Probability-Enhanced Predictions in the Anticipatory Classifier System

Martin Butz[*]
Illinois Genetic Algorithms Laboratory
Department of General Engineering
University of Illinois at Urbana-Champaign
butz@illigal.ge.uiuc.edu

David E. Goldberg
Illinois Genetic Algorithms Laboratory
Department of General Engineering
University of Illinois at Urbana-Champaign
deg@illigal.ge.uiuc.edu

Wolfgang Stolzmann
Institute for Psychology III
University of Wuerzburg
Germany
stolzmann@psychologie.uni-wuerzburg.de

Abstract

The Anticipatory Classifier System (ACS) recently showed many capabilities new to the Learning Classifier System field. Due to its enhanced rule structure with an effect part, it forms an internal environmental representation, learns latently besides the common reward learning, and can use many cognitive processes. This paper introduces a probability-enhancement in the predictions of the ACS which enables the system to handle different kinds of non-determinism in an environment. Experiments in two different mazes will show that the ACS is now able to handle action-noise and irrelevant random attributes in the perceptions. Furthermore, applications with a recently introduced GA will reveal the general independence of the two new mechanism as well as the ability of the GA to substantially decrease the population size.

1 Introduction

In contrast to the condition-action-payoff structure of classifiers in other Learning Classifier Systems (LCSs), the Anticipatory Classifier System (ACS) has a condition-action-effect-payoff structure (Stolzmann, 1997, Stolzmann, 1998). Furthermore, the learning is based on the accuracy of the predicted effects (i.e. the anticipations) rather than on the payoff predictions, as in original LCS, or the accuracy of the payoff predictions, as in XCS (Wilson, 1995).

The anticipation based learning enables the ACS to learn latently (i.e. learning without getting any reward) which was already investigated in Stolzmann (1997). Moreover, the enhanced classifier structure results in a complete internal model of the environment rather than a condition-action-payoff model. This model enables the ACS to use more sophisticated processes (i.e. cognitive processes). Stolzmann, Butz, Hoffmann, and Goldberg (2000) investigates the present cognitive capabilities, introduces two new ones and outlines possible future applications.

Until now, the ACS considered all changing attributes as results from its actions. Butz, Goldberg, and Stolzmann (1999) revealed the limits of such an approach. It was shown that the ACS

[*]Visiting student from the University of Wuerzburg, Institute for Psychology III, Germany

relies on a perceptual causality in the environment and that any non-determinism challenges the learning mechanism. In particular, an environment was investigated where randomly changing attributes or noise in the actions challenged the ACS. Stolzmann (1997) already proposed the formation of an attention spot in the ACS by introducing an additional 'don't care' symbol in the effect part (rule right-hand side). However, an attention spot cannot solve tasks where an action in a particular situation can lead to different results. Thus, this paper introduces probability-enhanced effects (PEEs) in the right-hand side of a rule, that can form an attention spot as well as can consider different possible changes in the environment.

The next section gives an overview of the ACS with all its current mechanisms. Next, section 3 introduces the PEEs of a classifier and the associated mechanism. Section 4 gives results in mazes with disturbing random attributes as well as noise in the actions. Finally, a discussion is given.

2 Overview of the ACS

In Stolzmann (1997) the basic structure of the ACS with its anticipatory learning process (ALP) was introduced. Stolzmann (1999) published the additional mark in the ALP. Finally, Butz, Goldberg, and Stolzmann (2000) introduced an enhancement of the application of the ALP and a genetic algorithm (GA) to the ACS. This section gives an overview of the current state-of-the-art. Further details of the mechanisms in the ACS can be found in the cited papers.

2.1 The Basic Structure

An ACS always interacts with an environment. In order to be able to learn, it relies on a perceptual causality in successive states. At each time step t it perceives a state $\sigma(t) \in \{s_1, s_2, ..., s_m\}^L$, executes an action $\alpha(t) \in \{r_1, r_2, ..., r_n\}$ and receives payoff $\rho(t) \in IR$. m is the number of different possible detector values. (m can be greater than two but so far we only worked with small values of m). s_i represents the different detector values, L the number of detectors, n the number of different executable actions, and r_i the different actions.

