

# An Intelligent Assistant for Power Plants based on Factored MDPs

Alberto Reyes  
Instrumentation and Control Department  
Instituto de Investigaciones Eléctricas  
Cuernavaca, México  
Email: areyes@iie.org.mx

Matthijs T. J. Spaan  
Institute for Systems and Robotics  
Instituto Superior Técnico  
Lisbon, Portugal  
Email: mtjspaan@isr.ist.utl.pt

L. Enrique Sucar  
Computer Science Department  
INAOE  
Puebla, México  
Email: esucar@inaoep.mx

**Abstract**—Making good operation decisions during abnormal power plant conditions represents in many cases the possibility to avoid a unit trip or having economical losses. This paper introduces AsistO, an intelligent assistant for the decision support based on decision theoretic planning techniques. It provides power plant operators with useful recommendations to (i) maintain a plant running under safe conditions, or (ii) deal with process transients when an unexpected event occurs. We present the formalism of Markov decision processes as the core of the intelligent assistant which uses a factored representation of plant states. We also show a very intuitive algorithm to approximate decision models based on training data collected through random exploration routines in a simulated environment. We have tested our system in the steam generation system of a combined power plant to deal with load disturbances.

## I. INTRODUCTION

In many industrial processes, plant operators are faced with a large amount of problems and information based on which they have to take decisions. To support such decisions, automatic assistants exist that provide operators with a list of suggested commands [1]. Either when the operator executes a command or when a unexpected disturbance occurs, a new list of recommended actions is presented. This recommendation process can be modelled as a sequential decision problem under uncertainty with an optimization criteria such as performance, availability, reliability, or security. Thus, we suggest the use of Markov Decision Processes (MDP) [2], a well known stochastic method for sequential decisions.

In the context of our application, intelligent assistants (IA) are knowledge-based systems for the decision support. They should provide users with accurate information at the right moment, and do suggestions and criticisms during the decision making process [3]. The basic functions of an IA are: knowledge acquisition and representation, simulation, test case generation, problem solving, and knowledge transfer through explanations to users [4]. They could additionally include an effective human-computer interaction mean to i) provide the system with big amounts of world knowledge [5], and ii) integrate data base management functions with other information and knowledge-based systems [3]. Optionally, an assistant could also adapt to users through simply observing their general behavior [6]. In other words, the system could learn from user's experiences.

Among the most representative work in the field of intelligent assistants, the following systems can be found: AS-TRAL [7], which is a simulator-based assistant for power operator's training. Its main functions are the recognition of the actions executed by an operator in a plant simulator, and the classification of detected errors with respect to an expected behavior. The main effort in this work is oriented to the development of explanation systems that support the operator's understanding about the plant state. SOCRATES [8] is a real time assistant for control center operators in alarm processing and energy restoration. The core of the system is SPARSE, an expert system initially developed to use it in power transmission and distribution control centers. SOCRATES also provides an intelligent tutor, SPARSE-IT, which fulfills two purposes: i) to show users how a trained operator solves problems and ii) to train the user to deal with specific situations and evaluate its performance. SART [9] is a traffic control support system for the French subway system, and implements an intelligent tutor with several functions such as: knowledge acquisition from operators, traffic simulation, model management of the network to test with alternative cases, enumeration of alternative solutions to incidents, and training of new operators. SART uses a multi-agent approach to have an evolutionary architecture.

In this paper we present AsistO (standing for *Operation ASSISTant* in Spanish), an intelligent assistant for power plant operators with similar features to the systems described above. Its main advantage is that it provides on-line guidance in the form of ordered recommendations based on techniques such as decision-theoretic planning, machine learning, and probabilistic reasoning. The system allows to deal with abnormal situations, non-expected events, or the occurrence of process transients.

The paper is organized as follows: Section 2 presents a typical load disturbance in the steam generation system of a combined cycle power station that can be treated using an intelligent assistant. In section 3, we explain the formalism of factored MDPs and show a simple algorithm for learning factored decision models. Section 4 shows the AsistO's general architecture. In section 5 some experiments are commented. Finally, we discuss some important issues and future work in section 6.

