Intelligent monitoring and maintenance of power plants

Dimitrios Kalles[1], Anna Stathaki[1] and Robert E. King[2]
[1]Computer Technology Institute,
PO Box 1122, 261 10, Patras
[2]Department of Electrical & Computer Engineering,
University of Patras, Patras
kalles@cti.gr

ABSTRACT

Conventional predictive maintenance involves continuous processing of real-time data from plant sensors of critical variables that are indicators of the health of the equipment. Some intelligent monitoring systems using rules elicited from maintenance personnel have being developed to infer the causes of impending faults. In this paper we propose a novel approach to intelligent predictive maintenance based on reinforcement learning. Following an outline of reinforcement learning, we explore the possibility of using the technique of reinforcement learning as the basis of an intelligent plant monitoring and predictive maintenance system.

INTRODUCTION

The electric power industry is continuously searching for ways to increase its efficiency, availability and reliability. In the light of de-regulation, public utilities must now compete with more efficient private utilities that are often keener to use new technologies with which to capture a large market share. Public utilities thus have little option but to adopt advanced computer-based technologies increasingly in order to maintain their market share. The power industry has not remained impassive to the use of artificial intelligence to solve many of its problems yet most attempts were not developed beyond the prototype level. In a series of special issues of *IEEE Expert* (Dabbaghchi *et al.*, 1997) the advances in the use of artificial intelligence in the power industry were analyzed at length. Power plant life assessment and maintenance, of paramount importance to these utilities, is finally attracting the attention it rightly deserves (Wehenkel, 1997).

Proper maintenance of power plant equipment is essential to high system availability. The complexity of modern power plant equipment is such, that few plant maintenance personnel are truly experts in diagnosing faults and even fewer are capable of predicting them. One approach to power plant equipment maintenance is predictive maintenance, which is used to examine the health of equipment. Here, off-line measurements of vibrations and temperatures, or signatures, from rotating equipment are taken periodically, analysed using well-known signal processing techniques and compared with desired signatures or previous signatures to observe how they evolve. When significant deviations occur, these are flagged so that appropriate action can be taken. The decision on whether or not the equipment requires maintenance is taken by human experts though expert systems are commercially available to this end. Maintenance is carried out only when necessary and not at regular intervals as in preventive maintenance, thereby reducing maintenance costs and the possibility of major breakdown significantly.

Essential to conventional predictive maintenance is any platform capable of data acquisition from suitable sensors, which supply a continuous stream of data on the health of the plant in addition to a real-time decision mechanism to predict impending faults so that immediate remedial action can be taken. Both qualitative as well as quantitative information from the plant can be fused in the inference mechanism to decide on the condition of the plant equipment and what maintenance is required and most importantly, when. The diagnostic problem starts with the observation of some measure of the deviation in the behavior of the plant equipment compared to that desired or expected. When a malfunction is observed, maintenance personnel hypothesize on its causes, based on their knowledge and experience. Attempting to accept or reject some of these hypotheses leads to further tests or uses deep knowledge of the specific malfunctioning equipment. This knowledge usually concerns the structural and behavioral characteristics of the equipment. Qualitative reasoning has attracted much attention in the past. Rule-based diagnostic systems with deterministic or fuzzy reasoning have proved capable of reaching conclusions using only shallow knowledge about the plant. Artificial Neural Networks allow for learning by viewing this problem as a pattern classification activity. Their mapping ability can be exploited to perform associations between input patterns derived from plant sensors and patterns representing fault conditions. Input patterns are represented by arrays of measurements while the output data from the system are arrays of the fault space representing the condition of the health of the equipment.

There is little doubt that plant equipment maintenance can be significantly enhanced if all primary mechanical and electrical plant equipment is monitored continuously and the data acquired is processed and deposited in a historical database. A decision support system could subsequently be used to infer impending faults in the equipment. Expert systems, some using fuzzy reasoning, have been implemented on a limited scale for fault prediction of large equipment.

