Lookahead and Latent Learning in ZCS

Larry Bull
Faculty of Computing, Engineering & Mathematical Sciences
University of the West of England
Bristol BS16 1QY
larry.bull@uwe.ac.uk

Abstract

Learning Classifier Systems use reinforcement learning, evolutionary computing and/or heuristics to develop adaptive systems. This paper extends the ZCS Learning Classifier System to improve its internal modelling capabilities. Initially, results are presented which show performance in a traditional reinforcement learning task incorporating lookahead within the rule structure. Then a mechanism for effective learning without external reward is examined which enables the simple learning system to build a full map of the task. That is, ZCS is shown to learn under a latent learning scenario using the lookahead scheme. Its ability to form maps in reinforcement learning tasks is then considered.

1 INTRODUCTION

Traditional Learning Classifier Systems (LCS) [Holland 1976] use genetic algorithms (GA) [Holland 1975] and the bucket brigade algorithm [Holland 1986] to produce an interacting ecology of rules for a given task. Holland et al. [1986] proposed a number of mechanisms by which LCS could potentially realise many complex inductive processes. However, the basic architecture was difficult to use and understand. Wilson [1994] presented ZCS which "keeps much of Holland's original framework but simplifies it to increase understandability and performance" [ibid.]. Bull and Hurst [2002] have recently shown that, despite its relative simplicity, ZCS is able to perform optimally through its use of fitness sharing. That is, ZCS was shown to perform as well, with appropriate parameters, as the more complex XCS [Wilson 1995] on a number of tasks. The significant difference between the two systems being XCS's ability to build a complete, maximally general map of the given problem.

In this paper the basic ZCS architecture is extended to include mechanisms by which cognitive capabilities, along the lines of those envisaged by Holland et al. [1986], can emerge; the use of predictive modelling within ZCS is considered through an alteration to the rule structure. Using a maze task loosely based on that of early animal behaviour experiments, it is found that ZCS can learn effectively when reward is dependent upon the ability to accurately predict the next environment state/sensory input. This ZCS with lookahead is then extended to work under latent learning. That is, an approach is presented which allows ZCS to build a full map of its task without external reinforcement. This result is then suggested as significant for traditional payoff-based LCS when the aforementioned difference to XCS is considered.

The paper is arranged as follows: the next section briefly describes ZCS. Section 3 considers the use of lookahead in general and presents results from its use within ZCS. In Section 4 the use of latent learning with the predictive form of ZCS is presented. Finally, all findings are discussed.

2 ZCS

ZCS is a "Zeroth-level" Michigan-style Classifier System without internal memory, where the rule-base consists of a number (N) of condition/action rules in which the condition is a string of characters from the usual ternary alphabet {0,1,#} and the action is represented by a binary string. Associated with each rule is a fitness scalar which acts as an indication of the perceived utility of that rule within the system. This fitness of each rule is initialised to a predetermined value termed S_0.

Reinforcement in ZCS consists of redistributing fitness between subsequent "action sets", or the matched rules from the previous time step which asserted the chosen output or "action". A fixed fraction (β) of the fitness of each member of the action set ([A]) at each time-step is placed in a "common bucket". A record is kept of the previous action set $[A]_{-1}$ and if this is not empty then the members of this action set each receive an equal share of the contents of the current bucket, once this has been reduced by a predetermined discount factor (γ). If a reward is received from the environment then a fixed fraction (β) of this value is distributed evenly amongst the members of [A]. Finally, a tax (τ) is imposed on all matched rules that do not belong to [A] on each time-step in order to encourage exploitation of the stronger classifiers. Wilson notes that this is a change to the traditional LCS bucket-brigade algorithm [Holland 1986] since there is no concept of a rule 'bid', generalisation is not considered explicitly, sets of rules are updated and the pay-back is reduced by 1-γ on each step (see [Bull & O'Hara 2001] for related discussions).