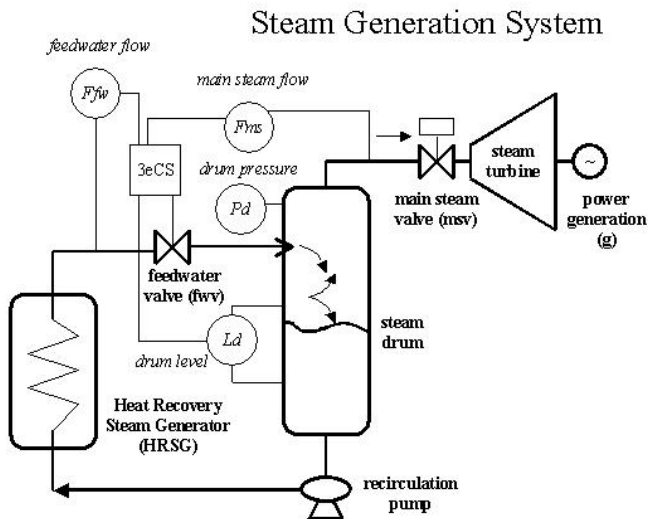


Fig. 1. Simplified diagram of the steam generation system showing its main components, control devices and instrumentation. The gas turbine connection is not shown.

## II. PROBLEM DOMAIN

Modern power plants are following two clear tendencies. First, they are very complex processes working close to their limits. Second, they are highly automated and instrumented, leaving the operator with very few decisions. However, there still exist some maneuvers that require the experience and ability of the operator. In order to illustrate how important the decisions of a human operator are to overcome a transient, we have selected an electric load disturbance as a typical problem in the steam generation system of a power plant.

The complete process control domain is shown in Fig. 1. A heat recovery steam generator (HRSG) is a process machinery capable of recovering residual energy from the exhaust gases of a gas turbine to generate high pressure ( $Pd$ ) steam in a special tank (*steam drum*). The *recirculation pump* is a device that extracts residual water from the steam drum to keep a water supply in the HRSG ( $Ffw$ ). The result of this process is a high-pressure steam flow ( $Fms$ ) that keeps running a *steam turbine* to produce electric energy ( $g$ ) in a *power generator*. The main control elements associated are the feedwater valve ( $fww$ ) and main steam valve ( $msv$ ). An electric load disturbance ( $d$ ) is an exogenous event that, as well as the control valves, could induce a state transition. The problem is to obtain a control strategy that considers stochastic commands on the valves and, according to an experience-based preference function, maximizes the security in the drum, and/or the power generation.

A practical solution is the use of an intelligent operator assistant providing recommendations about how to take the best action on the process that corrects the problem. The operator assistant should be able to find an action policy according to the crisis dimension, take into account that actuators are not perfect and can produce non-desired effects,

and consider the performance, availability and reliability of the actual plant installations under these situations.

## III. FACTORED MARKOV DECISION PROCESSES

A Markov decision process (MDP) [2] models a sequential decision problem, in which a system evolves over time and is controlled by an agent. At discrete time intervals the agent observes the state of the system and chooses an action. The system dynamics are governed by a probabilistic transition function  $\Phi$  that maps states  $\mathbf{S}$  and actions  $\mathbf{A}$  (both at time  $t$ ) to new states  $\mathbf{S}'$  (at time  $t + 1$ ). At each time, an agent receives a scalar reward signal  $R$  that depends on the current state  $s$  and the applied action  $a$ . The performance criterion the agent should maximize considers the discounted sum of expected future rewards, or value  $V$ :  $E[\sum_{t=0}^{\infty} \gamma^t R(s_t)]$ , where  $0 \leq \gamma < 1$  is a discount rate. The main problem is to find a control strategy or *policy*  $\pi$  that maximizes the expected reward  $V$  over time.

For the discounted infinite-horizon case with any given discount factor  $\gamma$ , there is a policy  $\pi^*$  that is optimal regardless of the starting state and that satisfies the *Bellman* equation [10]:

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

Two methods for solving this equation and finding an optimal policy for an MDP are: (a) dynamic programming [2] and (b) linear programming.

