Integrating Probabilistic and Knowledge-Based Systems for Explanation Generation

Francisco Elizalde[1,2], Enrique Sucar[1,3], Julieta Noguez[4] and Alberto Reyes[2]

[1] Tec de Monterrey, Campus Cuernavaca, Autopista del Sol km 104, Colonia Real del Puente, Xochitepec, Morelos, 62790, México.
[2] Instituto de Investigaciones Eléctricas, Reforma 113, Col. Palmira, Cuernavaca, Morelos, 62490, México.
[3] Instituto Nacional de Astrofísica Optica y Electrónica, Luis Enrique Erro No. 1, Tonantzintla, Puebla, 72000, México.
[4] Tec de Monterrey, Campus Cd. de México, Calle del Puente 222, Ejidos de Huipulco, 14380, México, D.F.

Abstract. 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. In this work we combine different knowledge sources to generate explanations for operator training. The explanations are based on a general template which is composed of 3 main parts: (i) the recommended action in the current situation; (ii) a graphical representation of the process highlighting the *relevant variable*; (iii) a verbal explanation. The optimal action is obtained from a Markov decision process (MDP) that guides the operator in the training session. For determining the relevant variable we developed a method that estimates the impact of each variable on the utility, selecting the one with highest impact in the current state. The verbal explanation is extracted from a domain knowledge base that includes the main components, actions and variables in the process, represented as a frame system. We present preliminary results of explanations generated in the power plant domain.

1 Introduction

An important requirement for intelligent trainers 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 [1]. We are particularly interested in training power plant operators. 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. Novice operators might not have enough experience to take the best action, and experienced operators might forget how to deal with emergency situations, as these occur sporadically in current plants. So in both cases, a simulator coupled with an intelligent assistant can help to train the operators so they can react appropriately when an emergency situation arises.

We have developed an Intelligent Assistant for Operator's Training (IAOT) [2]. 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 [3]. Operator actions are monitored and discrepancies are detected regarding the operator's expected behavior. If an error is detected, the assistant gives advice to the trainee, and might interrupt the session. In previous work [4] we used a set of pre-defined explanations obtained from a domain expert, and these were given to the user according to the current situation. A controlled user study showed that operators trained with the explanation mechanism have a better performance in similar situations [4]. But obtaining the explanations from an expert is a complex and time-consuming process, so it is desirable that the assistant can generate the explanations automatically.

We are developing an automatic generation explanation mechanism. For this, we analyzed the explanations given by the domain expert, and designed a general template. This template is composed of 3 main parts: (i) the recommended action and the relevant variable in the current situation; (ii) a graphical representation of the process highlighting the relevant variable, and (iii) a verbal explanation. To generate the information required to fill the template we combine several knowledge sources. The optimal action is obtained from a Markov decision process (MDP) that guides the operator in the training session. The relevant variable, which is a key element in the explanation, is obtained from the MDP by analyzing which variable has the biggest impact on the utility given the current state [5]. The graphical part is generated from a general block diagram of the process, where the relevant variable is highlighted. The verbal explanation is obtained from a domain knowledge–base (KB) that includes the main components, actions and variables in the process, represented as a frame system. To extract the important information from the KB, we use the information from the MDP (state, action and relevant variables) as pointers to the relevant components, actions and variables; and then follow the links in the frame system to extract other relevant information. The information in the explanation template depends on the level of the operator: novice, intermediate or expert; which is obtained from a simple student model incorporated in the IAOT.

In this paper we describe the explanation generation mechanism and present some preliminary results in the power plant domain. In particular, we show that the relevant variable obtained automatically agrees with the relevant variable given by the expert in a sample of cases. We also present some initial templates generated by our system.

2 Fundamentals

2.1 Factored Markov Decision Processes

A Markov decision process (MDP) [6] 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 Φ that maps states $\mathbf{S}$

and actions **A** to new states **S'**. At each time, an agent receives a reward R that depends on the current state s and the applied action a. Thus, the main problem is to find a control strategy or *policy* π that maximizes the expected reward V over time. For the discounted infinite-horizon case with any given discount factor γ, there is a policy π^* that is optimal regardless of the starting state and that satisfies the *Bellman* equation [7]:

$$V^{\pi}(s) = max_a\{R(s,a) + \gamma \sum_{s \in \mathbf{S}} \Phi(a,s,s')V^{\pi}(s')\} \quad (1)$$

Two methods for solving this equation and finding an optimal policy for an MDP are: (a) dynamic programming and (b) linear programming [6].