The ACS presents its knowledge in rules, which are called classifiers. A classifier consists out of a condition part C, an action part A, an effect part E and a mark M. $C, E \in \{s_1, ..., s_m, \#\}^L$, $A \in \{r_1, ..., r_m\}$ and $M \in (\{s_{11}, ..., s_{1o_1}\}, ..., \{s_{L1}, ..., s_{Lo_L}\})$ where $o_1, ..., o_L$ are the number of different values of each attribute of a mark. A '#'-symbol in C (i.e. a 'don't-care' symbol) matches any detector information in the attribute while a '#'-symbol in E (i.e. a 'pass-through' symbol) predicts that the corresponding detector value does not change after the execution of the action A. Furthermore, each classifier cl has got the following parameters: (1) a quality $q \in [0, 1]$ measures the accuracy of the anticipations; (2) a reward measure $r \in IR$ predicts the payoff from an environment; (3) a time stamp t_s specifies the time when last the GA was applied in a set where cl was in; (4) an experience measure exp counts how often the classifier was present in an action set; (5) a numeriosity measure num measures the number of classifiers that cl represents (identical to the macro-classifiers in XCS, Wilson, 1995).

Learning in the ACS starts always with a completely general knowledge (i.e. classifiers for each action with only '#'-symbols in C and E). A behavioral act (see Figure 1) forms at first a match set out of the current population considering the current state $\sigma(t)$. Next, the ACS decides with a probability of p_x whether to choose an action randomly or whether to choose a classifier by roulette-wheel selection with the bid $q * r$ in the match set and choose its action. Considering the action, an action set is formed. After the execution of the action, first the ALP (with respect to the resulting state $\sigma(t+1)$) and then the GA modify the action set and produce the learning set. Finally, the reward measure r is updated considering the perceived payoff $\rho(t)$ and the next match set $M_{set}(t+1)$.

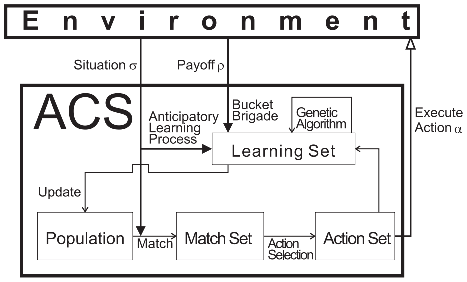

Figure 1: A behavioral act in the ACS with all the involved learning procedures

2.2 The ALP

The ALP modifies the action set with respect to the resulting state $\sigma(t+1)$. In the ALP new, more specialized classifiers are formed out of inaccurate, more general ones considering the mark M, the anticipation of the classifier, and $\sigma(t+1)$. Furthermore, the quality q is updated with learning rate b_q (i.e. increase: $q = (1-b_q)*q+b_q$ and decrease: $q = (1-b_q)*q$). A more detailed description of the ALP can be found in Butz, Goldberg, and Stolzmann (2000). If q decreases under $\theta_i = 0.1$, the classifier is considered as *inadequate* and is deleted. If q is greater than $\theta_r = 0.9$, ACS considers the classifier as *reliable* and uses it in the internal environmental model.

2.3 The GA

While the ALP only specializes rules and thus resembles a specialization pressure in the ACS, the GA generalizes. It is applied, after the ALP modified the learning set. It is only applied, if the average time since the last GA in the set is greater than the threshold $\theta_{ga} = 100$. Two classifiers are then selected with roulette wheel selection where the bid of each classifier is the cube of its quality (q^3). With a probability $\mu = 0.3$ each not-#-symbol in the condition part of the selected classifiers is changed to a #-symbol (i.e. mutated). Crossover is not applied. The resulting classifiers are inserted into the population if they are not completely general. If a produced classifier already exists, its numeriosity is increased. Finally, classifiers are deleted in the action set, if the size of the resulting set is greater than the action set size threshold θ_{as}.

2.4 Subsumption

In order to balance the specialization due to the ALP and the generalization due to the GA subsumption (similar to XCS Wilson, 1998) was introduced to the ACS (Butz, Goldberg, & Stolzmann, 2000). When a new classifier is generated and there is an accurate, more general classifier with a sufficient experience ($exp > 20$), then the new classifier is not inserted into the population but q or num of the old classifier is increased dependent if the ALP or the GA generated the new classifier, respectively. Although subsumption was introduced together with the GA it works independent from the GA and is applied in all the experiments presented here.