In a factored MDP, the set of states is described via a set of random variables  $\mathbf{S} = \{X_1, \dots, X_n\}$ , 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, as 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) [11], [12] 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 [13].

### A. Learning factored models

The MDP model is learned from data based on a random exploration in a simulated environment. We assume that the agent can explore the state space, and that for each state-action cycle it can receive some immediate reward. Based on this random exploration, the reward and transition functions are induced.

Given a set of  $M$  non-ordered and rough (discrete and/or continuous) random variables  $S^j = X_1, \dots, X_n$  defining a deterministic state, an action  $a^j$  executed by an agent from a finite set of actions  $A = \{a_0, a_1, \dots\}$ , and a reward (or cost)  $R^j$  associated to each state in an instant  $j = 1, 2, \dots, M$ , we can learn a factored MDP model as follows:

- 1) Discretize the continuous attributes from the original sample  $D = \{S, R, a\}$ . This transformed data set is called the discrete data set  $D_d = \{S_d, R_d, a_d\}$ . For small state spaces, use conventional statistical discretization techniques. However, in complex state spaces, abstraction techniques are more efficient. For further details see [14], [15].
- 2) From the subset  $\{S_d, R_d\}$  induce a decision tree,  $RDT$ , using the algorithm C4.5 [16]. This predicts the reward function  $R_d$  in terms of the discrete state variables,  $X_1, \dots, X_n$ .
- 3) Format the discrete data set in such a way that the attributes follow a temporal causal ordering. For example variable  $X_{0,t}$  before  $X_{0,t+1}$ ,  $X_{1,t}$  before  $X_{1,t+1}$ , and so on. The whole set of attributes should have the form  $X_t, X_{t+1}, a_t$ .
- 4) Prepare a data set for the induction of a set of 2-stage dynamic Bayesian nets. According to the action space dimension, split the discrete data set into  $|A|$  subsets of samples for each action. Remove the attribute  $a_t$  from all of them.
- 5) Induce a transition model for each subset using the K2 algorithm [17]. The result is a 2-stage dynamic Bayesian net for each action  $a \in A$ .

This approximate model can be solved using value iteration to obtain the optimal policy. This approach has been successfully applied in other domains [18].

#### IV. RECOMMENDER SYSTEM FOR THE STEAM GENERATOR OPERATION

AsistO is an intelligent assistant that provides useful recommendations for training and on-line assistance in the power plant domain. The assistant is coupled to a power plant simulator capable to partially reproduce the operation of a combined cycle power plant, in particular, the steam generation process described in section II.

The simulator (Fig. 2) is provided with controls for setting power conditions in the gas and steam turbines (nominal load, medium load, minimum load, hot standby, low speed, and start-up). It also provides an operation panel to set load demands, unit trips, shutdowns, and other high-level operations in different subsystems. It includes a visualization tool for tracking the behavior of user-selected variables in time, and recording historical data. The simulator, that was implemented in MS Visual C++, can replicate data to MS SQL-Server and MySQL database formats.

The AsistO system is composed by a decision model base, a simulation data base, and the following subsystems:

- 1) Data management.
- 2) Model management.
- 3) Planning subsystem.
- 4) User interface.

The simulation data base comprises process signals generated in the simulator (outputs), and control signals (inputs) sent from the user interface to set a specific electric load or failure condition in the process.

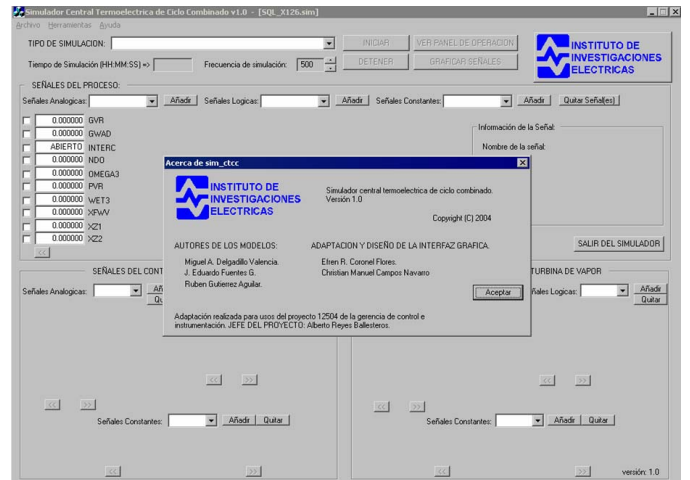


Fig. 2. Steam Generation Simulator provided with controls, an operation panel, and data visualization tools.

