Aliasing in XCS and the Consecutive State Problem : 2 - Solutions

Alwyn Barry

Faculty of Computer Studies and Mathematics,
University of the West of England,
Coldharbour Lane, Bristol, BS16 1QY, UK

Email: Alwyn.Barry@uwe.ac.uk
Phone: (++44) 117 965 6261 ext. 3777

Abstract

The 'Aliasing Problem' within XCS (Wilson, 1995, 1998), first identified by Lanzi (1997), does not only appear whenever the aliased states occur in separate environmental locations but also when they occur consecutively (Barry, 1999). Lanzi (1997, 1998) introduced a mechanism that could solve the Aliasing Problem through the use of memory mechanisms within XCS (Wilson, 1995; Cliff and Ross, 1994). Whilst this mechanism is a solution to the general problem of aliasing, it is a heavyweight solution. By limiting the scope of a solution to the Consecutive State Problem, which is shown to be a sub-problem of the Aliasing Problem, a simpler solution is proposed, and is shown to adequately address this problem. The application of a potential solution utilising explicit action duration identification is discussed and shown to be inadequate both as a solution to the Consecutive State Problem and for more general use within XCS.

1 INTRODUCTION

When Wilson (1995) presented the XCS classifier system he provided results from two sets of investigations. One, using the Multiplexor problems, demonstrated that XCS was capable of accurately learning the payoff mapping for previously unseen complex boolean relationships. It demonstrated the ability of XCS to form complete State × Action × Payoff mappings within single-step (immediate reward) environments. The second used a Woods environment (Woods-2) to illustrate the abilities of XCS within multi-step environments (environments where reward is obtained after a number of movements within the environment), and showed that XCS was able to form a compact classifier population maintaining the State × Action × Payoff mapping of the environment - an ability further enhanced by later modifications to XCS (Wilson, 1998). Whilst Kovacs (1996, 1997) has produced further results within single-step environments, particularly concentrating upon the formation and maintenance of the optimal classifier representation for a given environment, Lanzi (1997, 1998) has investigated the application of XCS to multi-step environments. In applying XCS to progressively more complex Woods environments he identified that XCS had difficulty in finding solutions within the "Woods102" environment. This environment is non-Markovian due to the duplication of an input vector in two separate positions within the environment which require the same action but present different payoffs. This finding is no surprise given the roots of XCS in the mechanisms of Temporal Difference methods of reinforcement learning (Sutton, 1988; Watkins, 1989). Nevertheless, it does impose limitations on the application of XCS, since the consequent requirement for complete and unambiguous sensory perception is often undesirable.

Lanzi (1997) termed this problem the 'Aliasing Problem' and sought a solution to it using a memory mechanism first proposed by Wilson (1994, 1995) and applied within ZCS by Cliff & Ross (1994). This mechanism used an additional memory bit vector which classifiers could match as part of their condition and set as part of their action. Using the generalization abilities of XCS it was hypothesized that optimal memory settings would emerge to solve the Aliasing Problem. The XCS would learn to set a memory bit before one of the two aliased states and thereby disambiguate the inputs of the two states. This would in turn cause separate classifiers to be created for each aliasing state. Unfortunately the extent of the disruptive nature of the aliasing states was not investigated, and whilst some success was reported it was not until the setting of memory bits was directly related to environmental rewards and limited to exploitation cycles (thereby sacrificing the complete input mapping properties of XCS) that a satisfactory solution was found (Lanzi, 1998).

Recently Barry (1999) has carried out further investigations into the Aliasing Problem as part of wider research into the emergence of hierarchical invocation of classifier sequences. This work identified a form of the Aliasing Problem where the aliasing states occur in consecutive states. Using this problem it was shown that within a simple environment consisting of a single chain of states the aliased states not only cause payoff prediction inaccuracy in the classifiers covering the aliasing states but may also generate inaccuracy in the classifiers covering the immediately preceding if exploration of the states is not uniform. It was also discovered that it was possible for the classifiers covering the aliased states to proliferate in an over-generalised form by trading off a small decrease in accuracy for the additional GA opportunities afforded by involvement in the action sets of non-aliased states. In situations where this proliferation was controlled and exploration was performed uniformly it was demonstrated that XCS was unable to sustain suitable covering classifiers for the aliased states where competition for population space was high, but was involved in a constant but fruitless exploration for adequate classifiers. Thus, within a single state chain environment it was demonstrated that in the presence of consecutive aliased states the ability of XCS to create a complete and accurate State × Action × Payoff mapping will be severely compromised.

Within certain environments the consecutive state problem will be likely to occur regularly. Consider, for example, the control of a robot moving about a room. The primitive wall following behavior required for this task will, in any robot without sensors which provide a unique 'global' reference point, present the same input to the robot whilst in wall following and therefore produce a set of consecutive aliased states along each wall. Lin (1993), for example, overcomes this problem by presenting a X,Y coordinate as an input to a robot navigation task which is used primarily to disambiguate environmental states. In general this form of global reference point is difficult to provide economically, and therefore an alternative solution to the consecutive state problem is desirable.

