An Explanation Unit generated for a trainee as a consequence of an action taken during a real training session.

The built-in $Unid_{Exp}$ depicted in Fig. 3 was generated by the IAOT during a training session. An $Unid_{Exp}$ has two main components, the first one (left side) is an explanation and the second one (right side) is the selected relevant variable (Var_{Rel}). Both sides (left and right) are the built-in explanation for a certain incomplete task due an incorrect action taken by the trainee. The two components in an $Unid_{Exp}$ are directly related sice the Var_{Rel} is an important component to fit the explanation in left side, for example, for the $Var_{Rel} = Ffw$ in the right side, the explanation in the left side of the Fig. 3 show the text: "*Acción que debió haberse tomado: Nula*", means: "Action that must be taken: Null" due the Var_{Rel}, and the text: "*Explicación - 1. Realizar una acción nula (optimo) ...*", refers to: "Explanation - 1. Taking the action Null (optimal) at this point is given that the feedback water flow level is descending to the set point due the normal ascending of the load", and the text "*2. El nivel se dirige (descenso) a su punto de ajuste, por lo que es correcto realizar la accin nula (optimo)*", means: "2. The level is towards (descending) the set point, therefore is correct to take the Null action (optimal)".

Then, from the optimal policy, the two main components to generate a complete $Unid_{Exp}$ are: 1. An optimal action given the state, and; 2. A relevant variable (Var_{Rel}) given the action-state. Therefore, the first component of the $Unid_{Exp}$ is an explanation of why the the specific action is selected given the current plan state. In the current implementation, a set of templates were defined for typical plant conditions with the aide of an expert in the domain. These templates are stored in a data base. Based on the plant state and the optimal action, the assistant extracts from the data base the most adequate template, which is presented to the user.

The second component (Var_{Rel}) is more critical under the current situation and which is modified by the correct action. A simplified representation of the state space includes 2 variables or dimensions and the expected value of some of the states obtained by solving the MDP. For the state in the variables an optimal action corresponds as an arrow pointing to next state. For instance, the (Var_{Rel}) is V_1, which is the variable in the Y axis in the state space, modified by the optimal action.

The Built-in explanation method starts with a representation of the optimal policy as a temporal evolution of the main variables of the process. An example of evolution of some of the variables and the optimal actions are shown in Fig. 4. In this case there are 4 main variables, G, Pd, Fms, Ffw, where G = Generation; Pd = Dome pressure; Fms = Main steam flow, and; Ffw = Flow water feedback. The variable A depicts the actions executed at different times. Therefore, an expert domain analyze and selects from this policy representation the source of the explanation. For example, for the first action that is highlighted in the figure, the Var_{Rel} is Pd which has reached a maximum value, and it starts to decrease as a result of this action and the associated $Unid_{Exp}$ will explain this to the operator.

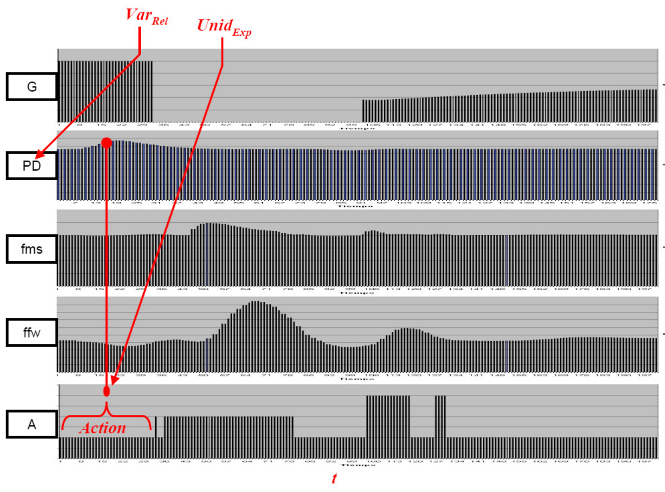
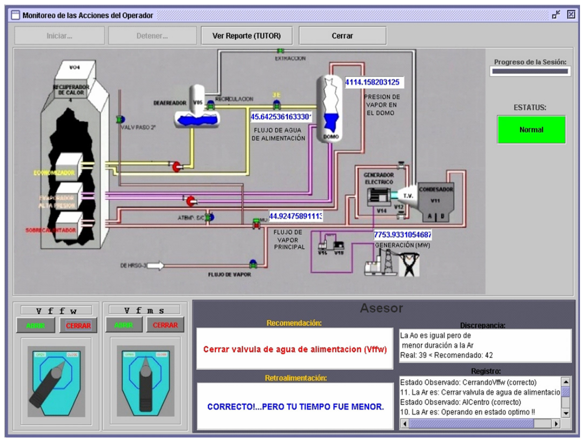