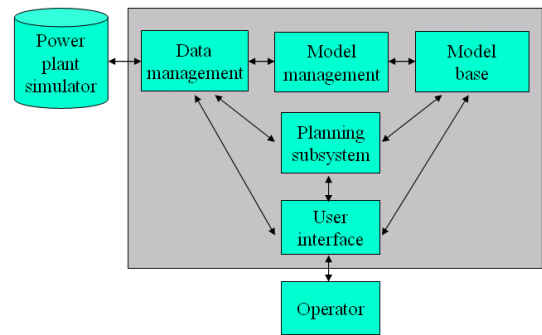


Fig. 3. AsistO's general architecture. Given a current state, the planning subsystem queries a recommendation to the model base. This is presented to the user interface for the decision making.

The decision model base stores the transition and reward functions in a factored form. The transition model is implemented in Elvira [19] (which was extended to compute Dynamic Bayesian Networks) and the reward function in Weka [20]. The transition and reward functions are effectively learned using the K2 algorithm available in Elvira, and the reward function using a Java implementation of the C4.5 algorithm available in Weka (J4.8).

In AsistO's general architecture the planning subsystem obtains the plant state from the simulation data base. Then it queries the policy function for the current state in the model base to obtain a recommendation. Both current state and recommendation, are shown graphically through the user interface to the operator who finally decides whether or not to execute the recommended command. Fig. 3 shows the basic AsistO's general architecture.

The data management subsystem is composed by a set of tools for data administration and analysis software. The formats supported by AsistO are Microsoft SQL Server and MySQL. MySQL is open source and provides fast performance, high reliability and easy use. This format was selected because it can run in more than 20 platforms such as Linux,

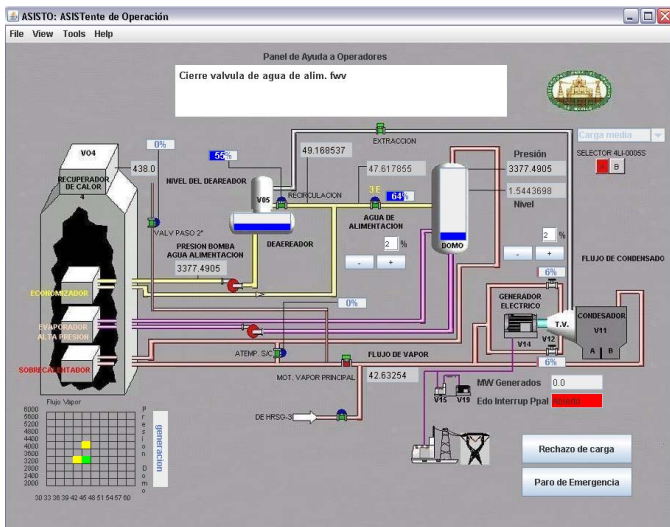


Fig. 4. The user interface is the link between the recommender system and the operator. It includes supervision, problem specification and manual control capabilities.

Windows, OS/X and HP-UX.

The model management subsystem manipulates the structures stored in the model base. It is also based on the academic tools Elvira and Weka.

The planning subsystem uses the decision models allocated in the model base and its inference algorithms to build an optimal policy plant state-recommendation. The resulting policy and utility functions are also stored in the model base. The planning subsystem in AsistO is based on SPUD [21], that includes a very efficient version of the value iteration algorithm for factored MDPs.

The user interface provides the communication with the environment. In this case, the power plant simulator is the environment, and the operator is the actor which provides the goals and executes the recommendations that modify the environment. The user interface is implemented in the Java language and it is provided with controls for command execution, load selection, failure simulation, and recommendation display. This module, that can also be used as a supervision console, includes the controls for random exploration and system sampling for the learning purposes described in section III-A. It also provides a graphical interface to observe how fast the correct execution of recommendations impact in the plant operation. The main features of the user interface are shown in Fig. 4.