We are presently exploring some of the new possibilities offered by advanced techniques of Computational Intelligence in engineering applications. One of these involves learning as part of the decision making process. Learning is used in the present context to predict plant faults and infer what preventive measures that must be taken to improve plant operation. In this paper we discuss the possibility of using *Reinforcement Learning* (Bertsekas and Tsitsiklis, 1989; Sutton and Barto, 1998) as a mechanism for fault prediction and how a learning machine can be used as part of an Intelligent Monitoring and Maintenance System for fault prediction, predictive maintenance and plant life assessment. The next section presents a brief outline of the essentials of reinforcement learning and of necessity it is presented on an abstract level. In the last section these ideas are directed to the specific application under consideration.

ESSENTIALS OF REINFORCEMENT LEARNING

Reinforcement Learning (or RL) is a relatively new computational approach to automating goal-directed learning and decision-making. It differs from other computational approaches by emphasizing on learning from direct interaction with its environment without relying on exemplary supervision or complete models of its environment. This section describes the essentials of RL very briefly as it could apply to predictive maintenance. We refer the reader to the book by Sutton and Barto (Sutton and Barto, 1998) for further reading. RL uses a formal framework defining the interaction between an *agent* (the medium for fault diagnosis) and its *environment* (i.e. power plant equipment) in terms of states, actions and rewards. The problem can be considered as one of attempting to achieve an explicit goal (i.e. fault prediction) through a sequence of causes and optimal effects in an environment that is both uncertain and vague. A schematic of the agent/environment interaction is shown in Figure 1.

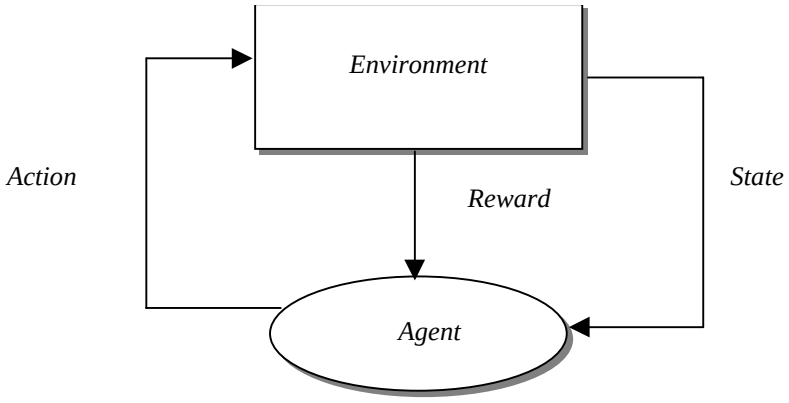

Figure 1: The RL agent-environment interface

The agent's function is to find a policy mapping states to actions that maximizes some long-term measure of reinforcement. We expect, in general, that the environment will be non-deterministic; that is, that taking the same action in the same state on two different occasions may result in different next states and/or different reinforcement values. However, we assume the environment is stationary; that is, that the probabilities of making state transitions or receiving specific reinforcement signals do not change over time. Beyond the agent and the environment, there are four main elements to any RL system:

- a *policy*,
- a *reward function*,
- a *value function* and
- a *model* of the environment.

A policy defines the way in which the agent is learning at a given time and the manner in which perceived states of the environment are transformed into actions that must be taken. In some cases the policy may be the result of inference by some expert system, whereas in others it may involve extensive searching. A reward function defines the objective or goal and maps the state-action pairs of the environment to a reward indicative of the intrinsic desirability of the state. The objective of the agent is to maximize the total reward it receives over a

finite time span. The reward function defines the good and bad events for the agent. Whereas a reward function indicates what is good in an immediate sense, a value (or return) function specifies what is good in the long term.