In a factored MDP, the set of states is described via a set of random variables $\mathbf{S} = \{X_1, ..., X_n\}$, where each X_i takes on values in some finite domain $Dom(X_i)$. A state $\mathbf{x}$ defines a value $x_i \in Dom(X_i)$ for each variable X_i. Thus, when the set of states $\mathbf{S} = Dom(X_i)$ is exponentially large, it results impractical to represent the transition model explicitly as matrices. Fortunately, the framework of dynamic Bayesian networks (DBN) gives us the tools to describe the transition model concisely. In these representations, the post-action nodes (at the time *t+1*) contain smaller matrices with the probabilities of their values given their parents' values under the effects of an action. For a more detailed description of factored MDPs see [8].

2.2 Frame Systems

The frame based approach [9] is a popular scheme for knowledge representation which has evolved over the years. This approach is particularly useful when not only a data structure is desirable, but when certain intelligence could be added to it. In general, a frame system can represent stereotyped situations through different levels of abstraction. Higher levels are fixed and are used to represent things that usually are true. Lower levels have many terminals that can be instantiated with specific data. A frame has multiple slots used to define the various attributes of an object. The slots can have multiple facets for holding the value for the attributes, defaults, or procedures which are called to calculate a value. The various frames are linked together in a hierarchy with a-kind-of (ako) links that allow inheritance. There can also be defaults for attributes which might be overwritten for specific frames. Another feature of a frame based system is demons. These are procedures which are activated by various updating procedures.

3 Intelligent Assistant for Explanation Generation

We have developed an Intelligent Assistant for Operator's Training (IAOT), see Figure 1. The input to the IAOT is a recommended-plan generated by a decision-theoretic planner, which establishes the sequence of actions that are needed to reach the optimal operation of a power generator [3]. Operator actions

are monitored and discrepancies are detected regarding the operator's expected behavior.

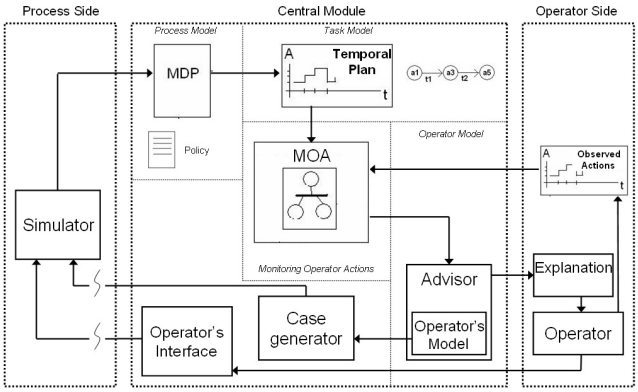

Fig. 1. IAOT block diagram [2]. It consists of 3 main parts: process side (simulator), operator side and the central module. Based on the optimal policy obtained from the MDP a temporal plan is generated. The operator actions are compared to the plan (MOA) and according to this the Advisor generates explanations.

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 and an optimal policy obtained from an MDP, a temporal plan is generated to reach the desired state. 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 type of error and the operator's model (novice, intermediate or advanced), a *Built-in explanation* is generated. These explanations, developed by a domain expert, tell the operator the correct action and why, based on a variable that is the most critical under the current situation (relevant variable). If the error is not critical the training session continues, otherwise it is terminated.

3.1 Domain

We considered a training scenario based on a simulator of a combined cycle power plant, centered in the drum (a water tank) and the related control valves. Under certain conditions, the drum level becomes unstable and the operator has to return it to a safe state using the control valves.

The state variables considered in this scenario are: (i) Drum pressure (Pd); (ii) Main steam flow (Fms); (iii) Feed water flow (Ffw); and (iv) Generation (G). A schematic of the process showing these variables is depicted in Figure 2 (right side).

