Relational MDPs using Qualitative Proportionality Predicates: An Application in Power Generation

Alberto Reyes†, L. Enrique Sucar‡, Eduardo Morales‡, and Pablo Ibargüengoytia†
† Instituto de Investigaciones Eléctrica, Reforma 113, Cuernavaca, Mor., México
‡ ITESM Cuernavaca, Reforma 182-A, Cuernavaca, Mor., México
{areyes,pibar}@iie.org.mx, {esucar,eduardo.morales}@itesm.mx

Abstract

We propose an alternative form of abstract MDP representation, where state variables are qualitative proportionality predicates and actions are STRIPS-like operators. Based on this representations, we use learning techniques to extract the action and the transition models from data. K2 and EM are used to learn the structure and parameters of the a dynamic Bayesian network that specifies the transition function, and C4.5 to obtain the relational predicates that represent the states. The combination of rule induction to learn the relational states, and DBN learning to obtain the transition model, make it possible to specify an MDP for complex domains, such as power generation. We present initial results of an experiment with a simulator of a turbogas power plant.

1 Introduction

Markov Decision Processes (MDPs) [3] have developed as a standard method for representing uncertainty in decision-theoretic planning. Traditional MDP solution techniques have the drawback that they require an explicit state representation, limiting their applicability to real-world problems. Recent work addresses this drawback via compactly specifying the state-space in factored form as the set of possible assignments to a set of state variables by using decision trees and dynamic Bayesian networks as a compact representation [9]. Such Factored MDPs allow for representing exponentially large state spaces [5]. The algorithms for planning using MDPs, however, still run in time polynomial in the size of the state space or exponential in the number of state-variables.

Many complex systems are naturally represented using some form of relational representation. We investigate methods for incorporating relational representations into the state-space of MDPs to allow their application to complex problems for which it may be impractical to use less expressive representations. Recent work has developed Probabilistic Relational Models (PRMs) for representing probability distributions over relational domains, building on the ideas of BNs [8, 6]. Morales [7] represents states through sets of relational properties. In this representation, actions are STRIPS-like operators where preconditions and postconditions are also relational states. We propose to represent relational states as sets of qualitative proportionality predicates Q^+ and Q^- [4], specially in cases where primitive state variables are continuous. Based on this representations, we use learning techniques to extract the action and the transition models. K2 and EM are used to learn the structure and parameters of the a dynamic Bayesian network that specifies the transition function, and C4.5 to obtain the relational predicates that represent the actions. We present initial results of an experiment with a simulator of a turbogas power plant.

2 Abstract MDPs

Traditional MDP solution techniques have the drawback that they require an explicit state space, limiting their applicability to real-world problems. Factored and relational representations address this drawback via compactly specifying the state-space in factored form. In a factored MDP, the set of states is described via a set of random variables X=X1,.., Xn, where each Xi takes on values in some finite domain Dom(Xi). The framework of dynamic Bayesian networks (DBN) [11] gives us the tools to describe the transition model function concisely. For each action, a two-stage DBN specifies the transition model. An even more compact representation can be obtained by representing the transition tables and value functions as decision diagrams [5].

2.1 Relational Representations.

The main idea behind this proposal is to represent states as sets of properties that can be used to characterize a large set of states. In a relational MDP, state variables are predicates, so a relational state (r–state) is a conjunction of first–order relations. The extension of an r–state is the set of states which are covered by its description. Each description is an instance of one and only one r–state. In the power generation domain, these relations could be: *close_to(current_mw, demand_mw), not away_from(turbine_vel, synchronism_vel)*, etc. A relational state can cover a large number of states, for instance: *away_from(current_mw, demand_mw) AND not away_from(turbine_vel, synchronism_vel)* can cover all cases where simultaneously power generation is much less that demanded power, and turbine angular velocity is slightly less than synchronism velocity.