The value of a state is the total reward an agent can expect to accumulate over the future, starting from any given state. Whereas rewards determine the immediate desirability of any state, values indicate the long-term desirability of states after taking into account the state transitions that are likely to follow in the future and the rewards available in those states. Clearly, without rewards there could be no values and the only object of estimating values is to reap greater reward. Thus values are decisive in making and evaluating decisions. The choice of what actions must be taken at any juncture is based on value judgments. Actions that yield states of highest value, not highest reward, are sought because they maximize reward. In decision-making and planning, the derived value is the one with which we are most concerned. Unfortunately, it is much harder to determine values than rewards since they involve evaluation of a succession of cause-effect interactions over time. Rewards are basically given directly by the environment, but values must be estimated and re-estimated from the sequences of observations an agent makes over its entire lifetime.

The fourth and final element of some RL systems is a *model of the environment*. This is something that mimics the behavior of the application environment. For example, given a state and action, the model might predict the resultant next state and next reward. Models are used for planning, i.e. of predicting a course of action by considering possible future situations before they are actually experienced. The incorporation of models and planning into RL systems is a relatively new development. Early RL systems were explicitly trial-and-error learners. RL methods are closely related to dynamic programming and so they are closely related to state-space transition methods.

Originally posed as a technique for learning by trial and error by psychologists, RL was later reformulated as an optimal control problem or Markovian decision process, that can be solved using dynamic programming, which was introduced by Bellman in the mid-1950s (Bellman, 1957). The words *reinforcement* and *reinforcement learning* appeared for the first time in the engineering literature in the early 1960s in the context of learning machines and learning automata (Narendra and Thathachar, 1974). The work of Minsky (Minsky, 1961) on computational models of RL using early versions of artificial neural networks was particularly influential. The essential idea of RL, according to Klopf (Klopf, 1982), *"is the hedonistic aspect of behavior, i.e. the drive to elicit results from the environment while controlling it by driving it to desired ends and away from undesired ends"*.

Unlike supervised learning, RL involves search and memory: search in the form of trying and selecting from among many actions in each situation and memory in the form of remembering what actions worked best, associating them with the situations in which they worked best. Supervised learning is more pertinent to classifier systems, artificial neural networks and pattern recognition. RL differs from the more widely studied problem of supervised learning in several ways, the most important of which is that it does not depend on presentation of input/output pairs. Instead, after choosing an action the agent is told the immediate reward and the subsequent state, but is not told which action would have been in its best long-term interests. It is necessary for the agent to gather useful experience about the possible system states, actions, transitions and rewards actively to act optimally. Another difference from supervised learning is that on-line performance is important: the evaluation of the system is often concurrent with learning.

Some aspects of RL are closely related to search and planning issues in artificial intelligence. AI search algorithms generate a satisfactory trajectory through a graph of states. Planning operates in a similar manner, but typically within a construct with more complexity than a graph, in which states are represented by compositions of logical expressions instead of atomic symbols. These AI algorithms are less general than RL methods in that they require a predefined model of state transitions, and with a few exceptions assume determinism. On the other hand RL, at least in the discrete case for which some theory has been developed, assumes that the entire state space can be enumerated and stored in memory, an assumption to which conventional search algorithms are not tied.

The most important component of any RL schema is the method for estimating values efficiently. A variety of techniques have been proposed to this end. Search methods, using evolutionary computation techniques (such as genetic algorithms and simulated annealing) can be used to advantage to solve such problems directly. These methods search the space of policies directly, without requiring computation of value functions. If the space of policies is sufficiently small, or can be structured so that good policies are common or easy to find, then evolutionary methods are very effective. In addition, evolutionary methods have advantages on problems in which the learning agent cannot accurately sense the state of its environment. RL involves learning while interacting with the environment.