2.5 Reinforcement Learning

In order to realize reward learning, the reward measure r of the classifiers in the resulting learning set is updated similar to Q-learning and the Bucket-Brigade idea:

$$r = r + b_r * (\gamma * max_{cl \in M_{set}(t+1)}(q_{cl} * r_{cl}) + \rho(t) - r)$$

b_r is the learning rate and γ the discount factor. However, this paper investigates only the latent learning abilities of the ACS and thus, the formation of an environmental representation. The reinforcement learning abilities are not investigated herein.

3 The Probability-Enhanced Effects (PEEs)

In Butz, Goldberg, and Stolzmann (1999) two particular problems were identified that challenged the ACS mechanism. (1) Violated Relevance: When an environment sends additional randomly changing attributes, each classifier predicts the outcome of a random attribute with a probability of 50%. Thus, no classifier gets reliable and no environmental model is formed. (2) Action-Noise: When the result of an action is not deterministic (identical to the stochastic environments in Lanzi, 1999), the environmental model can only be built, if the probability of a successful execution is greater than 0.9 (otherwise, most classifiers do not get reliable). This section introduces the PEEs that will solve both challenges.

3.1 The New Structure of the Effect Part E

To be able to generate classifiers that predict all possible effects correctly and thus become reliable in these environments, E needs to be enhanced. Each attribute in E is no longer a symbol, but instead is replaced by a set of symbols with respective probabilities for the occurrence of each symbol, where the probabilities in each attribute always sum to one. Thus, for example the not enhanced effect part $E = [0, 1, \#, 2]$ is now coded as $E = [\{(0, 1.0)\}, \{(1, 1.0)\}, \{(\#, 1.0)\}, \{(2, 1.0)\}]$. We will write a not enhanced attribute in E simply as x rather than $\{(x, 1.0)\}$ to allow the old notation for not enhanced effects and to simplify the notation of PEEs. Enhancing the first attribute of E predicting a 60 to 40 percent change of 0 or 1, respectively, changes E to $[\{(0, 0.6), (1, 0.4)\}, 1, \#, 2]$. A classifier with PEEs is basically treated like a normal classifier. However, when the anticipations of the classifier are generated it does not predict a certain state anymore, but instead it gives a probability distribution of the attributes in the next state. If in each attribute one particular symbol predicts the corresponding attribute in the resulting state, then the classifier anticipates correctly and it is handled accordingly in the ALP. For now, the respective probabilities of the particular symbols are not considered in the learning process. However, when a classifier with PEEs anticipates correctly, the respective probability values in the PEEs are increased with the update parameter $b_p = 0.05$: $p = p + b_p(1 - p)$ and next the relative probability of each value in an attribute is recalculated dividing each probability by the sum of the probabilities in one attribute. A closer observation of the evolving classifier lists (not shown here) showed that the probabilities in the PEEs reflect the environmental settings accurately.

3.2 How and When to Enhance E

The enhancement in E can easily lead to an over-generalized rule. What about a classifier that is fairly general in C and predicts that basically anything can happen? Such a classifier would become reliable, since it always predicts correct changes. Thus, an enhancement in E must only occur if it is really necessary. Consequently, each classifier gets an additional boolean parameter ee (the 'enhance effects' parameter) which specifies if it must be enhanced in E. The idea is that when a classifier cannot be further differentiated in the conditions and still does not predict the correct effects all the time, it sets ee to true indicating that in the specified situation the execution of the specified action results in different effects. When a classifier is generated, ee is always set to $false$. Moreover, ee is set back to $false$ when the classifier gets an additional mark that differs from the current M (indicating that the classifier can be differentiated from this state). The parameter ee is set to $true$, when the classifier cl is (1) in the expected case, (2) it has a mark M, and (3) C cannot be further differentiated from M. Furthermore, in this case the classifier looks for another classifier cl_j in the current learning set that is ready to be joined. It must be assured that $ee_{cl_j} = true$, M_{cl_j} is not enhanced, and $M_{cl_j} = M_{cl}$. In this case a new classifier is generated that merges the two E parts to the above structure and combines the C parts so that the resulting C has #-symbols only in the attributes in that both parents have a #-symbols.