3.2 Pre-defined Explanations

We initially defined a set of explanation units with the aid of a domain expert, to test their impact on operator's 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. These explanations defined by an expert are encapsulated in explanation units ($Unid_{Exp}$), an example is shown in Figure 2. Each unit has three main components: (i) the recommended action (upper left side), (ii) a verbal explanation of why this is the best action (lower left), and (iii) the relevant variable (V_R) highlighted in a schematic diagram of the process (right side). In this example the relevant variable is *generation*, $V_R = G$, as the absence of generation is the main reason to close the feed–water valve.

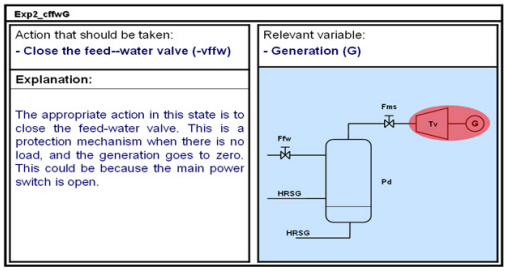

Fig. 2. An example of an explanation defined by a domain expert.

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 users were divided into two groups: G1 with explanations; and G2 without explanations. 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. During each session, the suggested actions and detected errors are given to the user, and for G1, also an explanation. After some training sessions with the aid of the IAOT, the users were presented similar situations without the aid of the assistant. An analysis of the results [2] shows a significant difference in favor of the group with explanations. These results give evidence that explanations help in the learning of skills such as those required to operate an industrial plant. However, obtaining the explanations from an expert is a complex and time–consuming process, so we want to generate the explanations automatically.

4 Automatic explanation generation

The explanation generation mechanism is based on the explanations provided by the domain expert. To build the explanations we combine several knowledge sources: (i) the MDP that represents the process and defines the optimal actions; (ii) a domain knowledge base; (iii) a set of templates; and (iv) an operator model.

The explanations generator module consists of three main stages (see Figure 3). As mentioned before, the relevant variable is a key factor to build an explanation. Therefore the first stage obtains the relevant variable and additional elements as the current state S_i and the optimal action a^*. In the second stage, according to the operator model (novice, intermediate or advanced), a template is selected. In the last stage, the template is filled with information from a domain knowledge base. Each of these stages is detailed in the following sections.

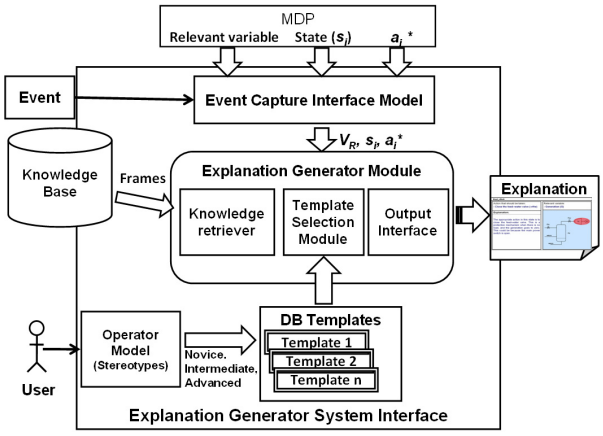

Fig. 3. Block diagram of the explanation generation system. When the trainee makes an error (event), the current state, optimal action and relevant variable are obtained from the MDP. These are used as pointers to obtain the relevant elements from the KB to fill–in an explanation template. The type of template is selected according to the user's model.

4.1 Explanation Template

The explanations are based on a predefined structure or *template*, inspired by the explanations obtained from the domain experts. Each template has a predefined structure and is configured to receive additional information from different sources. There are three types of templates according to the user: novice, intermediate and advanced. The main difference is in the amount and depth of

the information provided: for novice users it is more detailed while for advance operators a more concise explanation is given.

The explanation template has three main components, which are basically the same as those in the experts' explanations (see figure 2):

1. A schematic diagram of the process, highlighting the relevant variable.
2. The optimal action given the current state.
3. A description in natural language of the main reasons for the previous action, which depends on the user level.

The schematic diagram is previously defined for the process (the goal is to find the most relevant variable for certain state and action). Then the optimal action is directly obtained from the MDP given the current state of the plant, and a knowledge base on the domain is used to generate automatic explanations based on their variables, components, actions, and relationships.