APPLICATION OF RL TO FAULT PREDICTION AND PREDICTIVE MAINTENANCE

We propose a setting in which RL can be applied to predictive maintenance of power plant equipment. This description is based on assumptions that are as weak as possible, in order to avoid application pitfalls and to ensure generality of the approach. A conceptual model of cause-effect pairs facilitates the description of interactions between system components by concentrating on the interchange of information between components of the system. The user arbitrarily sets the resolution of this model, i.e. the detail with which prognosis is to be made. The basic premises in developing the model of the system are the following:

(1) Each component of the system, C_i, is described as being in a state $C_i(t)$ at any time instant t, where $C_i(t)$ is some n_i -tuple of measurements of critical plant variables on which predictions will be based on whether the system requires maintenance or not.
(2) Each component of the n_i -tuple has an appropriate operating range [*min.*..*max*], from which some measure of the discrepancy from normal operating conditions is computed. It is noted that the range of every state variable can be suitably defined to permit open-ended intervals (e.g., of the [*min.*…] type) and that fuzzy characterizations can be included in this definition.
(3) The objective of the proposed diagnostic procedure is to infer (and then strive to avoid) the states that lead to an increase in the distance from the normal operating conditions.

To facilitate our treatment of predictive maintenance as an RL problem, we must make as association between the measurements (the various $C_i(t)$ described above) which furnish information on the state of the health of the plant equipment acquired by the plant data acquisition system and the maintenance actions, which are the key elements of searching the space. By way of example, consider a situation in which three interactive sub-processes are monitored and denote them by C_1, C_2, C_3. Let C_1 be a coal mill (with $C_1(t)$ involving, for instance, the pressure, temperature and feed rate), C_2 be a filter (with $C_2(t)$ being the filter voltage) and C_3 be a boiler (with $C_3(t)$ involving the pressure, temperature and water flow). The three sub-processes are closely linked and any action taken in one may affect the variables in the others. For instance, the temperature of $C_1(t)$ may affect the pressure of $C_3(t)$ at some instant $t+T$, i.e. $C_3(t+T)$. The temperature of $C_1(t)$ can be controlled manually or automatically. We model temperature at the coal mill as an action taken by an agent (human or automatic) that could affect the state of the boiler. This allows us to perceive measurements acquired from some equipment as actions in a generalized state space.

We now turn to the learning problem itself. The most straightforward situation can be cast as a dynamic optimization problem of the value function for a particular sub-process. Using our example, we require to minimize the reward function, which is a measure of the discrepancy of the temperature of the boiler from its nominal state. This can be represented by inverting the concept of reward and considering it as a penalty, assigning zero penalties for normal operation and positive ones for deviations from nominal operation. The measure could be defined conveniently in fuzzy terms. Penalties must be weighted over a time horizon since more recent penalties are clearly more important.

The problem state space can be represented as the set of all components (all $C_i(t)$ values, for each C_i monitored). We make the simplifying assumption that all process paths have unit path length. Though this may not always apply, it should not have any degenerating effect in our approximation, since the RL algorithm that will uncover the significant relationships does not really depend on exact knowledge of the process paths.

Approximations to the state space can be implemented using an artificial neural network, as is the case in the TD-Gammon system (Tesauro, 1995). Somewhat subtler, but tractable via extrapolation nonetheless, is the alignment of the time variable t, since it is unlikely that all components of the state are measured simultaneously.

The above description settles the problem for the monitoring of a single piece of equipment. We claim that by generalizing the concept for all monitored equipment, we will be able to identify faults or misbehaving paths. The conceptual extension is rather simple. We use the previous representation for *all* monitored components and try to optimize the penalties for all paths (this can be visualized as a complete directed graph between all components' n-tuple measurements).

The foregoing scheme can now be used in a predictive maintenance setting. By solving the RL problem for one component we can, in effect, establish how a particular state may affect the target equipment component over time. Note that although ambiguity between measurements is possible, there is always an upper bound on the time lag between an observed measurement and a penalty, as derived by our training data. Thus, while monitoring the primary variables we can maintain a frontier of possible problems over future time, for each component. This frontier expands and shrinks depending on what manual or automatic intervention is effected. That a problem persists at this frontier is an indication of the build-up of an impending fault condition.