4 The Resulting Performance in Different Mazes

This section presents results from experiments using the ACS with PEEs in two two-dimensional mazes with additional misleading properties. The Woods1 environment was previously investigated with ZCS and XCS (Wilson, 1994, Wilson, 1995). The Maze4 environment was introduced by (Lanzi, 1997) for XCS. In the presented results we use the following parameters: $\rho = 1$, $b_q = b_r = 0.05$, $\gamma = 0.95$ and $p_x = 0.8$. Perceptions in the two mazes are coded as a string with eight attributes for the eight adjacent cells. $\sigma \in \{., O, F\}^8$ where a '.' represents an empty position, an 'O' an obstacle, and an 'F' a position with food. Executable actions are movements in the eight directions, $\alpha \in \{N, E, S, W, NE, SE, SW, NW\}$. All results are averaged over ten runs. After examining an example of the new mechanism in Woods1, we will give results without GA and in section 4.4 with GA application in both environments violating relevance as well as adding action-noise.

4.1 Performance in Woods1

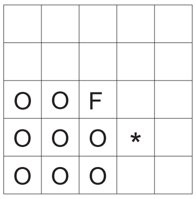

Figure 2: Woods1

Woods1 is very suitable to examine the behavior of a learning agent. It is a Markov environment where the accessibility of all states is equal. Figure 2 shows Woods1 with an animat indicated by the '*'. This section gives first an example of the mechanism associated with the PEEs. Next, it presents first results.

The animat at the indicated position perceives the situation $\sigma = (.....OOF)$. When adding an additional random attribute, the animat either perceives $\sigma = (.....OOF0)$ or $\sigma = (.....OOF1)$. At a particular time t the ACS may perceive $\sigma(t) = (.....OOF0)$ and move one step north ($\alpha(t) = N$). The consequent state is either $\sigma(t+1) = (.....OF.0)$ or $\sigma(t+1) = (.....OF.1)$. Both results will be experienced sometime resulting in two classifiers
$cl_1 = [\#\#\#\#\#\#OF\#] - N - [\#\#\#\#\#\#F.\#]$ and $cl_2 = [\#\#\#\#\#\#OF0] - N - [\#\#\#\#\#\#F.1]$ (showing $C - A - E$). cl_1 can still be applied in $\sigma = (.....OOF1)$. Thus, applying the marking mechanism a new classifier cl_3 will be formed where the #-symbol in the last attribute is specialized to 0. Now, cl_2 and cl_3 can only be applied in $\sigma = (.....OOF0)$ and will be eventually marked by the state itself, since they predict a result which occurs at a 50% chance. Thus, ee of both classifiers will be set to $true$ since there is no difference between the mark and the indicated state when applied successfully. Consequently, a new classifier will be formed that has the structure
$cl_4 = [\#\#\#\#\#\#OF0] - N - [\#\#\#\#\#\#F.\{(0, 0.5), (1, 0.5)\}]$.
This classifier now predicts correctly the deterministic changes due to the execution of N and further predicts that the last attribute will change to 1 with a 50% chance. Since this classifier always anticipates correctly, its quality q will increase over 90% and will consequently become part of the internal environmental representation.

Figure 3 shows the resulting performance in Woods1. As a comparison, also the learning curve without any non-determinism in the environment is shown. The correct anticipations measure is evaluated by considering in each situation in the environment each possible movement and checking if there is a reliable classifier that matches in the conditions and predicts the correct effects of the movement. The population size is the number of distinct classifiers in the population. When adding randomly changing attributes, the evolution of the whole internal environmental representation takes longer, but the ACS is able to build it completely. The population size grows to a relatively high level, when adding the random attributes as should be expected. In a maze with i random attributes one situation looks like 2^i different situations. Although the ACS is able to solve the effect-problem, it still needs to solve the puzzle for all possible situations, which can also be observed