4.2 Relevant Variable Selection

A key element for generating the explanations is to find the most relevant variable V_R for certain state s and action a. All the explanations that we obtained from the experts are based on a variable which is considered the most important under the current situation and according to the optimal policy. Intuitively, we can think that the relevant variable is the one with greatest effect on the expected utility, given the current state and the optimal policy. So as approximation to estimate the impact of each factor $X(i)$ in the utility, we estimate the value function U for all the values of $X(i)$, $X(i)_1$, ..., $X(i)_m$; and compare each value to the one for the current state, $X(i)_j$, obtaining the maximum of these differences. The process is repeated for all the variables, and the variable with the highest average difference in value is selected as a relevant variable. However, for some states, the maximum difference is the same for more than one variable (there is a tie), so the algorithm for automatically determining the selection of the relevant variable considers two stages:

Stage 1–Utility Variation: Analyzes the changes in utility when we vary the value of a variable.

$$relevance_s^1(X) = \max_{s' \in neigh_X^1(s)} V(s') - \min_{s' \in neigh_X(s)} V(s') \tag{2}$$

where $neigh_X^1(s)$ is the set of states that have the same values as in s in all the variables, except at most in a variable different from X.

Stage 2–Changes in the Optimal Action: Explore the optimal policy to detect changes in the optimal action for the state.

$$relevance_s^2(X) = \#s' : s' \in neigh_X^2(s) \wedge \pi^*(s) \neq \pi^*(s') \tag{3}$$

where $neigh^2_X(s)$ is the set of states that take the same values than s in all the variables except at most in variable X, and π^* is the optimal policy of the MDP. If we have a state s, then when varying a variable X we find that the optimal policy is the same than for s. We assing a value 0 for $relevance^2_s(X)$, indicating therefore that the presence of its value in the state has not affected the optimal action.

4.3 Knowledge Base

To complement the information obtained from the MDP model, additional domain knowledge is required about relevant concepts, components and actions in the process. Frames provide a natural way for representing the relevant elements and their relations, to fill–in the explanations templates.

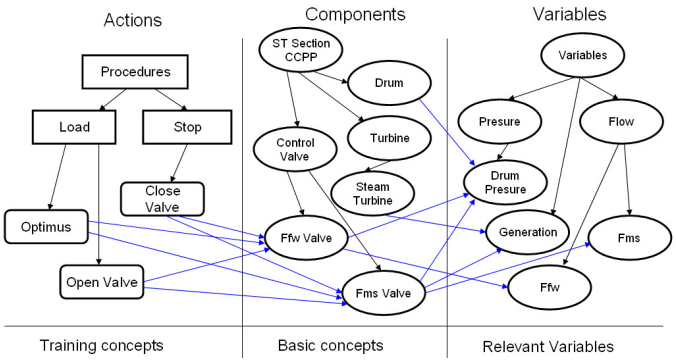

Fig. 4. The KB, represented as a frame hierarchy, is divided in 3 parts: (i) actions, (ii) components, and (iii) variables.

Figure 4 depicts in the KB, that the frames store the basic knowledge about the domain components, variables, actions, and their relationships. This representation is an extension of the one proposed in [10]. The KB includes three hierarchies: (i) procedures and actions; (ii) components; and (iii) variables. It also includes relationships between the frames in different hierarchies: which actions affect each component and the variables associated to each component.

Part of the component hierarchy is depicted in Figure 5. It shows the frame for a *control valve*, and two subtypes, the feed water valve (Ffw) and the main steam valve (Fms). A set of attributes is defined for each valve, with general aspects for the general valve and more specific ones for the others.

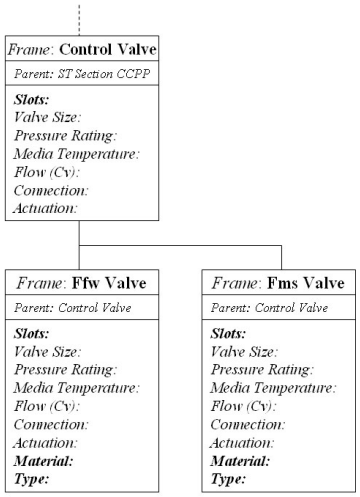

Fig. 5. A partial view of the component hierarchy. It shows the frame for a *control valve* and two subtypes: Ffw and Fms valves.