ZCS employs two discovery mechanisms, a panmictic GA and a covering operator. On each time-step there is a probability *p* of GA invocation. When called, the GA uses roulette wheel selection to determine two parent rules based on fitness. Two offspring are produced via mutation (probability μ, probability of inserting a wildcard $p_{\#}$) and crossover (single point with probability χ). The parents then donate half of their fitnesses to their offspring who replace existing members of the rule-base. The deleted rules are chosen using roulette wheel selection based on the reciprocal of rule fitness. If on some time-step, no rules match or all matched rules have a combined fitness of less than ϕ times the rule-base average, then a covering operator is invoked.

The default parameters presented for ZCS, and unless otherwise stated for this paper, are: $N = 400$, $S_0=20$, $\beta = 0.2$, $\gamma = 0.71$, $\tau = 0.1$, $\chi = 0.5$, $\mu = 0.002$, $p = 0.25$, $\phi = 0.5$, $p_{\#}=0.33$.

3 LOOKAHEAD

Holland [1990] presented a general framework for incorporating future state predictions into LCS, termed lookahead (after [Samuel 1959]). Lookahead allows a learning entity to construct an internal model of its environment, " ... a matter of implementing the rule 'IF the environment is in state S AND action A is taken THEN (the system expects) state S2 will occur'" [Holland 1990]. Under Holland's scheme tags, which (potentially) facilitate rule coupling under normal operation [Holland 1986], would be used rather than an explicit representation of the expected environmental state. A "virtual" bucket brigade algorithm would then pass payoff back through "cones" of likely future outcomes to influence the action selection process on any given step. Stolzmann [1998] has presented a heuristic-driven LCS, ACS, which uses the explicit next-state rule structure noted above to build anticipatory models of an environment. The accuracy of the rules' predictions are factored into their utility. Later work added a GA which used this measure for rule fitness resulting in improved performance [e.g. Butz et al. 2000]. LCS which use rule-linkage over succeeding timesteps [e.g. Tomlinson & Bull 1998] also implicitly build predictions of future states; the condition of a linked rule must represent the next environmental state after the action of its predecessor is taken.

3.1 THE APPROACH

In this paper, as in [Riolo 1991][Stolzmann 1998][Gerard & Sigaud 2001] and suggested in [Wilson 1995], an explicit representation of the expected next environmental state is used. That is, rules are of the general form:

<condition> : <action> : <anticipation>

Generalizations (#'s) are allowed in the condition and anticipation strings. Where #'s occur at the same loci in both, the corresponding environmental input symbol "passes through" such that it occurs in the anticipated description for that input. Similarly, defined loci in the condition appear when a # occurs in the corresponding locus of the anticipation. Each rule also maintains the usual fitness parameter as in ZCS.

One further mechanism is incorporated: the first *N* random rules of the rule-base have their anticipation created using cover (with #'s included as usual) in the first [A] of which they become a member. This goes some way to make " ... good use of the large flow of (non-performance) information supplied by the environment." [Holland 1990] and can be seen to create a supervised learning task during initialization (this is particularly significant in Section 5). Rules created under the cover operator also receive this treatment. In this way the GA explores the generalization space of the anticipations created by the simple heuristic (as opposed to [Stolzmann 1998]).

All other system functionality is as described in Section 2, except that members of a given [A] only receive bucket payments if they correctly predicted the next state. Hence, in effect, incorrect predictors are taxed at the learning rate. Predictions are not tested for external reward receiving rules.

3.2 THE TASK

The aim of this paper is to show that ZCS can be extended to exploit lookahead and latent learning to build a more comprehensive internal model of the task. The general motivation for such work with machine learning algorithms comes, in part, from experiments undertaken with rats by Tolman [e.g. see Mackintosh 1974], Seward [1949] and others. It was shown that rats appear able to construct internal models of simple mazes of the general form shown in Figure 1 so that, when later placed at the start (lowest cell), they would find the food via the shortest route. This will be returned to in Section 4, a goal-directed only version being examined here.