## V. EXPERIMENTAL RESULTS

We used AsistO to run a series of experiments with different complexities. In the first set of experiments, we specified a 5-action hybrid problem with 5 variables ( $Fms$ ,  $Ffw$ ,  $Pd$ ,  $g$ ,  $d$ ). We also defined a simple binary reward function based on the safety parameters of the drum ( $Pd$  and  $Fms$ ). The relationship between their values and the reward received can be seen in Fig. 5 (top). Central black squares denote safe states (desired operation regions), and white zones represent non-rewarded

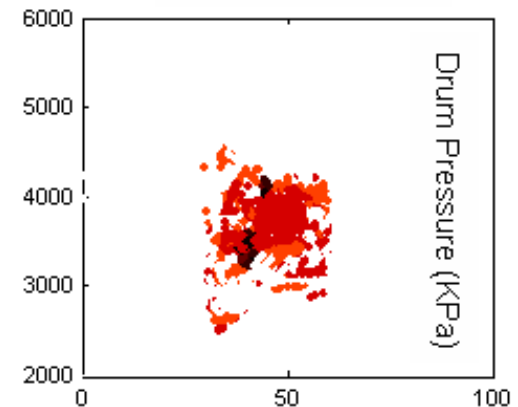
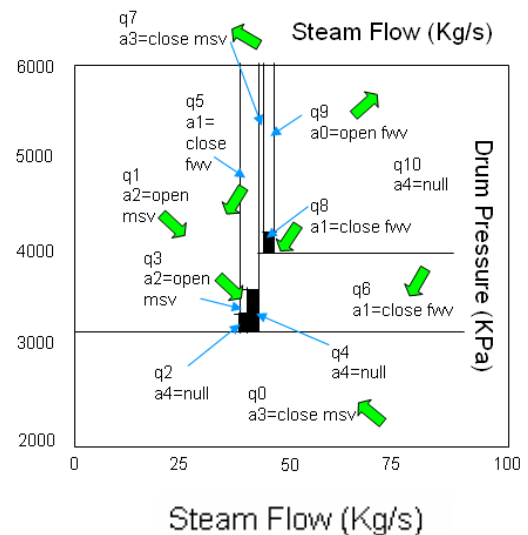


Fig. 5. Process control problem. Top: reward function where black dots represent desired regions, discrete state partition (in this case 10 states) and policy found with value iteration. Bottom: exploration trace, where black dots Black represent sampled states with positive reward, red dots have no reward, and white regions unexplored zones.

zones (indifferent regions). To learn the model and the initial abstraction, samples of the system dynamics were gathered using simulation. Black dots in figure 5 (bottom) represent sampled states with positive reward, red dots have no reward, and white zones were simply not explored. Figure 5 (top) shows the state partition and policy found (green arrows) by the learning system. For this simple example, although the resulting policy is not very detailed (discrete states are quite large), it follows the idea of going to the lower black regions. When analyzed by an expert operator, this control strategy is near-optimal in most of the abstract states.

We solved the same problem but adding two extra variables, the position for valves  $msv$  and  $fww$ , and using 9 actions (all the combinations of open-close valves  $msv$  and  $fww$ ). We also redefined the reward function to maximize power generation,  $g$ , under safe conditions in the drum. Although the problem increased significantly in complexity, the policy obtained is “smoother” than the 5-action simple version presented above. To give an idea about the computational effort, for a fine

discretization (15,200 discrete states) this problem was solved in 859.2350 seconds, while using a more abstract representation (40 discrete states) it took only 14.2970 seconds. In both cases, the approximated models were found using the SPUDD system.

## VI. DISCUSSION AND FUTURE WORK

This paper introduced AsistO, an intelligent assistant based on factored MDPs that provides power plant operators with useful recommendations. We explained the formalism of factored Markov decision processes and a intuitive algorithm to approximate decision models based on training data. The paper also describes the AsistO's general architecture which in general might be implemented using conventional software tools. The academic software for planning and learning is also a well known and robust platform for research in artificial intelligence. The results demonstrate the factibility of the method for the decision making and plant optimization, an reveal that eventually the assistant could also be extended for personnel training purposes.