The qualitative proportionality predicates $Q+$ and $Q-$ [4], or their generalization $M+$ and $M-$ from QSIM [1], are special cases of first order relations. They only consider three kind of predicates: increase, decrease, or no_change, and only relate two terms, which are usually but no necessarily continuous variables. A way to represent the system state is through qualitative change vectors at a particular point in the function (see fig. 1).As an example, taken from [10], consider a closed container where pressure is a function of gas temperature and volume according to: $Pres = 2\ Temp/Vol$. A way to represent the system state is through qualitative change vectors at a particular point in the function. For example, the point *(Temp=300, Vol=50, Pres=12.00)* give the qualitative change vector $q3 = (q_{Tem} = neg, q_{Vol} = neg, q_{Pres} = pos)$ referred to $e = (Temp = 315, Vol = 56, Pres = 11.25)$. This is equivalent to the relational state $neg(Temp._3, Temp._e)$ AND $neg(vol_3, vol_e)$ AND $pos(pres_3, pres_e)$.

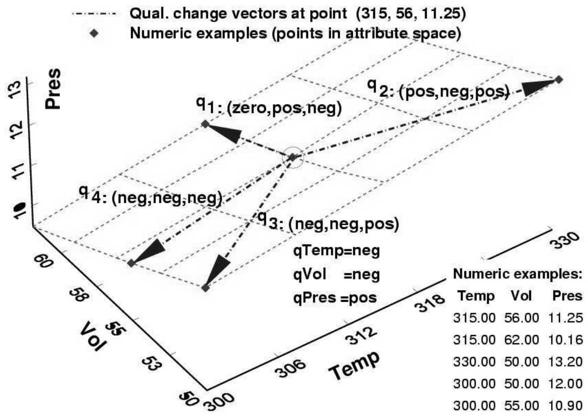

Figure 1: Temperature, volume and pressure are represented as points in the variable space. Arrows denote qualitative change vectors at the circled point.

Once a set of relations has been defined, the search space in the relational space is completely specified. Considering that relations can also occur in their negated form, for N relations there are potentially $2N$ r–states. Fortunately, not all combinations of the relations are possible in any particular domain, and huge reductions in the number of states and actions can be obtained in practice.

The set of actions also use a first–order relational representation. Rather that trying to apply all the available actions, we want to apply only relevant actions. A relational action (r–action) can be applied to many r–states, and as expected, not all r–actions apply to all r–states. An r–action is a STRIPS–like operator where preconditions are r–states. The following clause represents the r–action *rise–load*, using Prolog notation:

```
raction(rise(vel), Postcondition):-
    increase(demand_mw, current_mw),
    decrease(current_vel, synchronism_vel),
    effect(Postcondition).
```

In this operator, the precondition is the relational state defined by the two first predicates, while the postcondition is the deterministic effect resulting from the action. This r–action is only relevant under this precondition. We assume that post conditions can be obtained from the transition model and can be omitted. The r–action selected from the conflict set will depend on the utility function.

For an r–action to be properly defined, it must satisfy the following condition: if an r–action is applicable to a particular instance of an r–state, then it should be applicable to all of the instances of that r–state.

3 A case study: power generation

The main components of the gas turbine (GT) system are: the starting device, air compressor, gas turbine and generator. The starting device is a 500 HP diesel engine which accelerates the turbine from turning gear, through fuel ignition and light off, until about 2140 RPM; at this point the fuel combustion process suffices to sustain acceleration and the starting engine is turned off. The compressor accepts filtered air from the environment and passes it through 17 stages. Bleed valves bypass some part of the compressor air during startup to stabilize the turbine acceleration and to avoid the surge phenomenon. Inlet Guide Vanes at the front of the compressor are modulated to regulate the inlet air flow to improve GT performance. The combustor burns natural gas with the proper air mixture to provide hot gases for the two-stage power turbine. The fuel throttle valve is controlled to develop speed for the GT first and megawatt generation later, both within appropriate turbine exhaust temperature limits.

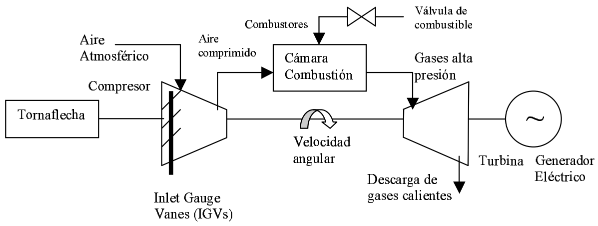

Figure 2: A basic schema of a gas turbine.