In this work the hypothesis that the Consecutive State Problem is a sub-problem of the Aliasing Problem is presented. Although the Consecutive State Problem could be solved using the memory technique proposed by Lanzi, this mechanism requires modification to XCS to introduce the additional memory techniques discussed above, and has been problematic to implement (Lanzi, 1998). Furthermore, it's adequate implementation requires a change in the explore/exploit regime, thus removing one of the key features of XCS learning, namely the formation of a complete, accurate, and optimally general state × action × payoff mapping (Kovacs, 1996). If the Consecutive State Problem is a sub-problem of the Aliasing Problem, it should be possible to devise alternative solutions to this sub-problem which may be simpler to implement with less impact upon the operation of XCS. This paper seeks to investigate this hypothesis and provide results from two potential solutions to the Consecutive State Problem.

2 XCS STRUCTURE AND OPERATION

The XCS Learning Classifier System (Wilson, 1995, 1998) is, on an initial inspection, similar to traditional Learning Classifier Systems. Detectors interact with an 'environment' to produce a binary encoded message which becomes the input to the XCS. This is *matched* against a population of classifiers, each consisting of a ternary coded condition and an encoded action, in order to identify those classifiers which are relevant to the current input condition. Those classifiers which *match* the message are used to create the *Match Set* [M] of the classifier system - the set of *Action Sets* which each identify: an action, the classifiers which have been matched that propose the action, and the predicted payoff that will be received upon performing the action calculated from a weighted sum of the payoff prediction of each classifier in the action set. An action set [A] is chosen from [M] to perform an action; chosen arbitrarily if *exploring* to enhance the classifier representation or chosen by selecting the highest predicted payoff Action Set if seeking to *exploit* the learnt classifier representation. The action advocated by [A] is performed in the environment by decoding the action representation through an effector interface. If a reward R is received from the environment the goal is considered to have been reached and R is used to update the predictions of all classifiers in [A] using the modified Widrow-Hoff update mechanism known as MAM (Venturini, 1994). If no reward is received, and the environment is potentially a multi-step environment, the action is considered to be one action en route to the goal and payment is taken from the maximum prediction of [M] in the next iteration discounted by a discount factor γ ($0 < \gamma < 1$). Thus, any accurate classifiers in an [A] which leads directly to a reward R can be expected to converge to a prediction of R, those one step back will converge to a prediction of γR, and those i steps before the reward will converge to $\gamma^i R$. The speed of convergence is controlled by the learning rate parameter β ($0 < \beta \leq 1$) within the Widrow-Hoff update equations.

Although these processes are clearly related to traditional LCS, in particular Animat and ZCS (Wilson, 1983, 1994), the update method is novel within an LCS. In fact, the entire XCS 'strength' formulation is novel. Each classifier carries with it not only the *Prediction* measure, the prediction of the average payoff it receives when invoked, but also two other related measures - the *Error* and *Accuracy* which identify the accuracy of this prediction. A classifier can be inaccurate because it's prediction has not yet been updated sufficiently to make it accurate or because it has an over-general condition which involves the classifier in too many [M]. Inaccurate classifiers could nonetheless have a high prediction and therefore it is important to remove them in favor of accurate classifiers. To facilitate this, the GA induction element of XCS separates the measure used in the selection of classifiers for crossover and/or mutation from the prediction. The new measure introduced is termed the classifier *Fitness*, and is the accuracy of the classifier relative to other classifiers in the [A] the classifier occurs within. Thus, the GA will favor accurate classifiers over inaccurate and will, over time, replace inaccurate classifiers with accurate versions. Furthermore, the fitness is used to weight the contribution of the classifier's prediction within [A] so that accurate classifiers contribute more and drive the System Prediction towards higher accuracy whilst increasing the calculated error within the inaccurate classifiers. Interestingly, because an accurate general classifier occurs in more [A] than an accurate but more specific classifier, and because the invocation of the GA is tied to occurrences within [A], the more specific classifiers are also driven out of the population. The classifier deletion mechanism, used when the population becomes full, deletes classifiers based on the average number of classifiers which exist in the Action Sets each classifier appears within, thereby dynamically adjusting the population composition to provide sufficient population niches (Booker, 1989) for all the accurate optimally general classifiers (given sufficient population space). The Optimality Hypothesis (Kovacs, 1996) suggests that XCS is thus capable of identifying and maintaining the accurate optimally general population (termed [O]), and this has been demonstrated for a number of small problems (Kovacs, 1996, 1997; Saxon and Barry, 1999).