A drawback of an MDP model is that it assumes that all the state variables relevant for decision making are observed without noise. However, real-world sensors are prone to noise. Furthermore, there might be situations that cannot be detected directly using the available sensors. In this case the state observed by the MDP is no longer Markovian, and hence the value of the computed policies will no longer be accurate.

For instance, during normal operation, the conventional three-element feedwater control system (*3eCS*) commands the feedwater control valve (*fwv*) to regulate the steam drum level (*Ld*). However, when a partial or total electric load rejection (a partial or total decrement in the load connected to the power generator *g*) is presented this traditional control loop is not longer capable to stabilize the drum level. In this case, the steam-water equilibrium point changes, causing an enthalpy change of both fluids (steam and water). Consequently, the enthalpy change causes an increment in the water level because of a strong water displacement to the steam drum. The control system reacts by closing the feedwater control valve. However, an increment of feedwater is needed instead of a decrement. A similar effect is presented when a sudden high load demand occurs.

To tackle at the same time the problem of noisy sensors and limited observability, we are extending AsistO to consider Partially Observable MDPs (POMDPs) [22]. POMDPs extend the MDP framework with an observation function, which stochastically related observations to states. Instead of reasoning over states, policies now map probability distributions over states, so-called beliefs, to actions. Efficient algorithms for approximately solving factored POMDPs are available [23], and no global changes to the AsistO architecture have to be made.

We can model the load rejection problem detailed above by adding a (binary) state variable that models whether or not a load rejection is occurring. This variable cannot be observed directly, but we maintain a belief whether its true or false. This

belief can be updated (using Bayes' rule) given observations of the other state variables, as they can give a clue about its state, for instance because actions do not have the intended effect. In a POMDP formulation such clues can be directly coupled to the state of the load-rejection variable. Adding the variable in a POMDP setting will allow the system to consider the possibility of a load rejection, and to optimize the policy in case it happens, even if it cannot be detected directly.

Finally, since AsistO is aimed at either operation assistance or operator training, we are currently developing an extra module that explains the recommended commands generated by the planning subsystem and, provides, after a bad decision, the reason why a recommendation should have been followed [24]. We are also integrating capabilities of diagnosis [25] into AsistO.

## ACKNOWLEDGMENTS

This work was supported by Instituto de Investigaciones Eléctricas-México (project no. 13773); the International Cooperation Programme for the Promotion of Scientific and Technological Research of the European Union and Mexico (FONCICYT project no. 95185); Fundação para a Ciência e a Tecnologia-Portugal (ISR/IST pluriannual funding) through the POS\_Conhecimento Program that includes FEDER funds, and through grant PTDC/EEA-ACR/73266/2006.