Basically, the GT control system strategy consists of the fuel flow and the air flow regulation according to each specific operating phase, while keeping the acceleration, temperature and compressor discharge pressure under secure operating conditions. However it does not take into account that the commands sent to actuators or motors could not have the desired effect, or that valves could stoke. The action policy of the control system do not always consider the optimal actions, in particular during disturbances. We believe that decision-theoretical planning methods can deal well with these problems.

4 Preliminary Results

For the experiments, we used a gas turbine simulator. A set of 21 continuos variables were considered as a simplified model of the gas turbine. From the values of these variables under different conditions, a DBN representation of the transition model was obtained using K2 as implemented in *Elvira* [2]. The structure of the resulting model is coherent with intution, and most of the dependencies correspond to what are known functional dependencies in a gas turbine.

To learn the r-actions for velocity control, we represented the state variables as qualitative changes between consecutive times. Then we used C4.5 to induce the r-actions, which are represented as Prolog rule. In total, we obtained 25 clauses. Some of the r–actions obtained with c4.5 are shown in fig. 3, including the relative precision of each rule.

The state variables that are relevant for each action are in the ones included in the clause body, so these can be used to learn a transition model for relational states.

5 Conclusions and future work

We propose an alternative form of abstract MDP representation, where state variables are qualitative proportionality predicates and actions are STRIPS-like operators. Based on this representations, we use learning techniques to extract the action and the transition models. K2 and EM are used to learn

```
increaser(vel, vele):-
        increase(dem_val_control_gas, dem_val_control_gase). [95%]
increaser(vel, vele):-
        decrease(signal_ctrol_vel, signal_ctrol_vele ),
        increase(temp_gases_descarga_1, temp_gases_descarga_1e),
        increase(valv_ratio_paro_vel,, valv_ratio_paro_vele ). [88.2%]
no_change((vel, vele):-
        decrease(signal_ctrol_vel, signal_ctrol_vele ),
        decrease(temp_gases_descarga_1, temp_gases_descarga_1e),
        increase(dem_val_control_gas, dem_val_control_gase),
        no_change(pres_descarga_compresor, pres_descarga_compresore),
        increase(valv_ratio_paro_vel,, valv_ratio_paro_vele ). [87%]
no_change((vel, vele):-
        no_change(signal_ctrol_vel, signal_ctrol_vele ),
        increase(temp_gases_descarga_2, temp_gases_descarga_2e),
        increase(dem_val_control_gas, dem_val_control_gase). [80%]
```

Figure 3: Examples of r-actions determined by C4.5.

the structure and parameters of the a dynamic Bayesian network that specifies the transition function, and C4.5 to obtain the relational predicates that represent the actions. We present initial results of an experiment with a simulator of a turbogas power plant, for which we obtained the transition model as a DBN and the actions in a relational representation.

As future work, we will induce qualitative decision trees using QUIN [10] to represent the utility functions. We also plan to learn the transition model on the qualitative states. Finally, we will develop a solution for MDPs based on this representation.

References

[1] Kuipers B. Qualitative simulation. *Artificial Intelligence*, 29:289–338, 1986.

[2] Elvira Consortium. Elvira: an environment for creating and using probabilistic graphical models. Technical Report PGM-02, Elvira Consortium, Spain, 2002.

[3] Bellman R. E. *Dynamic Programming.* Princeton University Press, Princeton, N.J., U.S.A., 1957.

[4] K. Forbus. Qualitative process theory. *Artificial Intelligence*, 24:85–168, 1984.

[5] J. Hoey, St-Aubin R., A. Hu, and Boutilier C. Spudd: Stochastic planning using decision diagrams. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence, UAI-99*, pages 279–288, 1999.

[6] Daphne Koller and Avi Pfeffer. Probabilistic frame-based systems. In *Proc. National Conf. on Artificial Intelligence*. AAAI, 1998.

[7] Eduardo Morales. A relational representation for reinforcement learning. Technical report, Computer Science and Engineering, University of New South Wales, Sidney, Australia, 2001.

[8] David. Poole. Probabilistic horn abduction and bayesian networks. *Artificial Intelligence*, 64:81–129, 1993.

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

[10] D. Suc and Ivan Bratko. Qualitative reverse engineering. In *Proceedings of the 19th International Conference on Machine Learning*, 2000.

[11] Dean T. and Kanazawa K. A model for reasoning about persistence and causation. *Computational Intelligence*, 5(3):142–150, 1989.