An Evolutionary Approach for Generating a Learning Classifier System Reward Policy: Review and Prospects

Tiago Sepúlveda and Mário Rui Gomes

IST – Instituto Superior Técnico
INESC – Instituto de Engenharia de Sistemas e Computadores
R. Alves Redol, nº9, 6º Andar, 1000-029 LISBOA
{tiago.sepulveda,mrg}@inesc.pt

Abstract

In this paper we review the evolutionary approach we proposed in previously published papers, regarding the emergence of a Learning Classifier System (LCS) reward policy. The idea behind our approach is to induce the emergence of a LCS reward policy, through the evolution of a population of LCS based agents. The present review intends to shed light on some aspects that were not sufficiently emphasized in previous papers and, on other hand, to prospect future work regarding this approach. First, we describe a simple, but generic architecture of an evolutive LCS based agent. The couple of modules constituting the architecture are a (LCS based) control model, generating the agent behaviour, and a biological model regulating the biological aspects of the agent life. Second, we perform an analysis of the factors influencing the outcome of reward policy evolution, like the reward regimes to adopt, or the genetic operators that one should use. Finally, we evaluate the requirements to extend our approach to Special Classifier Systems (XCS) based evolutive agents.

Keywords: Classifier Systems, Evolution, Reward Policy and Learning.

1 Introduction

Learning Classifier Systems (LCSs) are a methodology based on a genetic paradigm, that is able to find solutions for problems involving a large search space. LCSs are a type of parallel production system where rules – classifiers, are evolved, using a genetic algorithm, in order to maximize the expected system *payoff*.

The unit of a LCS is the classifier; a classifier is a codification of an IF-THEN rule. The classifier is usually encoded in a binary alphabet, enriched with a universal symbol (# - *don't care*) in order to enable the expression of clusters of rules in a single classifier. For instance, the string *01:1* is an example of a classifier that could express the following behaviour rule:

IF lion-right and grass-left THEN left

The process of classifiers optimisation relies on the categorization of the system classifiers according their performance. The categorization, and therefore performance measurement, is based on a variable assigned to each classifier, called *utility*. We call the set of criteria establishing the LCS performance metric, the LCS *reward policy*.

From the above requirements, the LCS architectural cycles (fig. 1) appear naturally. They can be summarized in the following points:

- *Performance cycle*: world events are translated in LCS messages and placed in the system message set; classifiers are selected accordingly the current message set; selected classifier actions are performed.
- *Reward cycle*: the LCS reward system generates a numerical payoff based on the evaluation of the performed action (based on the system reward policy); payoff is added to the utility of the classifier(s) proposing the referred action.
- *Revising cycle*: the best LCS classifiers are chosen; these classifiers are bred; the offspring replaces classifiers with poorer performance. The period of the revising cycle is larger than the period of the previous cycles.

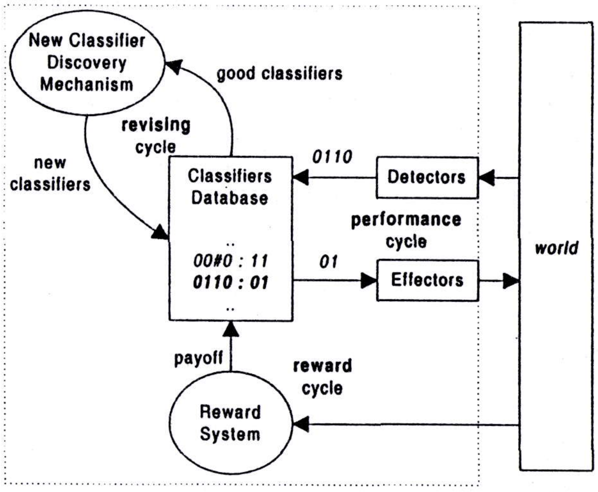


Fig.1: LCS architecture.

The interaction between the reward and revising cycle is the support of the LCS learning process. Therefore, we refer to this couple of cycles as the LCS *learning cycles*.

This brief description enables us to extract the most important features regarding LCS:

1. Independence of the LCS classifier database from the environment the system faces.
2. Domain knowledge must be embedded in the LCS reward policy.

This couple of features conditions the domain of application of LCSs. On one side, the independence of the LCS internal model from the domain, enables LCS to be a horizontal approach, in the sense that it can be applied in a number of distinct scientific domains, such as Robotics (Colombetti, 1993), Economy (Mitlohner, 1996) or Simulation (Smith, 1999). On the other side, the strict requirement regarding the reward policy limits the vertical application of LCSs, meaning that LCSs cannot be applied to domains or problems where it is hard to find a good reward policy.