Within the limits imposed the explanation of the XCS structure and operation is necessarily truncated, and other novel features (such as the *MacroClassifier* formulation and *Subsumption Deletion*) have not been addressed. The interested reader is directed to Wilson (1995, 1998) and Kovacs (1996) for more detailed explanations.

3 CONSECUTIVE STATE PROBLEM

A Finite State World (Grefenstette, 1987; Riolo, 1987) is an environment consisting of *nodes* and directed *edges* joining the nodes. Each node represents a distinct environmental *state* and is labelled with a unique state identifier. Each node also maintains a message that the environment passes to the XCS when at that state. Each edge represents a possible *transition* path from one node to another and is labelled with the XCS generated action(s) that will cause movement across the edge in the stated direction to a destination node. An edge can lead back to the same node. Each node has exactly one label and message, and each message is unique within a Markovian FSW and normally equivalent to the node's label. Non-Markovian environments can be created by allowing a message generated by one node to be re-used by other nodes. Each edge may have one or more labels and these will be re-used on edges emanating from any node which allows that action to be executed when in the state represented by the node. At least one node must be identified as a *start* state, signifying that the XCS will be operating in that state when each new learning trial begins. If more than one start state is provided the actual state from which a trial is started is chosen randomly from the available start states. Additionally, one or more nodes must be identified as *terminal* states. Transition to any one of these states represents the end of a learning trial and each will have an associated reward value representing an environmental reward which is passed to XCS upon transition into such a state. Terminal states do not have any transitions emanating from them - upon arrival the trial is ended, the next iteration will represent a new trial and the environment will reset to a [selected] start state.

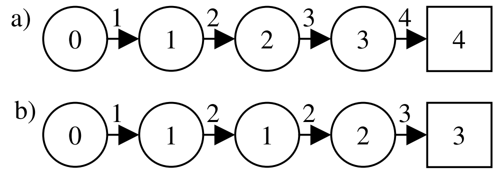

Figure 1: Markovian and Non-Markovian FSW

Consider a finite state world consisting of the five states s_0 to s_4, depicted in Figure 1a. The state s_0 is the start state, s_4 is the terminal state generating reward R. To create a non-Markovian FSW the states s_1 and s_2 are labeled with a detector message d=1, and the edges $s_1 \rightarrow s_2$ and $s_2 \rightarrow s_3$ are labeled with action a=2. This finite state world will be termed *FSW-5A-2* to identify the 5 states, an aliased world, aliased over two states.

Three classifiers are required to traverse this FSW : 0→1, 1→2, 3→4. Call these classifiers c_1, c_2, and c_3. Upon reaching s_4 classifier c_3 receives a reward R. c_3 will eventually converge so that $p = R$. From this point, for moving from s_2 to s_3, c_2 will be consistently given a payoff γR. However, for moving from s_1 to s_2 the classifier c_2 will also be given a payoff which is γP_2. If the learning rate β within the Widrow-Hoff mechanism was 1, then the prediction would oscillate well within the limits $\gamma^2 R$ and γR. For simplicity, let us assume that P_2 varies around the average payoffs that would have been received at the states had they not been aliased $((\gamma R + \gamma^2 R) / 2)$, and that $0 < \beta < 1$. In this case the variance will reduce to $\pm \beta((\gamma R - \gamma^2 R) / 2)$. Unless the value of β is very small, or the aliased states are sufficiently far from the reward source for the successive application of the discount factor γ to reduce the payoff to a very small amount[1], the variance will remain sufficient to produce an oscillation in P_2 which is greater than ε_0, the minimum error for a classifier to be considered accurate.

4 HYPOTHESIS

Barry (1999) demonstrated that the Consecutive State Problem is a form of the Aliasing Problem and identified a number of important consequences of the Consecutive State Problem for the formation of an accurate optimally general State × Action × Payoff mapping of the environment. From this work it seemed that the

[1] At present XCS presents the allowable prediction error ε_0 as an absolute parameter. However, given that the discount factor for an arbitrary classifier i steps from the reward source c_i is $\gamma^i R$, classifiers more than about 12 steps away from even a moderately generous reward source such as those in the Woods environments used within (Wilson, 1995) will have sufficiently low prediction to allow significant variance relative to it's stable prediction without exceeding ε_0. The investigation of relative error measures for the accuracy calculations, similar to those used by Barry (1999) for collecting error readings across all classifiers within a trial, may therefore be productive.

Consecutive State Problem could be solved using techniques which are simpler but not appropriate for all forms of the Aliasing Problem. This gave rise to the following hypothesis:

The Consecutive State Problem is a sub-problem of the Aliasing Problem. The Consecutive State Problem will admit to specific solutions that cannot address the Aliasing problem as a whole.

To see the rationale for the development of this hypothesis we need to introduce two lemmas.