In this section the task is seen as the well-known animat problem [Wilson 1987]. As such, ZCS is used to develop the controller of a simulated robot/animat which must traverse the maze in search of food. It is positioned randomly in one of the blank cells and can move into any one of the surrounding eight cells on each discrete time step, unless occupied by a tree. If the animat moves into the food cell the system receives a reward from the environment (1000), and the task is reset, i.e. food is replaced and the animat randomly relocated.

On each time step the animat receives a sensory message which describes the eight surrounding cells. The message is encoded as a 16-bit binary string with two bits representing each cardinal direction. A blank cell is represented by 00, food (F) by 11 and trees (T) by 10 (01 has no meaning). The message is ordered with the cell directly above the animat represented by the first bit-pair, and then proceeding clockwise around the animat.

The trial is repeated 10,000 times and a record is kept of a moving average (over the previous 50 trials) of how many

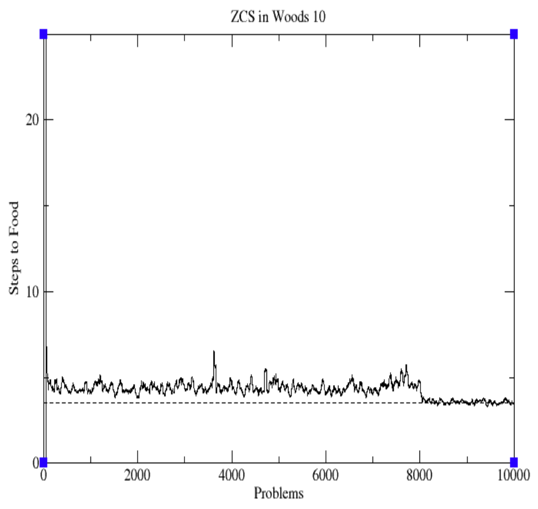
T	T	T	T	T	T	T
T	F	T	T	T		T
T		T	T	T		T
T						T
T	T	T		T	T	T
T	T	T		T	T	T
T	T	T	T	T	T	T

Figure 1: The Woods 10 environment.

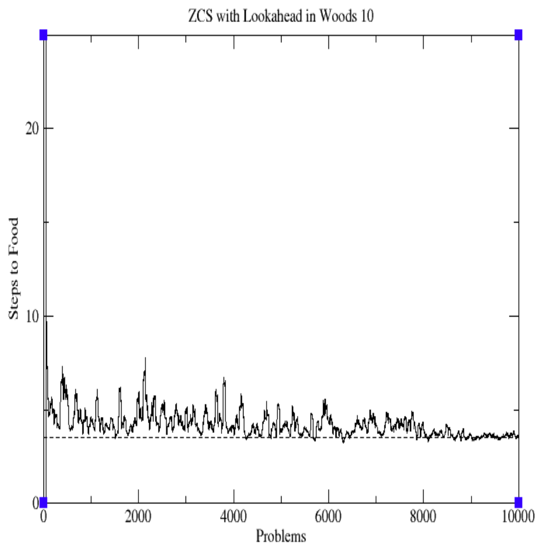
steps it takes for the animat to move into a food cell on each trial. Optimal performance is 3.5 steps to food. All results presented are the average of ten runs.

3.3 RESULTS

Figure 2(a) shows the performance of standard ZCS in Woods 10 with the same parameters as those in Section 2, except β=0.45. Optimal performance can be seen during the last 2000 trials where the GA was disabled and a deterministic action selection scheme used such that the action with the highest total fitness in [M] was always chosen (after [Bull & Hurst 2002]). Figure 2(b) shows the performance of ZCS using similar parameters but with the lookahead rule structure and scheme described above (γ=0.4, ρ=0.45). It can be seen that performance is equivalent and hence that ZCS is able to produce accurate next-state predictions. However, ZCS does not form a full state-action-anticipation map under reinforcement learning. The next section presents a mechanism by which such a map can be constructed under latent learning.



(a)



(b)

Figure 2: Performance of ZCS in Woods 10 and incorporating lookahead.