We believe however that, like LCS simplified the insertion of domain knowledge taking it from the design of internal models to the reward policy design, it is possible to take this process farer and simplify also the task of designing the reward policy. This would enable the deepening of the LCS "vertical" scope. The approach we took, which was already presented in previously published papers (Sepúlveda, 1999, 2000), was to use evolution and Artificial Life (AL) environments to induce the emergence of a LCS reward policy.

In this paper we review some aspects of our approach, generalizing it and make some considerations regarding its extension to the Special Classifier System (XCS) framework. The paper follows with a section where we present previous work focusing the LCS reward system. Then, on section 3, we review the approach we propose regarding the emergence of a reward policy, clarify some of the approach most relevant aspects and present a general model of an evolutive agent. On section 4, we emphasize the questions that still remain opened. On section 5, we analyse the extension of our framework to XCSs agents. Finally, conclusions are drawn in section 6.

2 Related work

Issues regarding credit (or payoff) assignment were focused since the beginning of LCS research. When Holland proposed the LCS (Holland, 1992), he also described an algorithm aimed to assign accurately payoff in environments characterized by intermittent reward. This algorithm, named after the metaphor from which it was built – Bucket Brigade (BB), relied on a mechanism of utility[1] passing, which, cycle after cycle, would theoretically produce the emergence of chains of classifiers setting the stage to enable other classifiers to get system payoff.

Even in the latest developments of LCS research, the issue of credit assignment has been taken care of. In the XCS framework, for instance, it was defined a mechanism very similar to the BB. In a XCS one must define a discount factor, a value determining

[1] In Holland's classifier systems: strength.

the fraction of classifier fitness to be added to the classifiers that have acted previously, much in the spirit of the BB algorithm.

Besides credit assignment, there is also a less remembered research issue regarding the LCS reward system – designing a reward policy, whose literature is very scarce. As far as we know, the only research line that dealt with this issue, was the work performed by Dorigo and Colombetti (1994). These researchers were the first to address the issue of suitability of LCS reward policies, showing experimentally that the performance of a LCS could vary accordingly the reward policy defined. In their paper, they stood that the *reward-the-result* policy was not always the best solution to generate LCS payoff, because it implied the design of a fitness function that is not always easy to devise. Additionally, they emphasized that, in dynamic environments, a pre-defined fitness function often did not performed well or did not meet the experimenters needs. To solve this problem, they introduced the system trainer figure, a metaphor of an external entity supervising the learning process. This solution however implied the existence of some supra-environmental entity (human or synthetic) to provide classifiers payoff.

3 The Emergence of a Reward Policy: a Review

The main goal of LCS research has been to find solutions for increasing the LCS learning efficiency, trying this way to broaden the application spectrum of these systems. There exist however issues, although preventing the application of LCS to more domains, did not receive attention from the LCS research community. One of the most relevant issues, in this context, is the design of LCS reward policies regarding environments involving a high degree of complexity. One typical example of a system requiring the development of a complex reward policy, is the virtual soccer game developed by Sanza (1999), where the multitude of sub-problems faced by a LCS based player, in a soccer game, had to be correlated within a weighted fitness function.

We defend that it is required a new approach to develop LCS reward policies, in order to solve the kind of problems that became patent in the previous example. The approach we proposed consisted in using a simulated evolution process aiming to induce the emergence of a reward policy in a population of LCS agents.

3.1 Integrating Evolution and LCS

Using evolution in conjunction with LCSs requires a careful analysis on the characteristics of both approaches in order to find the best way to take advantage of the synergy that may be established between them.

Regarding the establishment of a evolution process, it is required the existence of both a mechanism to increase the evolution subject variation and a corresponding selection operator, choosing the subjects that seem to lead to promising (evolutionary) paths. The conjunction of this couple of operators assures that a given group of individuals (a population) will evolve. The direction towards which the population evolves is greatly dependent on the type of filter that the selection operator implements. The selection operator should, therefore, to take into account the performance exhibited by the evolution subject on the environment it is facing.

Taking a selection mechanism based on *natural selection* (our approach) requires the design of a biological model supporting the implementation of this specific operator, since natural selection is a operator selecting individuals based on their reproductive success.

With respect to LCSs, as we have seen, LCSs are essentially a methodology to generate adaptive agent behaviour. This characteristic makes them to be an option to considerer regarding the implementation of an agent control model.