Lemma 1 - *The memory solution (Lanzi, 1997, 1998) is a general solution which is applicable to all occurrences of the Aliasing Problem.*

Consider the following formulation of the Lanzi (1997) memory solution to the Aliasing Problem applied to *FSW-5A-2*. One bit of memory which is appended to the input message created by the XCS detectors can be used to solve this FSW. Classifiers covering all non aliased states can ignore the setting of this bit by adding a wildcard at the designated bit position. When in state s_1 or s_2 the bit can be used to differentiate between the aliased states by using it's 0 value to identify s_1 and it's 1 value to identify s_2 [we shall not discuss how this might be achieved]. In order to create accurate classifiers the GA within XCS will then discover two separate classifiers distinguished by this bit value, each of which will accurately reflect the relevant discounted payoff value. If the FSW was changed so that the aliasing states were states s_1 and s_3, or any other combination of two states, the same technique could be used. This argument can be trivially extended to any reflect two aliased states on joined but distinct state chains at different payoff positions or at the same position in distinct state chains ending in different reward values.

Lemma 2 - *The Consecutive State Problem is distinct from the Separate State Problem*

Consider again the five state aliasing problem of *FSW-5A-2*. The inaccuracy of the classifier covering s_1 and s_2 was due to the discount of the payoff between invocations of the classifier. If the discounting mechanism was disabled until a change of input then the classifier would receive one payoff for its full time of activity and the payoff would be consistent, thereby making the classifier accurate. Now if the FSW was changed so that the aliasing states were states s_1 and s_3, or any other combination of two non consecutive states, the same technique could not be used to achieve classifier accuracy due to the correct discounting of the payoff for any intervening classifiers. Clearly the same argument can be applied to any situation where two aliased states on joined but distinct state chains at different payoff positions or at the same position in distinct state chains ending in different reward values are considered. Thus the Consecutive State Problem is a particular instance of the Aliasing Problem, separate but related to the case where aliased states appear at non consecutive states (which shall be termed the *Separate State Problem*).

If Lemma 2.1 and 2.2 are correct, then it is possible to suggest that the Consecutive State Problem and the Separate State Problem are two sub-problems which show the same properties and can admit to the same solution, but for which there can be devised independent solutions which do not cover the whole, thereby giving the hypothesis.

5 EXPERIMENTAL INVESTIGATION

In the previous section it was claimed that if the application of the reward could be delayed until the point of leaving the aliasing states, the discounting of payoff would not occur within the aliasing states and therefore the classifier covering the aliasing states would be able to represent a single payoff value accurately. In this section we identify and examine one candidate mechanism.

5.1 A PROPOSED SOLUTION

In seeking a solution the main obstacle is the difficulty in identifying the difference between an action that leads to a consecutive aliasing state and an action which leads back to the same state (a 'null' action). The same message is received from the environment in consecutive iterations in both cases and oscillations in prediction will still occur in the latter case for over-general classifiers matching in this state. However, if the environment does not allow null actions, then it would be possible to repeatedly re-choose the same action set while the message remains the same, rewarding the action set only when payoff is received from the first action set chosen from a different message. Preventing payoff in this way will ensure that only a single payoff is received for each distinct input vector (unless the Separate State Problem exists within the environment) and therefore eliminates the causes of the Consecutive State Problem.

There exists a simple implementation to this proposed mechanism which requires the storage of the message received in the previous iteration (set to a dummy message in the first iteration of a trial). At the start of a new iteration the message received is compared with the stored message and if the same then the previous match set and action set are restored. The action selection stage is therefore not needed in this iteration, and the restored action is applied to the environment. If an environmental reward is received, this is given to the restored action set, but if no reward is received the payoff to the previous action set is prevented. The induction algorithms operate as normal.

This mechanism would appear, on first inspection, to prevent the Animat exploring alternative reward routes from within the aliasing states. However, unless the alternative route could be reached from all the aliasing states within a set of consecutive states, the action leading to the alternative route will itself be aliasing. If the route can be reached from all the aliasing states, then it could be chosen at the first of the aliasing states and will still be explored.

5.2 THE TEST ENVIRONMENT

Both Wilson and Lanzi have utilized the Woods environments in their work, but these environments are not easily scaled with fine control in either length or complexity. Therefore, the Woods environments are set aside in favor of the Finite State World environments introduced in the preceding section. All Woods environments can be represented using FSW in any case, though not all FSW can be represented by Woods environments. For these tests a nine consecutive state FSW was constructed. The first state s_0 is also the start state, and the terminal state is s_8 at which a reward R=1000 is provided. The states s_i ($1 \le i \le 7$) each have two edges $s_i \rightarrow s_{i+1}$ and $s_i \rightarrow s_{i-1}$, labeled with actions 0 and 1 respectively. The state s_0 has a single edge emanating from it, $0 \rightarrow 1$, labeled with actions 0 and 1. This environment provides uniform exploration rates for all states but does not have any 'null action' edges leading back to the originating state as was the case in the environments used in Barry (1999). The states s_3 to s_6 were aliased by providing an appropriate message in each experiment. Other parameters for XCS were set in all experiments as follows: N=400, p_1=10.0, ε_1=0.01, f_1=0.01, R=1000, γ=0.71, β=0.2, ε_0=0.01, α=0.1, θ=25, X=0.8, μ=0.04, P(#)=0.33, s=20 (see Kovacs (1996) for a parameter glossary), and the maximum trial length was set to 50.