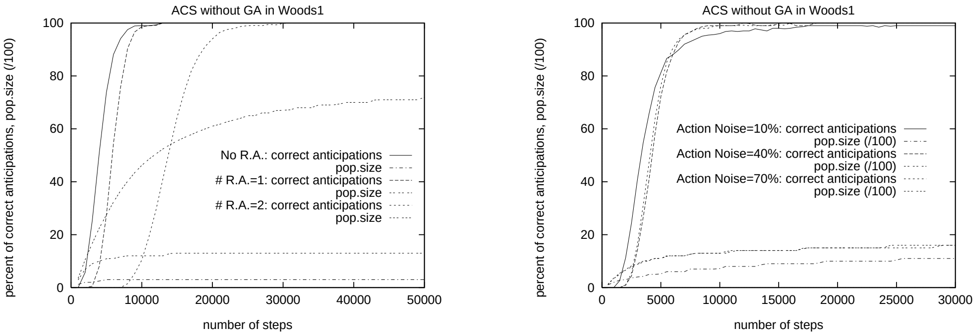

Figure 3: The ACS is able to learn the Woods1 environment with additional randomly changing attributes on the left hand side and action-noise on the right hand side due to the PEEs.

in the above example. Classifiers with different stages of PEEs increase the population size further.

Figure 3 shows results in Woods1 with action-noise p_n. With a probability of p_n an executed action does not lead to the intended position but to the position on the left or right. If a position is blocked, the action does not cause any effect. The figure reveals that the ACS is able to learn the relations. Interestingly, the best performance is not reached when the action-noise is on the lowest level but on the highest level. This is because when p_n is high, all three outcomes of the action have approximately the same probability. Thus, all three necessary classifiers evolve with the same probability and get joined quickly. When p_n is low, the classifiers that predict the improbable changes take longer to evolve and often get inadequate and thus deleted. Consequently, it takes longer to form classifiers with the adequate PEEs.

4.2 Performance in Maze4

O	O	O	O	O	O	O	O
O			O			F	O
O	O			O			O
O	O		O			O	O
O							O
O	O		O				O
O					O		O
O	O	O	O	O	O	O	O

Figure 4: Maze4

The Maze4 environment (Figure 4) is a different challenge for the ACS. While in the Woods1 environment every position is equally probable to access, in Maze4 the probabilities are distributed unequally. The whole area in the North-West quadrant of the maze is much less likely to be explored than the eastern area (because of the additional obstacles barring access to the North-West quadrant). Figure 5 shows the resulting performance in Maze4. The comparison of different numbers of random attributes shows again how the size of the population grows when additional random attributes are added. Furthermore, it shows how much longer it takes until a complete internal model is formed. As predicted, it is harder to achieve a complete knowledge since it takes much longer until all parts of the maze are experienced often enough. When adding action-noise to the environment, the same effect as in Woods1 can be observed. When p_n is low, the knowledge in the beginning increases faster but a 100% knowledge is hardly reached as the merging of classifiers is much more unlikely. When the noise level is on 70% the performance is the best, but also the population grows the most.

4.3 Action-Noise Combined with Random Attributes

So far, we have shown that the PEEs and the associated mechanism are able to detect attributes that change independently from their own actions as well as detect that an action can cause different effects with a certain probability. However, it still needs to be shown if the combination of the two sources of non-determinism can be detected. Is the mechanism able to focus on the relevant eight attributes and realize in the mean time that an action can cause three different effects? Figure 6

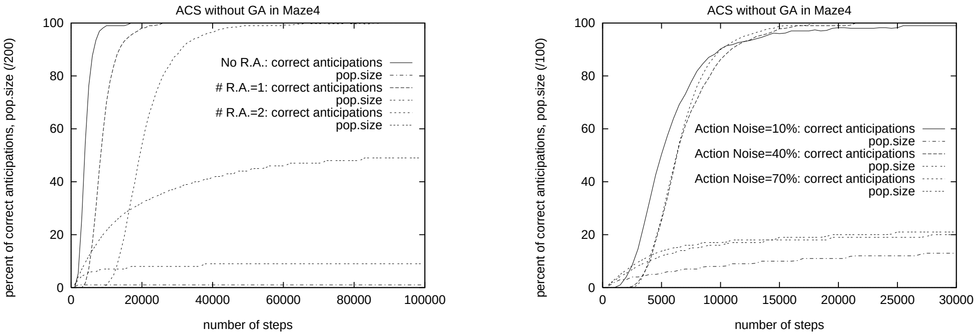

Figure 5: Also in Maze4 with additional random attributes on the left hand side and action-noise on the right hand side, the ACS is able to learn the internal environmental representation due to the PEEs.

shows that this is indeed possible. The ACS is able to form a complete internal model of the environment. It is forming classifiers that predict that after an action, the random attribute is going to change with an approximately 50% probability and furthermore that the action can cause three different effects. Maze4 again takes longer to be learned completely. However, as in all experiments with additional random attributes, the population size grows to a high level. In the next section we will show that the GA is able to decrease this population growth.