4 ZCSL: USING LATENT LEARNING WITH LOOKAHEAD

As noted above, one motivation for exploring the use of learning without external reinforcement comes from early experiments in animal behaviour. Typically, rats were allowed to run around a maze of the shape shown in Figure 1 where the food cell would be empty but a different colour to the rest of the maze. The rats would then be fed in the marked location. Finally, the rats were placed at the start location and their ability to take the shortest path, i.e. go left at the T-junction in Figure 1, recorded. It was found that rats could do this with around 90% efficiency. Those which were not given the prior experience without food were only 50% efficient, as expected.

Riolo [1991] extended a version of Holland's LCS to consider such learning without external reinforcement, using the same general rule form as that above and tags. The bucket brigade was then altered to consider the accuracy of rule's predictions of future states. Although Riolo did not incorporate rule discovery, he showed his CFCS2 could learn and exploit internal models to solve a version of the maze task described above. Both ACS [Stolzmann & Butz 2000] and the related YACS[Gerard & Sigaud 2001] have also been shown able to develop internal models under latent learning using heuristics.

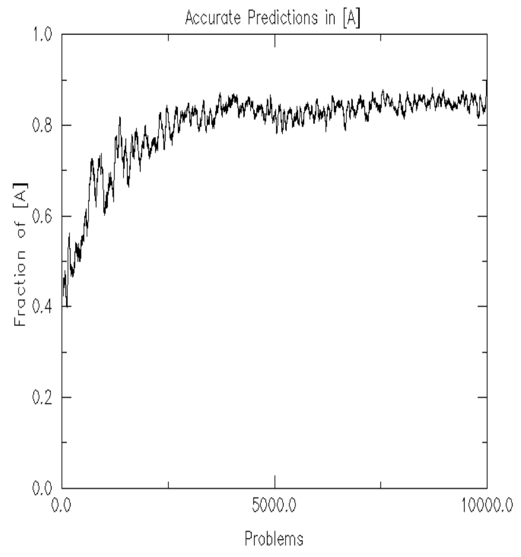
4.1 THE APPROACH

The *internal model building task can be cast as a single-step task*. In the simplest case, using lookahead under latent learning and the above rule structure, a single-step learning task is created whereby reward is given only if a rule predicts the expected outcome of taking its action under the condition matched. This is the approach used in ZCSL.

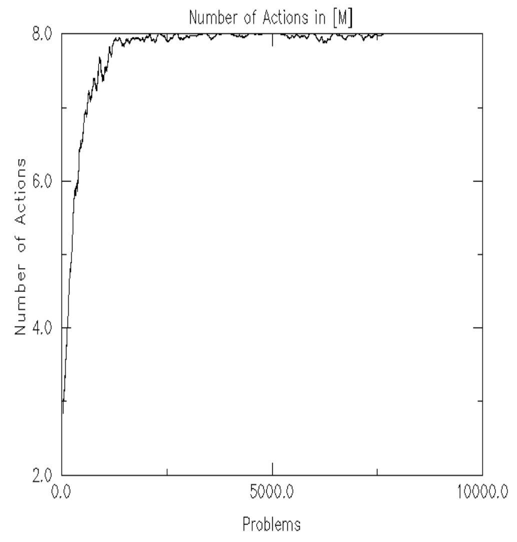
At the beginning of a trial, the animat is placed randomly in the maze. A matchset is formed and an action chosen at random. All rules which propose the chosen action form [A] and pay β of their fitness into the bucket as usual. All rules in [A] then construct their anticipated sensory input for the next state, i.e. pass-through is considered, and the action is taken. Each rule in [A] which correctly predicts the next state is rewarded with payoff 1000 divided by the number of correct rules in [A]; fitness sharing is used. Note that taxing the other members of the matchset is no longer appropriate as a full map of actions is required. This process is repeated for ten steps and then the animat is randomly replaced in the maze. It is important to note that although the animat can sense the food, it is not able to move onto that cell (but predictions are tested as if it had). All other operations are as before.

4.2 RESULTS

Figure 3 shows the behaviour of this form of ZCS in the Woods 10 maze. The parameters used were the same as those given in Section 2, except N=800, and the GA is again turned off for the last 2000 trials.