5.3 APPLYING ACTION PERSISTENCE

In order to provide an empirical proof for the hypothesis, the previously described persistence mechanism will be applied within the test environment.

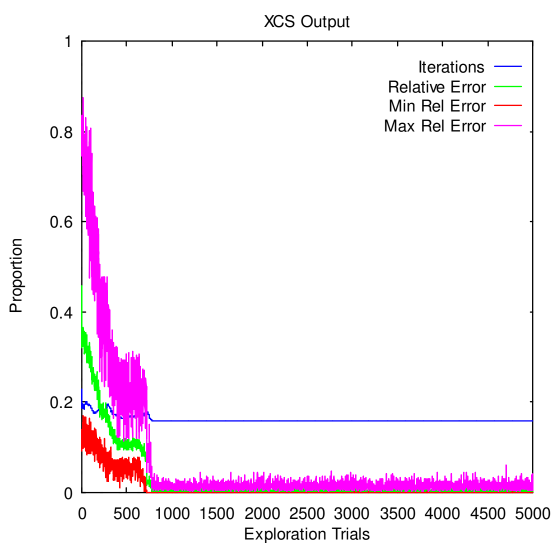

Figure 2 : The fall in System Relative Error in the non aliased FSW-9(2) environment averaged over 10 runs.

A non aliased version, termed *FSW-9(2)* was created by using state messages which were equivalent to the state numbers. The aliased version, *FSW-9A(2)-4* was created using the messages for each state as follows: 0- 00000, 1- 00001, 2- 00010, 3..6- 11000, 7-00011, 8-00100, chosen in order to minimize any likely aliasing state disruption to other classifiers (Barry, 1999). Baseline results for learning within the non aliased version of the environment and within the aliased version were obtained by running the XCS ten times in each environment, capturing the System Relative Error measures (Barry, 1999) at the end of each exploitation trial and averaging the results across each run. The results from the first 5000 exploitation trials within the non aliased environment are shown in Figure 2. These results are interesting when compared to the equivalent non-aliased nine state environment used Barry (1999). The greater oscillation in the maximum System Relative Error suggests that the wider range of movement and lack of null actions in the environment used for these tests makes the task of learning more difficult.

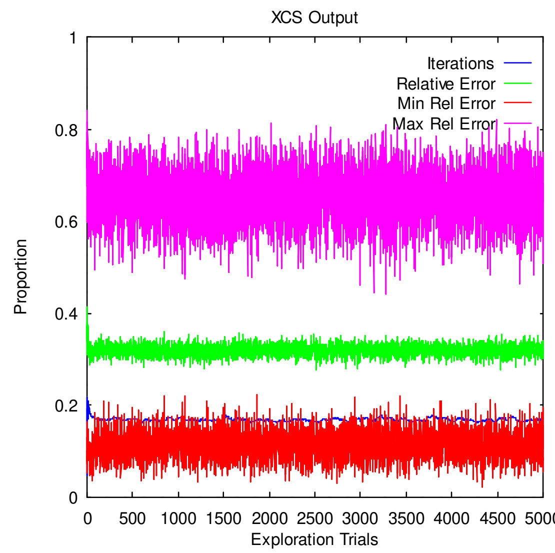

Figure 3 : The averaged System Relative Error results from 10 runs of XCS within *FSW-9A(2)-4*.

When the environment was modified to it's aliased form and the experiment was run it was therefore unsurprising that the degree of oscillation of the maximum and minimum System Relative Error readings was much higher than in previous environments tested. As a result of the greater movement allowed, the classifier covering the aliasing states was noticed to have an even more profound effect upon the classifiers covering the neighboring states than reported in Barry (1999) when the populations resulting from these runs was examined. A number of alternative message encoding schemes were tested but non appeared to provide any advantage over that which had already been chosen. The averaged results of this experiment are shown in Figure 3.

The XCS was modified in the manner outlined in section 5.1 to introduce action persistence over aliased states to the XCS. The modified XCS was initially tested by inserting the hypothesized [O] into the population, turning the induction algorithms off, and running the XCS with all other parameterization set to the values stated previously. These tests indicated that the modified XCS was able to find the optimal predictions for all classifiers with no error and in the same time span that would be expected for the equivalent non-aliasing five-state FSW environment. Knowing that the modified XCS could deal with the aliased states correctly, the ability of the modified XCS to learn within the *FSW-9A(2)-4* environment was examined by running the XCS in the environment ten times with no initial population and all induction algorithms on. The System Relative Error was captured from each run and averaged, and the results from the first 5000 exploitation trials are shown in Figure 4.