## REFERENCES

- [1] A. Reyes, L. E. Sucar, E. Morales, and P. H. Ibarguengoytia, "Abstract MDPs using qualitative change predicates: An application in power generation," in *Planning under Uncertainty in Real-World Problems Workshop. Neural Information Processing Systems (NIPS-03)*, Vancouver CA, Winter 2003.
- [2] M. Puterman, *Markov Decision Processes*. New York: Wiley, 1994.
- [3] A. Aamodt and M. Nygard, "Different roles and mutual dependencies of data, information and knowledge: An AI perspective on their integration," *Data and Knowledge Engineering (North-Holland Elsevier)*, vol. 16, pp. 191–222, 1995.
- [4] P. Brézillion and E. Cases, "Cooperating for assisting intelligently operators," in *Proceedings of COOP-95*. INRIA Ed., 1995, pp. 370–384.
- [5] G. Fischer, "Communication requirements for cooperative problem solving systems," *Information Systems*, vol. 15, no. 1, pp. 21–36, 1990.
- [6] J. Frontin, A. H. Kacem, and J. Soubie, "Acquérir des connaissances et structurer le système pour coopérer," in *2ndes Journées Acquisition des Connaissances*, Saint Raphaél, 1993.
- [7] M. Caimi, C. Lanza, and B. Ruiz-Ruiz, "An assistant for simulator-based training of plant operator," *Maria Curie Fellowships Annual Report*, Tech. Rep., 1999.
- [8] Z. Vale, C. Ramos, A. Silva, L. Faria, J. Santos, F. Fernández, C. Rosado, and A. Marques, "SOCRATES, an integrated intelligent system for power system control center operator assistance and training," in *Proceedings of the International Conference on Artificial Intelligence and Soft Computing (IASTED)*, 1998, pp. 27–30.
- [9] P. Brézillion, C. Gentile, I. Saker, and M. Secron, "SART: A system for supporting operators with contextual knowledge," in *Proceedings of the International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT-97)*, Rio de Janeiro, Brasil, February 1997.
- [10] R. Bellman, *Dynamic Programming*. Princeton, N.J.: Princeton U. Press, 1957.
- [11] T. Dean and K. Kanazawa, "A model for reasoning about persistence and causation," *Computational Intelligence*, vol. 5, pp. 142–150, 1989.
- [12] A. Darwiche and G. M., "Action networks: A framework for reasoning about actions and change under understanding," in *Proceedings of the Tenth Conf. on Uncertainty in AI, UAI-94*, Seattle, WA, USA, 1994, pp. 136–144.

- [13] C. Boutilier, T. Dean, and S. Hanks, "Decision-theoretic planning: structural assumptions and computational leverage," *Journal of AI Research*, vol. 11, pp. 1–94, 1999.
- [14] R. Munos and A. Moore, "Variable resolution discretization for high-accuracy solutions of optimal control problems," in *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI-99)*, T. Dean, Ed. Morgan Kaufmann Publishers, San Francisco, California, USA, August 1999, pp. 1348–1355.
- [15] A. Reyes, L. E. Sucar, E. Morales, and P. H. Ibarguengoytia, "Abstraction and refinement for solving Markov Decision Processes," in *Workshop on Probabilistic Graphical Models PGM-2006*, Czech Republic, 2006, pp. 263–270.
- [16] J. Quinlan, *C4.5: Programs for machine learning*. San Francisco, Calif., USA.: Morgan Kaufmann, 1993.
- [17] G. F. Cooper and E. Herskovits, "A bayesian method for the induction of probabilistic networks from data," *Machine Learning*, 1992.
- [18] A. Reyes, L. E. Sucar, E. Morales, and P. H. Ibarguengoytia, "Learning qualitative Markov decision processes," December 2005, presented at Neural Information Processing Systems NIPS 2005 Workshop on Machine Learning Based Robotics in Unstructured Environments.
- [19] E. Consortium, "Elvira: an environment for creating and using probabilistic graphical models," U. de Granada, Spain, Tech. Rep., 2002.
- [20] I. Witten, *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations, 2nd Ed.* USA: Morgan Kaufmann, 2005.
- [21] J. Hoey, R. St-Aubin, A. Hu, and C. Boutilier, "Spudd: Stochastic planning using decision diagrams," in *Proc. of the 15th Conf. on Uncertainty in AI, UAI-99*, 1999, pp. 279–288.
- [22] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial Intelligence*, vol. 101, pp. 99–134, 1998.
- [23] P. Poupart, "Exploiting structure to efficiently solve large scale partially observable Markov decision processes," Ph.D. dissertation, University of Toronto, 2005.
- [24] F. Elizalde, L. E. Sucar, M. Luque, J. Diez, and A. Reyes, "Policy explanation in factored Markov decision processes," in *Proceedings of the 4th European Workshop on Probabilistic Graphical Models (PGM 2008)*, Hirtshals, Denmark, September 2008, pp. 97–104.
- [25] E. Morales and P. Ibarguengoytia, "On-line diagnosis using influence diagrams," in *Advances in Artificial Intelligence - MICAI 2004, LNAI 2313*, G. A. L.E. Sucar R. Monroy and e. H. Sossa, Eds. Berlin Heidelberg: Springer-Verlag, 2004, pp. 546–554.