(a)



(b)

Figure 3: Showing the performance of ZCSL in the Woods 10 environment.

Figure 3(a) shows the fraction of rules of a given action set which correctly predicted the next environmental state. It can be seen that by around 5000 trials 80% (i.e. the majority) of the rules in each action set were accurate predictors. Figure 3(b) shows the number of actions represented in each matchset has risen to eight around the same time. Hence after 5000 trials (50,000 cycles) *ZCSL has constructed a full and accurate map of the maze*, assuming the most numerous anticipated next state of a given [A] is used. Note that ZCSL includes state-action pairs which lead to no change in stimulus without the explicit consideration of such circumstances. The original ACS did not develop rules for such cases, but was later modified [Stolzmann 2000] (see also YACS).

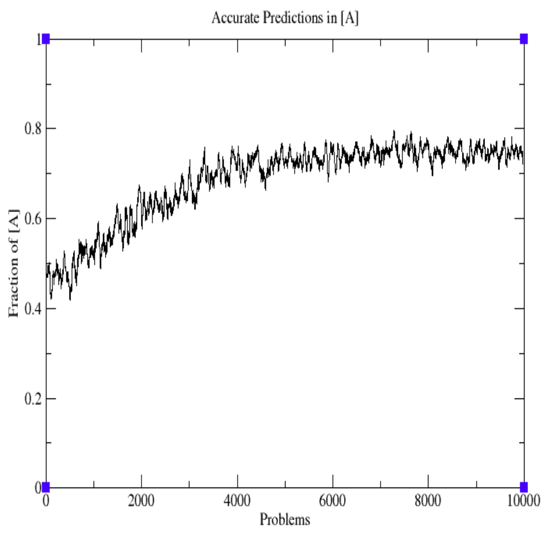
T	T	T	T	T	T	T	T	T
T						T	F	T
T			T		T	T		T
T		T						T
T				T	T			T
T		T		T				T
T		T						T
T						T		T
T	T	T	T	T	T	T	T	T

Figure 4: Maze 6.

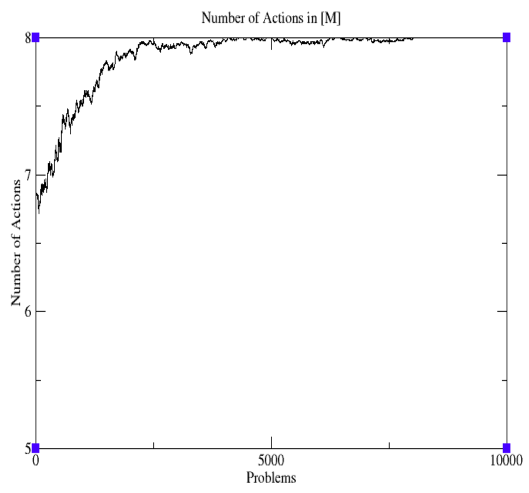
Therefore, the resulting LCS systems could be used to reproduce the rat experiments through any number of planning techniques, such as breadth-first search: from a given input the most numerous anticipation of each [A] can be considered to represent the next environmental input, and so on, until the goal state is seen. If this is done from the start location, firing the appropriate sequence of rules would give 100% efficiency at the task. That is, the ZCSL controllers have the ability to reproduce the general behaviour of the rats using a very simple LCS architecture. The following four rules show an example solution found in this way starting from the bottom cell (GA fitnesses not shown):

```
#0##1#101#1#1010 : N    : #000101000#01#00
0000101000101000 : NW : 1#1#0#001010000#
####10100#0#101# : NW : #0###0#0#0#0#010
#1#01#00#010#0#0 : N    : ##10#0#0##10#010
```

It can be noted that both generalization and pass-through are contained in the solutions although no explicit pressure for either exists within the simple system.



(a)



(b)

Figure 5: Showing the performance of ZCSL in the Maze 6 environment.