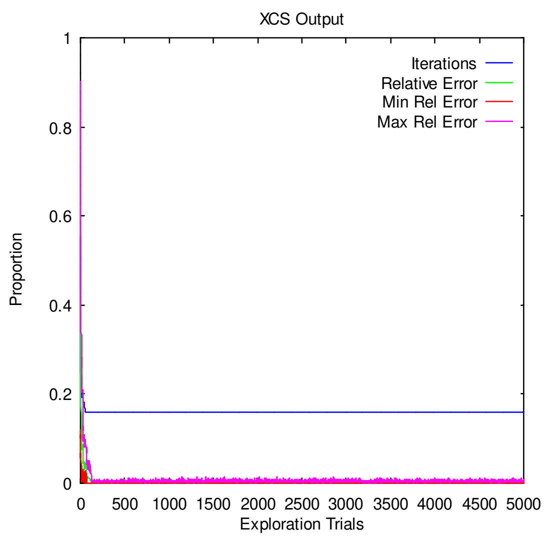

Figure 4 : The minimal System Relative Error illustrates that the environment *FSW-9A(2)-4* is mapped optimally when using persistence of actions over aliased states.

The rapid reduction in the System Relative Error measures indicates rapid and accurate learning. A comparison with Figure 1 illustrates that learning is actually more rapid within FSW-9A(2)-4 with persistence of actions than in the non aliased FSW-9(2) environment. This is due to the effective reduction in states to a five state environment, making the learning task much simpler. An examination of the final populations from the 10 runs showed that they had all converged on [O], with the total population size between 27 and 40 macro classifiers with all classifiers which were not members of [O] having low experience and very low numerosity indicating they were the unfruitful product of continued exploration. Table 1 gives [O] taken from a typical run.

This experiment has demonstrated that the consecutive aliasing state problem can be overcome by a mechanism, the use of persistent actions, which does not solve the more general Aliasing Problem. Therefore, it is concluded that the Consecutive State Problem is a sub-problem of the more general Aliasing Problem and that a solution exists for the Consecutive State Problem which does not address the whole aliasing problem.

Table 1 : Classifiers in [O] learnt by the persistent form of XCS acting within FSW-9A(2)-4

Class.	Pred.	Error	Fit.	Acc.	Num.	MS	Exp.
###11→0	1000.0	0.0000	1.00	1.00	41	42.7	27517
###11→1	503.35	0.0007	1.00	1.00	31	41.0	12371
1####→0	707.21	0.0013	0.92	1.00	38	44.6	40047
1####→1	357.91	0.0006	0.97	1.00	34	40.2	25079
###10→0	503.18	0.0007	0.99	1.00	49	51.7	53263
###01→0	358.76	0.0009	0.98	1.00	51	54.3	66974
###01→1	180.50	0.0004	0.92	1.00	36	45.4	52773
0##00→0	254.05	0.0003	1.00	1.00	38	44.0	41199
0###0→1	253.91	0.0003	0.95	1.00	48	58.0	81085

6 DISCUSSION

The experiments demonstrate that within the simple FSW developed for these tests it is possible to utilize action persistence to eliminate the Consecutive State Problem. However, this solution is limited by it's inability to deal with null actions. Unfortunately, even in environments like the Woods environments which would appear not to allow null actions these are often present - an action which is not allowed, such as an attempt to move onto a 'Rock' square within the Woods environments, will prevent movement to a new state and would therefore be modeled within a FSW as a null action. This is therefore a fairly severe limitation.

6.1 DEALING WITH NULL ACTIONS

Fortunately it is possible to conceive of a number of potential solutions to the problem. In environments such as the Woods environment the attempt to move to an illegal position can be easily detected by providing the XCS with feedback from the environment that the attempted move was illegal. This information can be used to prevent action persistence from starting, which together with the standard discounting payoff mechanism should allow XCS to learn to select more appropriate actions during exploitation trials. A similar effect can be gained by introducing an additional parameter π to XCS which gives a limit on the number of iterations over which action persistence may operate. On each iteration in which an action is reinstated because the same message is presented from the environment a counter is incremented. Upon reaching the limit π action persistence is stopped, a 'reward' equal to the minimum environmental reward is given, and the trial is concluded. Over a short number of explorations the classifier leading to a persistent action over a null action will converge to a low prediction and therefore not be selected during exploitation trials. It is worth noting that a parameter already exists within XCS that gives a maximum trial length before a trial is terminated and a new trial begins. This parameter prevents XCS from eternally iterating between two or more cyclically connected states, and has the same effect as the proposed new parameter. The new parameter π is introduced because the existing trial length parameter has a relatively high value which is inappropriate for the detection of null actions.

