Anticipatory Learning Classifier Systems and Factored Reinforcement Learning

Olivier Sigaud and Thomas Degris

Université Pierre et Marie Curie - Paris6
Institut des Systèmes Intelligents et de Robotique (ISIR), CNRS FRE 2507,
4 place Jussieu, F-75005 Paris, France
Olivier.Sigaud@isir.fr, thomas.degris@laposte.net

Abstract. In this contribution we compare Anticipatory Learning Classifier Systems and Reinforcement Learning methods designed to solve Factored Markov Decision Problems when the structure of the problem is not known in advance. In particular, we focus on two instances, namely MACS and SPITI.

1 Introduction

Learning Classifier Systems (LCSs) [1] are rule-based systems where the rules (called classifiers) are learnt from experience and benefit from a generalization capability to build a compact representation of the problem. Standard LCSs such as XCS [2] use condition-action classifiers and combine Reinforcement Learning (RL) methods with Genetic Algorithms (GAs) to learn a compact set of classifiers. Anticipatory Learning Classifier Systems (ALCSs) [3] deviate from the classical LCS framework on one fundamental point. Instead of [Condition] → [Action] classifiers, they manipulate [Condition] [Action] → [Effect] classifiers. The [Effect] part represents the expected effect (next state or set of next variables) of the action in all situations that match the [Condition] part of the classifier. Such a set of classifiers constitutes what is called in the RL literature a *model of transitions*. ALCSs are an instance of model-based RL architecture, a category of systems whose prototype is the DYNA architecture [4]. Indeed, ALCSs can be seen as combining two crucial properties of RL systems. Like DYNA architectures, they learn a model of transitions, which endows them with anticipation and planning capabilities and speeds up the learning process. Like classical LCSs, they are endowed with a generalization property, which lets them build much more compact models than tabular DYNA architectures [5].
In RL literature, [6] introduced the Factored Markov Decision Processes (FMDPs) framework to represent large and structured MDPs compactly. In this approach, a state is implicitly described by an assignment of values to some set of state variables, a representation which shares strong similarities with the one used in LCSs where the variables are named "attributes". But the structure of the model of transitions is assumed to be known in FMDPs, by contrast with the ALCS framework where this structure is learnt from experience.

Recently, [7, 8] proposed SDYNA, an approach to learn from experience the structure of the model of transitions in the FMDP framework, giving rise to Factored Reinforcement Learning (FRL), a short name for RL in FMDPs when the structure of the FMDP is not known in advance.
The goal of this contribution is to compare at the level of concepts and implementation one instance of ALCS named MACS and one instance of SDYNA named SPITI to highlight the superiority of the second framework. We briefly present the two systems in section 2, compare them in section 3 and conclude in section 4.

2 SPITI and MACS

A FMDP is described by a set of state variables $S = \{X_i...., X_n\}$, where each X_i takes value in a finite domain $Dom(X_i)$. A state $s \in S$ assigns a value $x_i \in Dom(X_i)$ to each state variable X_i. The state transition model T_a of an action a can be represented as a set of decision trees giving the probability distribution of each x_i given the values of all state variables at the previous time step. Given this representation of the model of transitions, structured dynamic programming algorithms such as SVI, SPI [9] and SPUDD [10] can be shown to converge to the optimal policy. They do so very efficiently.
SDYNA is a structured version of the DYNA architecture where the model of transitions and of reward are learnt from experience as a collection of decision trees using the ITI algorithm [11]. Diverse structured dynamic programming or linear programming algorithms can then be applied to this model. SPITI is a particular instance of SDYNA calling upon an incremental version of SVI to perform dynamic programming.
Among ALCSs, a comparison with SPITI is easier with MACS for two reasons:

- in MACS, the [Effect] part of classifiers anticipates the value of only one variable at a time for each action, thus one classifier is equivalent to one decision tree in the model of transitions of SPITI.
- MACS is the only ALCS that does not call upon a GA. Instead, to learn the model of transitions, it relies on the combination of a generalization heuristics and a specialization heuristics that collaborate to converge towards a compact and accurate model of transitions. This can be compared more easily than a GA to ITI which relies on an information metrics to grow a decision tree incrementally.

Though MACS converges faster and with a more compact model than other ALCSs [12], it has major weaknesses in the action selection mechanism. In particular, MACS does not generalize the models of the reward and the value functions. Instead, these models are represented by a table giving a value for each encountered state, which prevents using it for very large state space problems. By contrast, Butz [13] proposed XACS which also generalizes the value function in ACS2 [14], by using XCS.
Thus, if a performance comparison was to be done between SPITI and ALCSs, one would like to perform that comparison with an ideal combination of XACS and MACS that does not exist so far.