ZCSL has also been applied to the more challenging Maze 6 [Lanzi 1997] environment (Figure 4). The parameters used were the same as those above, except N=10,000. Again, the animat could not move onto the food cell. Figure 5(a) shows the fraction of rules of a given action set which correctly predicted the next environmental state. It can be seen that by around 5000 trials the majority of each action set (>75%) were accurate predictors. Figure 5(b) shows the number of actions represented in each matchset has again risen to eight around the same time. Hence, after 5000 trials ZCSL has constructed a full and accurate map of Maze 6.

5 DISCUSSION: XCS

As noted in the introduction, XCS attempts to build a full, non-overlapping, maximally general map of the problem space which can have advantages over traditional payoff-based LCS, such as ZCS, if *a posteri* explanatory power is required, as in data mining for example.

There seems no reason why the above modified ZCS system of Section 4 cannot also produce such predictive maps, particularly for single step tasks. That is, *if the anticipation is altered from being an expected environmental state to a numerical payoff value, a similar map can be formed.* I.e. rules are of the general form:

<condition> : <action> : <anticipated payoff>

This version has been explored using the well-known multiplexer task. These Boolean functions are defined for binary strings of length $l = k + 2^k$ under which the first k bits index into the remaining 2^k bits, returning the value of the indexed bit. A correct classification results in a payoff of 1000, otherwise 0.

All system functionality is as described in Section 4 except that the appropriate value (1000 or 0) is written on as the anticipation for a randomly created rule. Mutation causes a change in the value to another valid payoff level (1000 to 0, or vice versa here). A trial is two evaluations here.

Figure 6 shows the percentage of correct predictions in a given [A] and number of actions in [M] for the thirty seven bit multiplexer, using the same parameters as for Woods 10 except N=5000, β=0.8 and $p_{\#}$=0.8. It can be seen that, on average, correct predictors occur with highest numerosity and that both actions are present in each matchset after around 700,000 trials (1,400,000 evaluations). The evolved rules for the first two data lines from an example solution were as follows (GA fitnesses not shown):

```
000000############################### : 0 : 1000
000000############################0## : 1 : 0
000001###########################0### : 0 : 0
000001############################### : 1 : 1000
00001#0############################## : 0 : 1000
00001#0##0###########0############0## : 1 : 0
00001#1########0##################### : 0 : 0
00001#1############################## : 1 : 1000
```

It can again be noted that solutions are very general although no explicit pressure for this exists within the simple system. Results show that, the longer the system is left to run, the more general rules become.

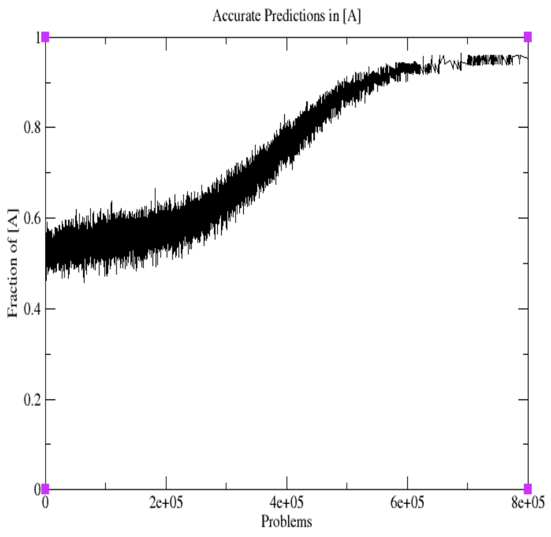
Butz et al. [2001] have recently examined the behaviour of XCS on a number of multiplexer problems, also solving the thirty seven bit version. They note that a stronger distinction between rule accuracies was required in comparison to the smaller multiplexer tasks and a larger $p_{\#}$. ZCSL also appears to need a strong pressure toward accurate predictors through an increase in the learning rate since this is also essentially the tax rate for erroneous rules. A large $p_{\#}$ also proved important. Butz et al. did not manage to solve the equivalent seventy bit task and initial attempts with ZCSL have also proven unsuccessful. This remains open to future investigation.