6.2 SPECIFYING ACTION DURATION

Cobb and Grefenstette (1991) employed classifiers which included actions which identified a duration over which the action of the classifier was to occur within the SAMUEL LCS. They were able to demonstrate that a LCS which included this facility was able to discover classifiers with suitable action duration under the action of the GA for a missile pursuit problem. On first inspection it would appear that this technique could be readily applied to XCS to solve the consecutive state aliasing problem. A classifier which identifies both the action and the correct duration for the action would receive a constant payoff and therefore be identified as accurate and of high fitness. A classifier identifying the incorrect duration would receive no payoff (if it persists too long), a fluctuating payoff (if it persists for too short a time and so is re-invoked), or a lesser payoff (if it persists for too short a time but is not re-invoked it will be further down the feedback chain), and therefore in each case will not be selected in exploitation. Therefore, without change to the credit allocation or induction mechanisms of the XCS, and with only minor changes to the performance component, XCS would appear to have all the mechanisms necessary to generate, identify and proliferate classifiers which act for the correct time period.

However, this approach has potential limitations. Firstly, if classifiers can be discovered which successfully move over all the aliased states, they will only be useful if all occurrences of consecutive aliasing state sets generating a given message are the same length since the length of invocation is hard coded within the classifier. Secondly, the addition of timing information to a classifier increases the action length (and thereby the search space) unnecessarily for the many other classifiers which do not require this facility. Thirdly, and finally, the XCS implementation proposed by Wilson (1995, 1998) includes only primitive search over the action space (mutation only) and thus any significant extension of the action encoding will necessitate the full application of GA search to the whole classifier in order to search over the duration fields adequately.

Unfortunately, the operation of the XCS itself provides some more fundamental problems. Firstly, in an environment where consecutive states have the same action message all the way to a terminal state a classifier with the move-to-goal action could develop a duration which continues the action over all intermediate states to the goal. Whilst this is potentially beneficial in the short term, it limits exploration of later states and prevents the timely production of a full State $\times$ Action $\times$ Payoff mapping within the classifier population. This problem was investigated by using *FSW-9* with one action (move forward) and a maximum action persistence sufficient to traverse from the start state to the terminal state. The population was initialized to cover all conditions and actions and the induction algorithms were disabled. After 15000 exploration trials the population was examined. The classifiers covering the start state that did not lead directly to the terminal state were found to have over 10 times more experience than the next most experienced classifiers in the population (average 2190 for s_0 compared with 219 for s_1 in one typical run), with the classifier that led directly to the terminal state having over 60 times more experience.

Secondly, the persistence of an action will cause the classifiers concerned to be given the payoff received from the destination state and so converge on a prediction which would normally occur much later in the state chain. Since the maximum system prediction is used as the payoff value within multi-step problems the prediction of these classifiers will be passed on to other preceding classifiers. In the investigation using *FSW-9*, all classifiers leading to the terminal state converged on R whilst all classifiers which lead to other states within the environment converged to γR. Whilst still allowing XCS to choose the optimal classifier, the destruction of the temporal difference properties of the mapping generated by XCS cannot be justified.

This problem would seem to be able to be addressed, for the sole solution of the consecutive state aliasing problem, by limiting the persistence of an action to the cases where the message remains the same in consecutive states. In this case a classifier which tries to persist with an action for longer than a message is consecutively posted can be rewarded an arbitrary very low reward so that the mapping for actions of an incorrect duration are poorly valued and thus not selected during exploitation.

This possibility was investigated using *FSW-9A(2)-4* and a two aliasing state version of *FSW-9A(2)-4* termed *FSW-9A(2)-2*. The XCS was modified so that the environment decodes the persistence specification in the action and repeats the action the number of times specified, or until the environment gives a new message, before handing control back to the XCS with an indication of whether the full action persistence was completed. If the full duration was completed before the message changed then the normal payoff mechanism is used at the end of the delay to give a constant feedback to the classifier. If, however, the full duration was not completed the XCS now pays back the minimum environmental reward in lieu of the normal payoff. If the duration is too short, the classifier will be inaccurate. If it is too long, the classifier will have a low prediction and not be used within exploitation trials. Thus, the classifiers proposing an action which persists for the correct duration should be selected.

The experiments consisted of 10 runs of 15,000 exploitation trials with all other parameterization kept at that described in section 5.2. The results showed that System Relative Error, although reduced, remained high for the four alias state test. The two alias state test was better, shown in Figure 5, but the System Relative Error was never eliminated.