Figure 4: Temporal evolution of the 4 main variables and the action. The relevant variable, Pd for the state-action at the start of the interval is highlighted.

In summary, we have designed as a first stage an explanation mechanism based on an optimal policy derived from an MDP, and explanation units defined by a domain expert. Using this mechanism, whenever the IAOT detects an error by the user, it gives him advice by displaying the correct actions and the associated explanation. In the next section we show empirical evidence that these explanations help in the learning process.

Preliminary Results

To evaluate the effect of the explanations on learning, we performed a controlled experiment with 10 potential users with different levels of experience in power plant operation. The set of participants was divided in two groups: 1. Test group (G1): it uses the IAOT with an explanation mode, and has five participants in a three-level profile (novice, intermediate and advanced); 2. Control group (G2): it uses the IAOT without explanations, only advice. Group G2 has a five participants with the same three-level profile.



4114.158203125 PRESION DE VAPOR EN EL DOMO

45.642536163330 FLUJO DE AGUA DE ALIMENTACIÓN

44.92475891113 FLUJO DE VAPOR PRINCIPAL

7753.9331054687 GENERACIÓN (MW)

Cerrar valvula de agua de alimentacion (Vffw)

CORRECTO!...PERO TU TIEMPO FUE MENOR.

La Ao es igual pero de
menor duración a la Ar
Real: 39 < Recomendado: 42

Estado Observado: CerrandoVffw (correcto)
11. La Ar es: Cerrar valvula de agua de alimentacio
Estado Observado: AlCentro (correcto)
10. La Ar es: Operando en estado optimo !!

Figure 5: User interface. In the background it is the simulator, controls and advisor.

Each participant has to control the plant to reach the optimal state under an emergency condition using a simulator and with the aid of the IAOT. Fig. 5 shows the system interface, including an $Unid_{Exp}$. A record is kept for each session, including if the user reached the goal or how much did he achieved. During each session, the suggested actions and detected errors are given to the user, and for G1, also an explanation.

A case was presented to each user according to his level, and was given up to 5 opportunities to reach the desired state with the help of the IAOT (G1 with explanations Fig. 6 left side and G2 without explanation Fig. 6 right side). Then, without the aide of the assistant, the user has to operate the plant (simulator) under a similar situation. Again each participant was given 5 opportunities. For each user in both groups we obtained the percentage of task completion achieved in each opportunity. We adjusted a line to each group to show the tendencies. Fig. 6 summarizes the results, showing a point corresponding to the percentage of task completion for each participant's opportunity (y axis) and the number of opportunities (x axis). A line (obtained with minimum squares) depicts the general tendency of each group. There is a difference between both groups, with a better tendency for the group with explanations (left) to reach faster the proposed goals during a training session.

We consider that these results give evidence that explanations help in the learning of skills such as those required to operate an industrial plant. The hypothesis is that explanations provide a deeper understanding of the process, so the advice can be generalized to similar situations. Of course this needs to be validated with further experiments.

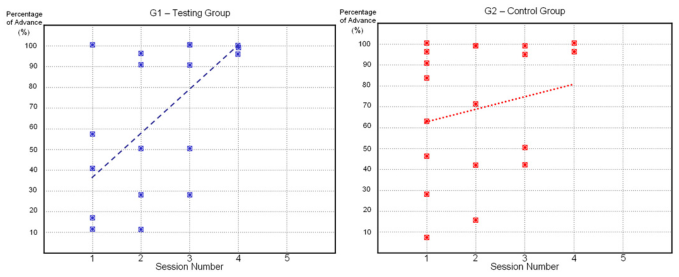


Figure 6: Graphs comparing the performance of both groups. Left: test group (G1). Right: control group (G2).