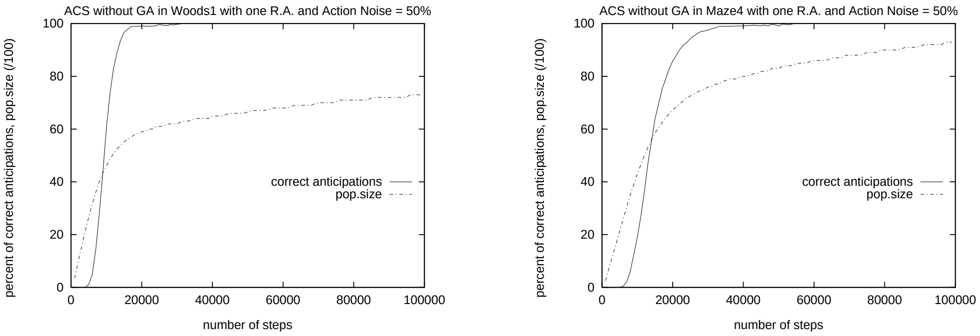

Figure 6: The ACS is also able to handle the combination of a random attribute and action-noise.

4.4 Combining the PEEs with the GA

In the experiments with random attributes, the population size increases with the number of random attributes. We claimed that there are two reasons for this behavior: (1) classifiers evolve that predict a part of the possible changes and stay as suboptimal classifiers in the population, and (2) classifiers that predict all possible changes specify one specific constellation of the random attributes in the conditions. Consequently, a classifier is needed for each combination of random attributes.

Overcoming this difficulty is exactly what the GA is meant for. The genetic generalization pressure can form an attention spot on those attributes which are relevant in the conditions. The ALP cannot realize that the result does not depend on the random attributes but the GA can. Generalizing the good classifiers by mutation leads to classifiers that match more often in each situation. Eventually, classifiers will evolve that completely ignore the random attributes in the conditions and predict that they will either change or stay the same.

Figure 7 shows that the ACS with GA is really able to generate this classifiers. The change in the population size shows that the ACS with GA is exploring early in the run. Once the knowledge increases, the population size starts to decrease and converges to the above-mentioned classifiers. In Woods1 with two additional random attributes, this results in a complete environmental representation with a ten times smaller population size. In Woods1 with three additional random attributes the population size in the ACS without GA increases to a level of 25000 classifiers while with GA the size of the population never reaches a size of 5000 and converges in the end to less than 600 classifiers – more than 40 times less than without GA. Also in Maze4 the population size decreases substantially. However, we can observe that the knowledge does not reach a 100% knowledge level permanently. This is the case because of the unequal frequencies of experiencing each of the states in Maze4. Since the ACS experiences part of the states much more often, the GA generalizes over this subset of states and sometimes drives out part of the classifiers necessary for the seldom experienced states. Nevertheless, the environmental representation is still near perfect.

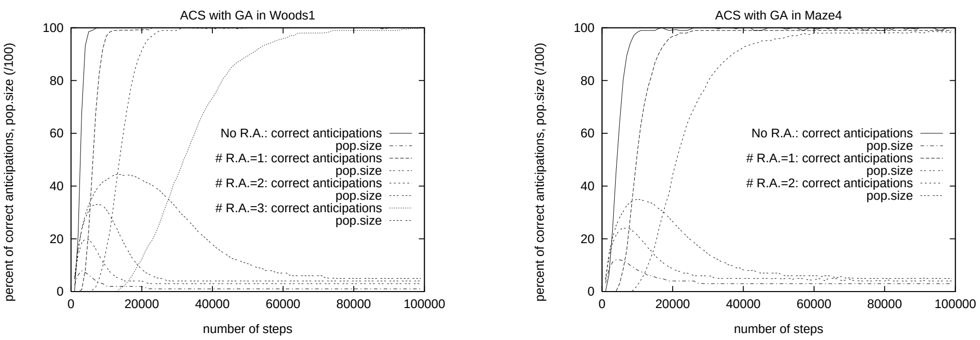

Figure 7: When the ACS with PEEs and GA is applied in Woods1 and Maze4 with additional random attributes, the result is a great decrease in the size of the classifier list and in no decrease in the performance.