4.4 Filling the Template

As mentioned before, a template is selected according to the user level. The optimal action and relevant variable are deduced from the MDP model of the process and the policy, and together with the process schematic diagram are inserted in the corresponding template. The missing element is the textual explanation of *why* the action should be selected in the current situation. The explanation mechanism must then determine *what* should be included and *how much* detail to give to the user. *What* is determined by the template structure and *how much* by the operator level.

Inspired by the expert's explanations, we define the following elements for the textual explanation structure:

1. Optimal action, including information on what is the purpose of the action obtained from the corresponding frame
2. Relevant variable, with information of the relevance of this variable and its relation to a certain component.
3. Component, including its main characteristics and its relation to the action.

The elements obtained from the MDP (S, a^*, V_R), are used as pointers to the corresponding frames in the KB from where the basic elements of the textual explanation are extracted. This textual explanation is extended by following the links in the KB hierarchies, adding information from frames in the upper part of the hierarchy (see Figure 6). For instance, if the important component is the feed water valve, it includes additional information from the control valve frame for intermediate users, and more general knowledge on valves for the novice user. In this way, the amount of detail in the explanation is adapted to the user level.

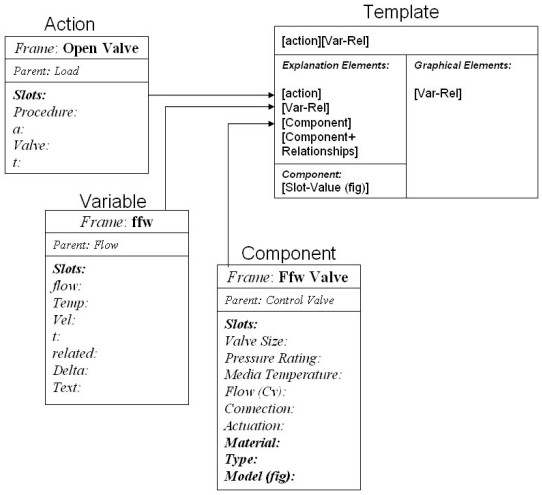

Fig. 6. Textual element of the explanation template is filled by the the relevant variable, action and component frames and adjusted according to the user level.

5 Preliminary Results

We have implemented and evaluated the explanation generation mechanism in the power plant domain. First we evaluate the relevant variable selection, and then we present preliminary results on template generation.

5.1 Relevant Variable Selection

We compare the relevant variable obtained by our method with the one given by the expert for 10 different cases. Results are summarized in Table 7. Although in some cases there was a tie among several variables in the first stage, the second stage broke these ties and we recover the same variable as the one given by the domain expert in all cases. These are very promising results, as the method is giving the expected V_R, which is the basis for producing automatic explanations.

5.2 Template Generation

Based on the explanation generation mechanism described above, we have generated some templates for different conditions and user levels.

Test	selected S					Experimental results		Relevant Variable:		
	Var					Model analysis & selection		Selected by Model	Selected by Expert	Obs
	fms	ffw	d	pd	g	1st ($max\ \Delta U$)	2nd (Π^* change			
1	0	0	0	0	1	pd = 3169.75 ffw = 3169.75 d = 3169.75	ffw	**ffw**	**ffw**	OK
2	0	0	1	3	1	pd = 3129.68 ffw = 3079.68 d = 3079.68	pd	**pd**	**pd**	OK
3	1	1	0	4	0	pd = 2413.56 d = 2413.56 g = 2413.56	d ≈ g	**g**	**g**	OK
4	2	0	0	7	1	fms = 5394.74 ffw = 5394.74 d = 5394.74	fms	**fms**	**fms**	OK
5	3	0	0	0	1	fms = 3781.33 ffw = 3781.33 d = 3781.33	fms	**fms**	**fms**	OK
6	3	1	0	2	1	fms = 3607.53 ffw = 3607.53 d = 3607.53	ffw	**ffw**	**ffw**	OK
7	4	0	1	0	1	fms = 3198.39 ffw = 3198.39 d = 3198.39	fms	**fms**	**fms**	OK
8	4	1	1	7	0	d = 3597.44 g = 3597.44 fms = 3385.63	d ≈ g	**g**	**g**	OK
9	5	1	0	1	1	pd = 5317.48 ffw = 5317.48 d = 5317.48	pd	**pd**	**pd**	OK
10	5	1	1	1	1	pd = 5329.85 ffw = 5329.85 d = 5329.85	pd	**pd**	**pd**	OK