An examination of the populations revealed that the XCS found classifiers with high numerosity which identified that no length three or four delays were required in any state, and no length two delays were required in the non-aliased states. It was also able to learn the generalized classifiers for most of the non aliased states, although there was a degree of disruption present. The classifiers covering the aliased states were present in small numbers with low experience. An examination of the location of the disruption of other classifiers revealed that, under exploration, a classifier could be invoked that moved into the second of the aliasing states from s_7. This would cause the invocation of the classifier providing a two step delay to receive a zero reward, generating inaccuracy. No solution to this problem, apart from memory solution similar to the approach used by Lanzi, exists. Thus the provision of action persistence specification within XCS is inappropriate both as a solution to the Consecutive State Problem and for general XCS use.

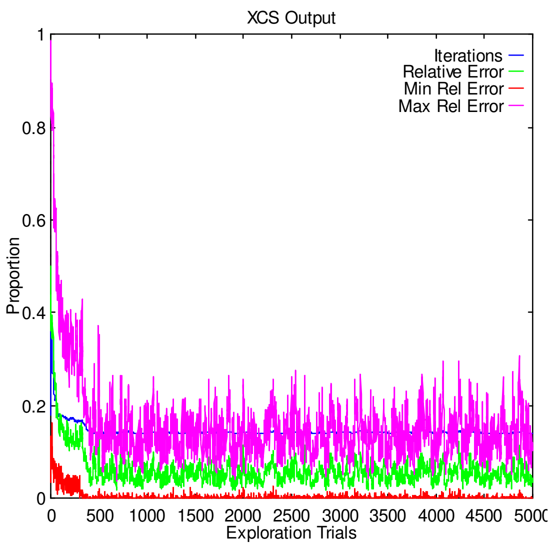

Figure 5 : System Relative Error remains within FSW-9A(2)-2 when attempting persistence delay learning.

7 CONCLUSIONS

Using results on the Consecutive State Problem from Barry (1999), it has been demonstrated that the Consecutive State Problem is a sub-problem of the Aliasing Problem. This finding allows the existence of a solution to the Consecutive State Problem that does not address the whole Aliasing Problem but can be implemented more simply. A solution was demonstrated which maintained an action whilst the same message was received by XCS, and its extension to environments which contain 'null actions' was discussed. An alternative candidate solution was identified based on a previous LCS implementation, but investigations have demonstrated that this solution is inappropriate for implementation within XCS.

Acknowledgements

The author wishes to acknowledge advice received from Stewart Wilson and Tim Kovacs in the development and testing of the XCS implementation now available from http://www.csm.uwe.ac.uk/~ambarry/LCSWEB

References

Barry, A.M. (1999), Aliasing in XCS and the Consecutive State Problem : 1 - Effects, submitted to the Intl Conf on Genetic and Evolutionary Computing,, 14-17 July, 1999.

Booker, L.B. (1989), Triggered Rule Discovery in Classifier Systems, in Schaffer, J.D. (ed.), *Proc. Third Intl. Conf. on Genetic Algorithms*, 265-274.

Cliff, D., Ross, S. (1994), Adding memory to ZCS, *Adaptive Behaviour 3(2)*, 101-150.

Cobb, H.G., Grefenstette, J.J. (1991), Learning the persistence of actions in reactive control rules.

Grefenstette, J.J. (1987), Multilevel Credit Assignment in a Genetic Learning System, in *Proc. Second Intl. Conf. On Genetic Algorithms and their Applications*, 202-209.

Kovacs, T. (1996), Evolving optimal populations with XCS classifier systems. Tech. Rep. CSR-96-17, School of Computer Science, University of Birmingham, UK.

Kovacs, T. (1997), XCS Classifier System Reliably Evolves Accurate, Complete, and Minimal Representations for Boolean Functions, WSC2: 2nd On-line World Conf. On Soft Comp. In Eng. Design and Manufacture.

Lin, L-J . (1993), *Reinforcement Learning for Robots using Neural Networks*, PhD Thesis, CMU-CS-93-103.

Lanzi, P.L. (1997), Solving problems in partially observable environments with classifier systems, Tech. Rep. N.97.45, Dipartimento di Elettronica e Informazione, Politecnico do Milano, IT.

Lanzi, P.L. (1998b), An analysis of the Memory Mechanism of XCSM, in *Proc. Intl Conf. on Genetic Programming.*

Riolo, R.L. (1987), Bucket Brigade performance: I. Long sequences of classifiers, in *Proc. Second Intl. Conf. on Genetic Algorithms and their Applications*, 184-195.

Venturini, G. (1994), *Apprentissage Adaptatif et Apprentisage Supervisé par Algorithme Génétique*. PhD Thesis, Université de Paris-Sud.

Wilson, S.W. (1983), Knowledge growth in an artificial animal, in *Proc. First Intl. Conf. on Genetic Algorithms and their Applications*, 196-201.

Wilson, S.W. (1994), ZCS, a zeroth level classifier system, *Evolutionary Computation 1(2)*, 1-18

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

Wilson, S.W. (1998), Generalization in the XCS Classifier System, in *Proc. Third Annual Genetic Prog. Conf.*