Combining Anticipation and Dynamic Programming in Classifier Systems

Pierre Gérard[1,2] and Olivier Sigaud[1]

[1] Dassault Aviation, DGT/DPR/DESA
78, Quai Marcel Dassault, 92552 St-Cloud Cedex
[2] AnimatLab (LIP6), 8, rue du capitaine Scott, 75015 PARIS
pierre.gerard@lip6.fr, olivier.sigaud@dassault-aviation.fr

This paper presents our work on the use of anticipation in Classifier Systems applied to Markov problems. It calls upon classifiers with a *[Condition]*, an *[Action]* and an *[Effect]* part. We discuss the generalization problem raised by our experiments.

1 Introduction

Our work takes place in the Learning Classifier Systems framework. We model an agent which gets a perceptions and rewards from the environment, and acts on it. In this framework, our basic assumptions are the following:

– rather than generating new classifiers with random genetic operators, we drive the classifier discovery process by experience;
– rather than using a plain reinforcement learning process, the agent performs *latent learning* by using anticipation capabilities.

2 Our algorithm

Like [Butz et al., 2000,Stolzmann, 1998], the system we designed uses classifiers composed of three parts. The classifier can be only fired if the *[Condition]* part matches the current perception. The *[Action]* part defines how the system acts on its environment when the classifier is fired. The *[Effect]* part anticipates the perception resulting from the action of the classifier if it is fired.

Each classifier is also given a quality which is used to compute an ϵ-greedy policy. In most cases, if several classifiers are matching the current perception, the system will choose to perform the action of the classifier with the highest quality.

The learning process differs from [Butz et al., 2000,Stolzmann, 1998] and [Witkowski, 1999]. It involves two complementary processes:

– the *latent learning* process discovers reliable classifiers modeling the dynamics of the environment ;
– the *reinforcement learning* process takes advantage of this model of the dynamics of the environment. It uses a Dynamic Programming algorithm. It estimates immediate rewards and uses the state transition information provided by the *[Effect]* parts to compute the qualities of the classifiers.

The latent learning itself involves into two distinct processes :

- the *[Effect]* parts are adjusted by comparing successive perceptions;
- the system uses a careful specialization process driven by experience to discover *[Condition]* parts with an adequate level of generalization.

3 The generalization problem

In classifier systems without *[Effect]* part [Wilson, 1995,Lanzi, 2000], a classifier is kept when it helps to maximize the payoff on the long run. When the system performs latent learning, the decision of keeping or removing a classifier only relies on its ability to predict the next perceptions. It does not take action optimality into account at all.

This way of considering the fitness of a classifier gives rise to a new way of considering generalization. A classifier is too general if a joker token in its *[Condition]* part prevents the anticipation to be accurate, regardless of the payoff. It is too specialized when its anticipation ability would remain accurate even if some joker were added in its *[Condition]* part, regardless of the payoff.

Some classifier may have the right degree of generalization with respect to the anticipation, and may prevent to use Dynamic Programming methods like Value Iteration. Such classifiers anticipate well but match with several different perceptions. Since Value Iteration updates the quality of the classifier regardless of the underlying state, they introduce perceptual aliasing.

So, Dynamic Programming methods seem inadequate for a system which uses both anticipation and generalization. A straight-forward solution is to use lookahead planning rather than a variety of the Value Iteration algorithm. But as a result, the system may suffer from a lack of reactivity, and we must deal with the reactivity/planning tradeoff.

References

[Butz et al., 2000] Butz, M. V., Goldberg, D. E., and Stolzmann, W. (2000). Investigating generalization in the anticipatory classifier system. Technical report, Illinois Genetic Algorithm Laboratory, University of Illinois at Urbana-Champain. Available at ftp://ftp-illigal.ge.uiuc.edu.

[Lanzi, 2000] Lanzi, P. L. (2000). Toward optimal performance in classifier systems. *Evolutionary Computation Journal*. in print.

[Stolzmann, 1998] Stolzmann, W. (1998). Anticipatory classifier systems. In Koza, J., Banzhaf, W., Chellapilla, K., Deb, K., Dorigo, M., Fogel, D., Garzon, M., Goldberg, D., Iba, H., and Riolo, R., (Eds.), *Genetic Programming*. Morgan Kaufmann Publishers, Inc., San Francisco, CA.

[Wilson, 1995] Wilson, S. W. (1995). Classifier fitness based on accuracy. *Evolutionary Computation*, 3(2):149–175.

[Witkowski, 1999] Witkowski, C. M. (1999). Integrating unsupervised learning, motivation and action selection in an a-life agent. In Floreano, D., Mondada, F., and Nicoud, J.-D., (Eds.), *5th European Conference on Artificial Life (ECAL-99)*, pages 355–364, Lausanne. Springer.