When we add the GA in the mazes with action-noise the knowledge does not completely stay at 100% (Figure 8). A closer look at the classifier list in this case shows that classifiers evolve that match in different situations and predict all possible changes in those situations. The GA generalizes classifiers that have PEEs. Accidently, such a classifier can now also predict a part or even all possible changes in another situation. If it predicts a part of another situation, the ALP will generate a more specialized classifier that suddenly only matches in the new situation but actually predicts more changes than are possible in this situation. However, for now, the GA works completely independently from the PEEs. We have shown that the GA is able to substantially decrease the size of the population while the formation of the environmental representation stays very close to perfect.

5 Discussion

Although a detailed observation of the evolving classifier lists (not shown herein) revealed that the probability distribution in the PEEs accurately reflect the current environmental settings, the usefulness of this distribution was not further investigated as yet. Enhancements in the reinforcement learning mechanisms as well as in the cognitive capabilities of the ACS are imaginable. The advantages and limits of such approaches need to be observed in detail.

Moreover, as already mentioned in Butz, Goldberg, and Stolzmann (1999), there are other possibilities of representing PEEs. One other possibility is to allow different effect strings in one classifier concretely specifying the distinct possible changes. Especially, a hybrid of the two approaches sounds reasonable, which consequently should form classifiers that predict a 50/50 chance

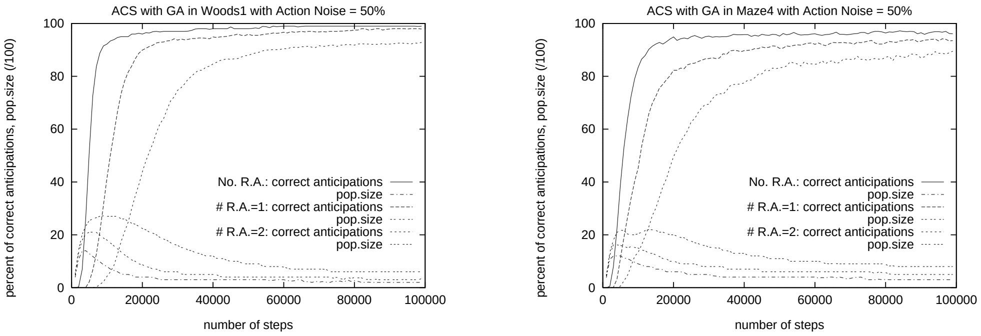

Figure 8: Applying the ACS with PEEs and GA in Woods1 and Maze4 with additional noise results again in a decrease in size but also a slight decrease in the performance.

in the random attributes in each effect-string. In the example of Section 4.1 with one additional attribute and 50% noise the perfect classifier would look like this:

$$[\#\#\#\#\#\#\#F\#] - N - \begin{array}{l} 0.25 : [\#\#\#\#O\#\#.\{(0,0.5),(1,0.5)\}] \\ 0.50 : [\#\#\#\#\#\#F.\{(0,0.5),(1,0.5)\}] \\ 0.25 : [\#\#OO\#...\{(0,0.5),(1,0.5)\}] \end{array}$$

Future research will show if the ACS is able to form such classifiers reliably.

Nonetheless, this paper has demonstrated a first approach in the ACS to handle non-determinism in an environment with PEEs. It enabled the ACS to handle randomly changing attributes as well as action-noise. When additionally applying a GA, the ACS was able to solve the tasks with a substantial decrease in the population size. However, a slight problem of over-generalization was observed.

In order to prevent the over-generalization of PEEs in the application of the ACS with GA in an environment with action-noise, the GA could consider the amount of enhancement in the PEEs in some way. It should prefer the classifiers that are accurate, maximally general but not over-general in the effect parts for reproduction and the over-general classifiers in the effect parts for deletion.

Considering the observed over-generalization of conditions in environments with an unequal probability distribution of experiencing states, we first need to reconsider what is our main goal. When we want to build a system which behaves similar to real animals, then this observation should actually not bother us too much. The loss of knowledge is similar to the process of forgetting special exceptions that occur more seldom. Moreover, the present near-complete internal environmental representation could directly be used to overcome the loss of knowledge. By introducing a new cognitive process, the ACS could directly explore regions in an environment that it did not experience for a longer time (a first approach is given in Butz & Stolzmann, 1999). On the other hand, if we view the ACS as a traditional LCS, we could approach the problem with default hierarchies (Holland, Holyoak, Nisbett, & Thagard, 1986). This should result in an implicit formation of more specialized exception rules that overrule more general rules in the special cases. A modification of the GA as well as a hierarchy in the usage of the environmental representation would be necessary to realize such an approach.