When compared to traditional expert fault diagnostic and maintenance systems, the proposed approach should, after the period of learning, be able to perform as well, if not better, based on the fact that knowledge bases are not easily maintainable, whereas RL is designed to work seamlessly and in an incremental manner. It is also reasonable to expect that that proposed approach will also outperform neural network based fault predictors, not only because it incorporates explicit modeling of time but also because it supports the creation of an explanatory component (the problem frontier).

The underlying premise of this application of RL is the holistic consideration of maintenance problems. However, two major issues remain to be resolved. The first refers to the correct definition of a learning episode, as there are no clear-cut intervals in plant operations and thus no obvious time horizon for training. The second issue is that actual faults must occur during training to infer non-standard behavior that might lead to problem prediction (this clearly holds for *any* prediction mechanism). Both caveats are liable to slow the learning process in its quest for convergence. To assist the process, any available domain knowledge has to be meticulously tracked down and accounted for (this, for example, could rapidly lead to a reduced version of the directed graph described above and a smaller search space).

Of course, an explicit trial-and-error strategy is *not* required for RL to be of practical value. An RL-based predictive maintenance system need only operate in parallel with existing supervisory and monitoring systems and learn by *their* errors. Such errors are bound to occur in any real industrial setting due to the multitude of unforeseen circumstances. What is more important, and just as practical, is that an RL system can learn part of its state space by monitoring deviations which do not necessarily lead to abnormal behavior, either due to self-correction or due to manual intervention. Such a feature, inherent in RL, allows a potential RL-based system to accumulate knowledge, as much as required by the maintenance engineers who can then decide when they wish to use it as an intelligent assistant among the maintenance techniques that already exist.

DISCUSSION

Predictive maintenance of large-scale plants is essential in today's competitive market where every minute of equipment down-time means loss of production, loss of income and loss of strategic position. It is true to say that in the next twenty years or so, fault prediction and predictive maintenance systems will become integral components of most supervisory control and data acquisition systems. New computational techniques offer a real challenge in the field, offering new mechanisms for learning about the plant, its fault patterns and inferring potential malfunctioning equipment. Combined with life cycle assessment of plant equipment, such systems could prove to be invaluable to plant management.

Learning the fault patterns and fault sequences by trial and error appears to be a viable way in which to reformulate the fault prediction problem. Reinforcement learning, which only recently has been considered in industrial applications, may well be the vehicle that offers solutions, where others have failed.

REFERENCES

Bellman, R. E. (1957). *Dynamic Programming*. Princeton University Press.

Bertsekas, D. P. and Tsitsiklis, J. N. (1989). *Parallel and Distributed Computation: Numerical Methods*. Prentice-Hall, Englewood Cliffs, NJ.

Dabbaghchi I., Christi R., Rosenward G. and Liu C. (1997), AI Application Areas in Power Systems, *IEEE Expert*, Jan-Feb, pp. 58-66.

Klopf, A. H. (1982). *The Hedonistic Neuron: A Theory of Memory, Learning, and Intelligence*. Hemisphere, Washington, D.C.

Minsky, M. L. (1961). Steps toward artificial intelligence. *Proceedings of the Institute of Radio Engineers*, 49, pp. 8-30. Reprinted in E. A. Feigenbaum and J. Feldman, (Eds), *Computers and Thought*. McGraw-Hill, New York, pp. 406-450, 1963.

Narendra, K. S. and Thathachar, M. A. L. (1974). Learning automata - a survey. *IEEE Trans. on Systems, Man, and Cybernetics*, 4, pp. 323-334.

Sutton R. and Barto A. G. (1998) : *Reinforcemnent Learning: an Introduction"*, MIT Press, Boston, Ma.

Tesauro G.J.. (1995), Temporal difference learning and TD-Gammon, *Communications of the ACM*, 38, pp. 58-68.

Wehenkel L. (1997). Machine-Learning Approaches to Power-System Security Assessment, *IEEE Expert*, Sep-Oct., pp. 60-72.