3 Similarities and differences

Both MACS and SPITI call upon a model-based RL process and are endowed with a generalization property that make them able to address large MDPs without prior knowledge of the structure. Their representations of the model of the transitions have a similar global structure, the value of each variable being anticipated separately for each action in function of variables defining the previous state of the model. But MACS and SPITI differ in several points:

- In MACS, building a compact model of the transition function relies on a complex combination of heuristics whereas SPITI calls upon the well established incremental decision tree induction ITI algorithm [11] and theχ^2 information metrics.
- In MACS, the dynamic programming component of model-based RL is applied to a flat representation of states, whereas SPITI calls upon SVI, SPI or SPUDD to perform this computation compactly, with guarantees of convergence to optimality as far as the model of transition is supposed perfectly accurate and very efficiently in practice.
- In MACS, the model of transitions is represented as a set of classifiers and the value function is tabular, whereas SPITI implements the model of transitions, the value function and the policy as decision trees which results in a much faster algorithmic access to the information.

All these differences speak in favor of SPITI against MACS. As a matter of fact, though we have never done any direct performance comparison, SPITI was shown to perform very well on problems that are out of reach of MACS. In [7, 8], we have shown that SPITI could quickly find a near optimal policy on problems with up to 10^6 actually reachable states while building a nearly optimally compact model of the transition and reward functions. Another instance of SDYNA based on linear programming was even able to tackle problems with more than 10^{12} states [15]. More recently, we have used SPITI on a complex non-Markov case study where the problem consists in learning an efficient behaviour to a bot playing Counter-Strike. A video of the result can be seen at `http://animatlab.lip6.fr/KodabotEn`.

4 Conclusion

The goal of this contribution was to show that ALCSs and FRL systems such as SPITI are conceptually very similar and share interesting properties, but FRL systems are mathematically better grounded and benefit from very efficient implementations. An extension of this contribution would consist in performing empirical comparisons between SPITI and MACS on large size benchmarks and in comparing SPITI with other ALCSs based on a GA such as XACS.

References

1. Sigaud, O., Wilson, S.W.: Learning classifier systems: a survey. Journal of Soft Computing **11**(11) (2007) 1065–1078
2. Butz, M., Kovacs, T., Lanzi, P.L., Wilson, S.W.: Toward a theory of generalization and learning in xcs. IEEE Transactions on Evolutionary Computation **8**(1) (2004) 28–46
3. Butz, M.V.: Anticipatory Learning Classifier Systems. Kluwer Academic Publishers, Boston, MA (2002)
4. Sutton, R.S.: Planning by incremental dynamic programming. In: Proceedings of the Eighth International Conference on Machine Learning, San Mateo, CA, Morgan Kaufmann (1990) 353–357
5. Gérard, P., Sigaud, O.: Designing efficient exploration with MACS: Modules and function approximation. In: Proceedings of the Genetic and Evolutionary Computation Conference 2003 (GECCO'03), Chicago, IL, Springer-Verlag (july 2003) 1882–1893
6. Boutilier, C., Dearden, R., Goldszmidt, M.: Exploiting structure in policy construction. In: Proceedings of the 14th International Joint Conference in Artificial Intelligence. (1995) 1104–1111
7. Degris, T., Sigaud, O., Wuillemin, P.H.: Chi-square tests driven method for learning the structure of factored MDPs. In: Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence, Massachusetts Institute of Technology, Cambridge, AUAI Press (2006) 122–129
8. Degris, T., Sigaud, O., Wuillemin, P.H.: Learning the structure of factored markov decision processes in reinforcement learning problems. In: Proceedings of the 23rd International Conference in Machine Learning, ACM, Pittsburgh, Pennsylvania (2006) 257–264
9. Boutilier, C., Dearden, R., Goldszmidt, M.: Stochastic dynamic programming with factored representations. Artificial Intelligence **121**(1-2) (2000) 49–10
10. Hoey, J., St-Aubin, R., Hu, A., Boutilier, C.: SPUDD: Stochastic Planning using Decision Diagrams. In: Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann (1999) 279–288
11. Utgoff, P.E.: Incremental induction of decision trees. Machine Learning **4** (1989) 161–186
12. Gérard, P., Meyer, J.A., Sigaud, O.: Combining latent learning with dynamic programming in MACS. European Journal of Operational Research **160** (2005) 614–637
13. Butz, M.V., Goldberg, D.E.: Generalized state values in an anticipatory Learning Classifier System. In Butz, M.V., Sigaud, O., Gérard, P., eds.: LNAI 2684: Anticipatory Behavior in Adaptive Learning Systems. Springer-Verlag (2003) 281–301
14. Butz, M.V., Goldberg, D.E., Stolzmann, W.: The Anticipatory Classifier System and Genetic Generalization. Natural Computing **1**(4) (2002) 427–467
15. Degris, T., Sigaud, O., Wuillemin, P.H.: Apprentissage par renforcement exploitant la structure additive des mdp factorisés (in french). In: Actes de la conférence JFPDA'07. (2007) 49–60