Hence this system uses a simple heuristic to promote accuracy in payoff predictions whilst the GA with fitness sharing apportions resources and encourages generalization. In contrast, XCS uses a four-step fitness update to promote accuracy in payoff predictions, an explicit replacement strategy to apportion (balance) resources and a triggered niche GA to encourage generalization. Whether the simple approach described here scales as well as XCS to tasks with more classes and prediction levels, noisy data, or can be used in multi-step tasks represents future work. The use of some of XCS's other features (subsumption, action set filling, etc.) may help.

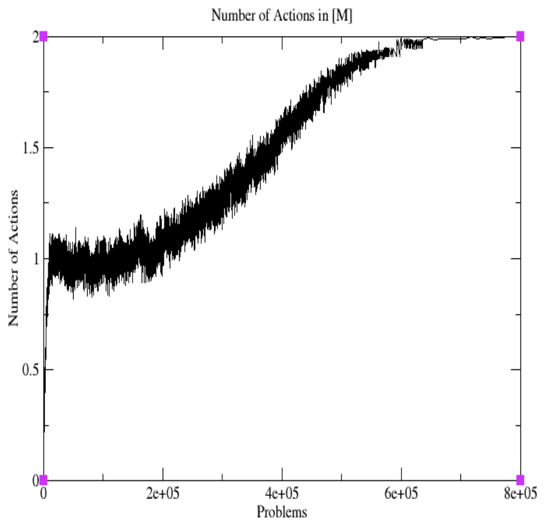
6 CONCLUSIONS

In this paper ZCS has been extended to incorporate lookahead and latent learning. Using a simple maze task, based on those used in early animal behaviour experiments, it has been shown that ZCS can build *partial* internal models under traditional goal-directed learning. The construction of a *full* internal environment model under latent learning with lookahead was then cast as a single-step reinforcement task and ZCSL was shown able to form accurate maps under fitness sharing. Future work will examine the inclusion of other mechanisms, such as an explicit unchanging component [e.g. Stolzmann 2000], to improve performance. Other schemes to encourage maximal generality within solutions will also be explored.

The use of the mechanisms within a more complex framework to exploit internal models during learning under reinforcement, after Sutton's Dyna [e.g. Sutton 1990] (see also [Donnart & Meyer 1996][Stolzmann et al. 200]), is also under investigation.



(a)



(b)

Figure 6: Showing the performance of ZCSL on the 37 bit multiplexer problem.

ACKNOWLEDGEMENT

Thanks to the members of the Learning Classifier System Group at UWE for many useful discussions during this work.

REFERENCES

Bull, L. & Hurst, J. (2002) ZCS Redux. *Evolutionary Computation,* in press

Bull, L. & O'Hara, T. (2001) NCS: A Simple Neural Classifier System. *UWE Learning Classifier Systems Group Technical Report 01-005.* Available from http://www.csm.uwe.ac.uk/lcsg.

Butz, M., Goldberg, D.E. & Stolzmann, W. (2000) Introducing a Genetic Generalization Pressure to the Anticipatory Classifier System: Part 1 - Theoretical Approach. In *Proceedings of the 2000 Genetic and Evolutionary Computation Conference - Gecco 2000.* Morgan Kaufmann, pp34-41.

Butz, M., Kovacs, T., Lanzi, P-L & Wilson, S.W. (2001) How XCS Evolves Accurate Classifiers. In *Proceedings of the 2001 Genetic and Evolutionary Computation Conference - Gecco 2001.* Morgan Kaufmann, pp927-934.

Donnart, J-Y. & Meyer, J-A. (1996) Spatial Exploration, Map Learning, and Self-Positioning with MonaLysa. In P. Maes, M. Mataric, J-A. Meyer, J. Pollack & S.W. Wilson (eds.) *From Animals to Animats 4: Proceedings of the Fourth International Conference on Simulation of Adaptive Behaviour,* MIT Press, pp 204-213.