Automatic explanations generation

Although manually generated explanations are useful, these are difficult to obtain, and to generalize to other plant subsystems or other domains. So we are interested in generating them automatically. For this, we consider at least 3 sources of knowledge: (i) a factored representation of the transition models in the MDP, (ii) an structured representation of the optimal policies and value functions, and (iii) domain knowledge stored in a knowledge base. Additionally, we have defined a set of general explanation templates based on the explanation units provides by the experts. Thus, our proposal for an automatic explanation mechanism, consists on extracting the relevant aspects from the knowledge sources to fill-in an explanation template. This process is illustrated in Fig. 7.

From the expert's explanations, we observe that an important aspect on all explanation units is a "relevant variable". That is the state variable that is most critical under the current situation, and which is usually affected by the optimal action. So, as a first step towards explanation generation, we developed a method to identify the relevant variable.

Relevant variable selection

The algorithm for determining the relevant variable considers two phases: (i) variable reduction, and (ii) variable ranking. For this we consider a structured representations of the

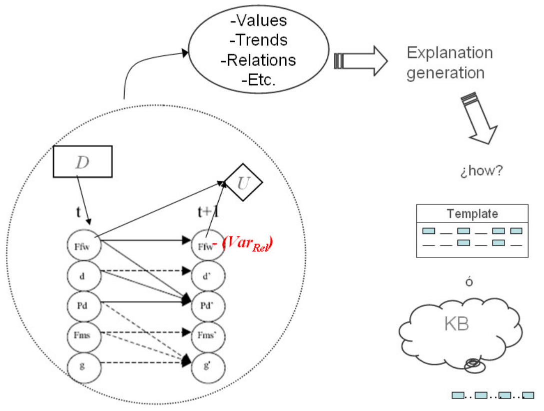


Figure 7: Proposal for an automatic explanations generation from a MDP factored representation.

transition function and the value functions for the MDP. The transition functions are represented as two-stage dynamic Bayesian networks (DBN) (Dearden & Boutillier 1997), one per action; see Fig. 8. The value function (as an example of a reward function structure) is represented as a decision tree (DT) (Reyes, Ibarguengoytia, & Sucar 2004), see Fig. 9.

In the first phase, the variables that are not relevant to the optimal action for the current state are eliminated using the following algorithm.

Variable reduction algorithm

Given the DBN for the transition function under action "*a*", and the DT for the value function:

1. Eliminate all the variables that are non-connected (isolated) nodes in the DBN.
2. Eliminate all the variables that depend only on the same variable in the DBN (i.e., Xt+1 depends only on Xt), and that do not change significantly (the conditional probability table is close to an identity matrix).
3. Eliminate all the variables that do not appear in the DT for the value function.

If after this stage, there remains more than one variable, we consider the second phase, variable ranking. To rank the variables, two aspects are considered: (a) the qualitative change of the variable in the transition model, and (b) the impact of the variable in the utility function.

To evaluate the qualitative change, we instantiate the variables at time *t* in the DBN, and via probability propagation, we estimate the values at time *t+1*. Then we obtain the difference for each variable ($\delta\ X = Xt{+}1 - Xt$), and qualify this into a set of qualitative changes (for ex., no change, a small change or a large change), according to a percentage of the nominal values.

To evaluate the impact on the utility function, we use as a heuristic the distance of the variable to the root of the tree. That is, variables that are in upper levels of the utility tree are more important.

Finally, we combined the two measures (using a weighted linear combination) to obtain a single measure of relevance

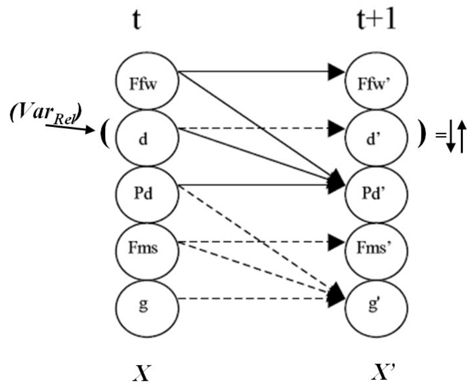


Figure 8: An example of a DBN representation of the transition function for an action.