Nevertheless, without any enhancement in the GA and thus a complete independence from the PEEs, the ACS was still able to form a near-optimal representation of the environment with a much smaller population size. In the application with only additional randomly changing attributes the environmental representation even got perfect and the population size decreased enormously. Thus, the ACS is now able to handle non-deterministic environments generating classifiers that are accurate and maximally general. More research is needed to reveal in what other types of environments the PEEs are useful as well as where they are limited.

6 Acknowledgments

The authors would like to thank Dimitri Knjazew, Martin Pelikan, and Stewart Wilson for valuable discussions and useful comments as well as the Automated Learning Group of NCSA at the University of Illinois in Urbana-Champaign for their support.

The work was sponsored by the Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grant F49620-97-1-0050. Research funding for this work was also provided by the National Science Foundation under grant DMI-9908252. Support was also provided by a grant from the U. S. Army Research Laboratory under the Federated Laboratory Program, Cooperative Agreement DAAL01-96-2-0003. The U. S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon. Additionally, research funding was provided by the German Research Foundation DFG.

The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Office of Scientific Research, the National Science Foundation, the U. S. Army, or the U. S. Government.

References

Butz, M. V., Goldberg, D. E., & Stolzmann, W. (1999). *New challenges for an Anticipatory Classifier System: Hard problems and possible solutions* (IlliGAL report 99019). University of Illinois at Urbana-Champaign: Illinois Genetic Algorithms Laboratory.

Butz, M. V., Goldberg, D. E., & Stolzmann, W. (2000). Introducing a genetic generalization pressure to the anticipatory classifier system: Part 1 - theoretical approach. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2000).* In press (also available as Technical Report 2000005 at the Illinois Genetic Algorithms Laboratory).

Butz, M. V., & Stolzmann, W. (1999). Action-planning in Anticipatory Classifier Systems. In *2.International Workshop on Learning Classifier Systems (2.IWLCS) on the Genetic and Evolutionary Computation Conference (GECCO-99)* (pp. 242–249). Orlando, Florida: Morgan Kaufmann.

Holland, J. H., Holyoak, K. J., Nisbett, R. E., & Thagard, P. R. (1986). *Induction: Processes of inference, learning, and discovery.*

Lanzi, P. L. (1997). A study of the generalization capabilities of XCS. In Baeck, T. (Ed.), *Proceedings of the Seventh International Conference on Genetic Algorithm* (pp. 418–425). San Francisco, California: Morgan Kaufmann.

Lanzi, P. L. (1999). An extension to the XCS classifier system for stochastic environments. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-99)* (pp. 353–360). Orlando, Florida: Morgan Kaufmann.

Stolzmann, W. (1997). *Antizipative Classifier Systems [Anticipatory Classifier Systems].* Osnabrueck, Germany: Shaker Verlag, Aachen, Germany.

Stolzmann, W. (1998). Anticipatory Classifier Systems. In *Genetic Programming '98* (pp. 658–664). University of Wisconsin, Madison, Wisconsin: Morgan Kaufmann.

Stolzmann, W. (1999). Latent learning in Khepera robots with Anticipatory Classifier Systems. In *2.International Workshop on Learning Classifier Systems (2.IWLCS) on the Genetic and Evolutionary Computation Conference (GECCO-99)* (pp. 290–297). Orlando, Florida: Morgan Kaufmann.

Stolzmann, W., Butz, M. V., Hoffmann, J., & Goldberg, D. E. (2000). *First cognitive capabilities in the anticipatory classifier system* (IlliGAL report 2000008). University of Illinois at Urbana-Champaign: Illinois Genetic Algorithms Laboratory.

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), 149–175.

Wilson, S. W. (1998). Generalization in the XCS classifier system. In Koza, J. R. e. a. (Ed.), *Genetic Programming 1998: Proceedings of the third annual conference* (pp. 665–674). San Francisco: Morgan Kaufmann.