Gerard, P. & Sigaud, O. (2000) YACS: Combining Dynamic Programming with Generalization in Classifier Systems. In P-L. Lanzi, W. Stolzmann & S.W. Wilson (eds) *Advances in Learning Classifier Systems: Proceedings of the Third International Workshop.* Springer, pp52-69.

Holland, J.H. (1975) *Adaptation in Natural and Artificial Systems*, University of Michigan Press.

Holland, J.H. (1976) Adaptation. In R. Rosen & F.M. Snell (eds) *Progress in Theoretical Biology, 4.* Plenum.

Holland, J.H. (1986) Escaping Brittleness. In R.S. Michalski, J.G. Carbonell & T.M. Mitchell (eds) *Machine Learning: An Artificial Intelligence Approach, 2.* Morgan Kauffman, pp48-78

Holland, J.H. (1990) Concerning the Emergence of Tag-Mediated Lookahead in Classifier Systems. *Physica D* 42:188-201.

Holland, J.H., Holyoak, K.J., Nisbett, R.E. & Thagard, P.R. (1986) *Induction: Processes of Inference, Learning and Discovery*. MIT Press.

Lanzi, P-L. (1997) A Model of the Environment to Avoid Local Learning. Technical Report N.97.46 Dipartimento di Elettronica e Informazione, Politecnico di Milano.

Mackintosh, N.J. (1974) *The Psychology of Animal Learning.* Academic Press.

Riolo, R. (1991) Lookahead Planning and Latent Learning in a Classifier System. In J-A. Meyer & S.W. Wilson (eds.) *From Animals to Animats: Proceedings of the First International Conference on Simulation of Adaptive Behaviour*, MIT Press, pp316-326.

Samuel, A.L. (1959) Some Studies in Machine Learning using the Game of Checkers. *IBM Journal of Research and Development,* 3: 211-232.

Seward, J.P. (1949) An Experimental Analysis of Latent Learning. *Journal of Experimental Psychology* 39: 177-186.

Stolzmann, W. (1998) Anticipatory Classifier Systems. In J.R. Koza (ed) *Genetic Programming 1998: Proceedings of the Third Annual Conference*. Morgan Kaufmann, pp658-664.

Stolzmann, W. (2000) An Introduction to Anticipatory Classifier Systems. In P-L. Lanzi, W. Stolzmann & S.W. Wilson (eds) *Learning Classifier Systems: From Foundations to Applications*. Springer, pp175-194.

Stolzmann, W. & Butz, M. (2000) Latent Learning and Action Planning in Robots with Anticipatory Classifier Systems. In P-L. Lanzi, W. Stolzmann & S.W. Wilson (eds) *Learning Classifier Systems: From Foundations to Applications*. Springer, pp301-320.

Stolzmann, W., Butz, M., Hoffmann, J. & Goldberg, D.E. (2000) First Cognitive Capabilities in the Anticipatory Classifier System. In J-A. Meyer, A. Berthoz, D. Floreano, H. Roitblatt & S.W. Wilson (eds) *From Animals to Animats 6: Proceedings of the Sixth International Conference on Simulation of Adaptive Behaviour,* MIT Press.

Sutton, R.S. (1990) Integrated Architectures for Learning, Planning and Reacting based on Approximating Dynamic Programming. *Proceedings of the Seventeenth International Conference on Machine Learning*. Morgan Kaufmann, pp216-224.

Tomlinson, A. & Bull, L. (1998) A Corporate Classifier System. In A.E. Eiben, T. Bäck, M. Schoenauer & H-P. Schwefel (eds.) *Parallel Problem Solving from Nature - PPSN V*, Springer, pp. 550-559.

Wilson, S.W. (1987) Classifier Systems and the Animat Problem. *Machine Learning,* 2: 199-228.

Wilson, S.W. (1994) ZCS: A Zeroth-level Classifier System. *Evolutionary Computation* 2(1):1-18.

Wilson, S.W. (1995) Classifier Fitness Based on Accuracy. *Evolutionary Computation* 3(2):149-177