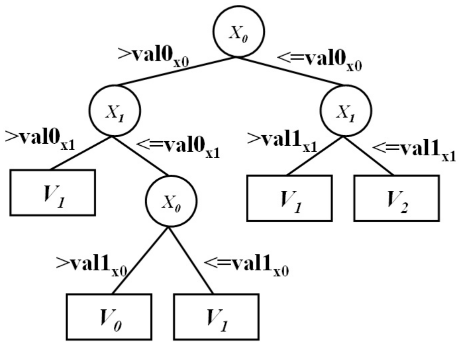


Figure 9: An example of a value function represented as a decision tree.

of a variable, for the current state and its optimal action. The top variable according to this ranking will be considered the relevant variable for the explanation.

We are currently evaluating this approach by comparing the relevant variable selected by our method to the one proposed by the expert for each explanation unit.

Conclusions and future work

In this work a new MDP-based approach for an automatic generation mechanism based on a structured representation of the transition and utility functions is presented. We described an algorithm to select the relevant variable, which is a key component of the explanations defined by the expert. Adequate explanations are selected based on the optimal policy. A catalog of explanation units is defined for each domain. Each explanation unit contains an explanations list for the action-state and a relevant variable. A controlled user study shows that explanations have a positive impact on learning. The results show a better performance for the users with explanations with respect to those without.

As a future work, we are developing an Explanation Model to complement the automatic mechanism. The built-in explanations provided by the expert will be compared with those generated by the automatic mechanism. The use of a factored representation is presented as a proposal.

References

Bellman, R. 1957. *Dynamic Programming*. Princeton, N.J.: Princeton U. Press.

Bielza, C.; del Pozo, J. F.; and Lucas, P. 2003. Optimal decision explanation by extracting regularity patterns. In Coenen, F.; Preece, A.; and Macintosh, A., eds., *Research and Development in Intelligent Systems XX*, 283–294. Springer-Verlag.

Dean, T., and Kanasawa, K. 1989. A model for reasoning about persistence and causation. *Computational Intelligence* 5:142–150.

Dearden, R., and Boutillier, R. 1997. Abstraction and approximate decision-theoretic planning. *Artificial Intelligence* 89:219–283.

Druzdzel, M. 1991. Explanation in probabilistic systems: is it feasible? is it work? In *Intelligent information systems V, Proceedings of the workshop*, 12–24.

Elizalde, F.; Sucar, E.; and deBuen, P. 2005. A prototype of an intelligent assistant for operator's training. In *International Colloquium for the Power Industry*. México: CIGRE-D2.

Herrmann, J.; Kloth, M.; and Feldkamp, F. 1998. The role of explanation in an intelligent assistant system. In *Artificial Intelligence in Engineering*, volume 12, 107–126. Elsevier Science Limited.

Lacave, C.; Atienza, R.; and Diez, F. 2000. Graphical explanations in bayesian networks. In *Lecture Notes in Computer Science*, volume 1933, 122–129. Springer-Veralg.

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

Petri, C. 1962. *Kommunikation mit Automaten*. Phd. thesis, Faculty of Mathematics and Physics at the Technische Universitt, Darmstadt, Germany.

Puterman, M. 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. New York: Wiley.

Renooij, S., and Van-DerGaa, L. 1998. Decision making in qualitative influence diagrams. In *Proceedings of the Eleventh International FLAIRS Conference*, 410–414. Menlo Park, California: AAAI Press.

Reyes, A.; Sucar, L. E.; Morales, E.; and Ibarguengoytia, P. H. 2006. Solving hybrid markov decision proceses. In *MICAI 2006: Advances in Artificial Intelligence*. Apizaco, Mexico: Springer-Verlag.

Reyes, A.; Ibarguengoytia, P. H.; and Sucar, L. E. 2004. Power plant operator assistant: An industrial application of factored mdps. In Monroy, R.; G, A.; Sucar, L.; and Sossa, H., eds., *MICAI 2004: Advances in Artificial Intelligence*, 565–573. Mexico, City: Springer-Verlag.

Wellman, M. 1990. Graphical inference in qualitative probabilistic networks. *Networks: an international journal* 20:687–701.