Fig. 7. This table summarizes the 10 cases in which we compared the relevant variable (V_R) selected based on our method against those given by the expert. For each case we show the state, the V_R after each stage, and the V_R according to the expert.

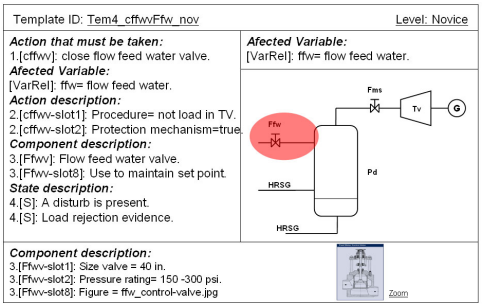

Fig. 8. An example of an explanation template generated.

Figure 8 depicts an example of an explanation template generated. It contains on the left side: (i) the optimal action, (ii) why is important to do this action, (iii) which component is related with optimal action, and (iv) a description of the current state. In the right side the V_R is highlighted in an schematic diagram of the process. The evaluation process will include a focus training group using the system with the intelligent assistant and the automatic explanation, and another group using only a simulator. This process will be supervised by some human experts on plant operation and they will give a grade about the recommended system actions and their own recommended expert actions.

6 Conclusions and future work

Explanations are important for intelligent systems, in particular for learning. We have developed an explanation generation mechanism for a training tutor that combines several knowledge sources to fill–in a template inspired on experts' explanations. For this we combine information extracted from an MDP model of the process, with domain knowledge represented with a frame hierarchy; and adapt it according to the user level. We applied it to the power plant domain with promising results. As future work we plan to refine the textual element of the explanation and evaluate it in a controlled user study. Also, include more domain variables will be an important issue in this work.

References

1. Herrmann, J., Kloth, M., Feldkamp, F.: The role of explanation in an intelligent assistant system. In: Artificial Intelligence in Engineering. Volume 12., Elsevier Science Limited (1998) 107–126
2. Elizalde, F., Sucar, E., DeBuen, P.: A prototype of an intelligent assistant for operator's training. In: International Colloquium for the Power Industry, México, CIGRE-D2 (2005)
3. Reyes, A., Sucar, L.E., Morales, E., Ibarguengoytia, P.H.: Solving hybrid markov decision proceses. In: MICAI 2006: Advances in Artificial Intelligence, Apizaco, Mexico, Springer-Verlag (2006)
4. Elizalde, F., Sucar, E., DeBuen, P.: Explanation generation trough probabilistic models for an intelligent assistant. IBERAMIA/SBIA/SBRN - WTDIA (2006)
5. Elizalde, F., Sucar, E., Reyes, A., DeBuen, P.: A MDP approach for explanation generation. In: Workshop on Explanation-Aware Computing, ExaCt07, Vancouver, CA., AAAI (2007)
6. Puterman, M.: Markov Decision Processes: Discrete Stochastic Dynamic Programming. Wiley, New York (1994)
7. Bellman, R.: Dynamic Programming. Princeton U. Press, Princeton, N.J. (1957)
8. Boutilier, C., Dean, T., Hanks, S.: Decision-theoretic planning: structural assumptions and computational leverage. Journal of AI Research **11** (1999) 1–94
9. Minsky, M.: A framework for representing knowledge. In Winston, P., ed.: The Psychology of Computer Vision, McGraw-Hill (1975)
10. Vadillo-Zorita, J., de Ilarraza, A.D., Fernández, I., Gutirrez, J., Elorriaga, J.: Explicaciones en sistemas tutores de entrenamiento: Representacion del dominio y estrategias de explicacion, Pais Vasco, Espa˜na (1994)