The features of both evolution and LCSs enable us to define a general agent model tailored to AL environments. This model (fig. 2), is then composed by:

- A LCS based control model, generating the host agent behaviour; and
- A biological model, providing support for the natural selection operator.

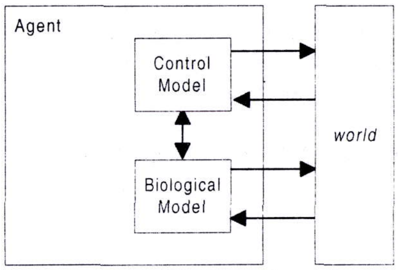


Fig. 2: Model of an evolutive agent.

3.2 The Emergence of a Reward Policy

The goal of our approach is to take the evolutionary model of figure 2 and induce the emergence of a reward policy in a population of LCS based evolutive agents. To accomplish this goal the general evolutionary model we presented should be further specified.

Regarding the biological model, there are no further considerations to make at this point, because the model details depend on the particular environment where the population of agents evolves.

Regarding the control model however it is possible to deepen the agent model, since the goal we are pursuing– emergence of a reward policy, imposes requirements that can be generalized regardless of the particular environment where our approach is used.

If we want to evolve a LCS reward policy, the reward policy should be the only source of agent diversity. If this condition is not met, there is the risk that a group of agents evolves, forming a dominant group due to a characteristic in which we are not focused. This implies that the operators who are responsible for the increase of LCS agent variation should act on the reward policy only. Therefore, the agent control model, the host of the LCS reward policy, should be designed taking this requirement in mind.

The second question affecting the agent control model is the choice of the most suitable way to express a LCS reward policy. If we require genetic operators to act on a LCS reward policy, as we mentioned, something must be said about the structure of the reward policy (what we call the *reward regime*). Should the criteria establishing the reward policy, be expressed by a fitness function? Should this function be linear? What reward policy parameters should be chosen to be evolved? Is there other ways to express a reward policy, other reward regimes, that might suit evolution?

We believed the issues underlying these questions deserved a careful study. The study was carried out in *woods*-type (Wilson, 1994) environment that we called *Saavana* (fig. 3). In this environment, a population of synthetic LCS based antelopes was subjected to evolution in order to improve the antelopes ability to increase internal energy (*E*).

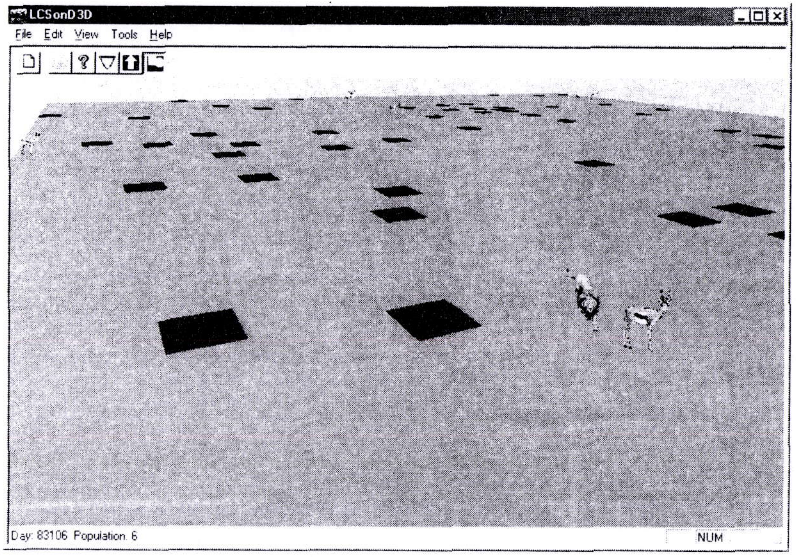


Fig. 3: *Saavana*'s snapshot.

3.2.1 Case Study Description

The antelopes biological model relied solely on the value of individual internal energy. The internal energy value could increase, if an antelope ate food cells, or decrease, if an antelope could not manage escaping the lion that also inhabited the environment. The antelopes died when their energy level reached a minimum value. Breeding occurred when the level of the antelope internal energy reached an upper threshold.

The core of the antelope control model was a LCS (called *coordinator LCS*), deciding the behaviour the antelope should follow: eating or escaping. The decision was based on the couple of presence bits delivered by the antelope sensor modules – one sensing the presence of food cells in the neighbouring cells, the other sensing the lion position. The antelope architecture is depicted in fig. 4.

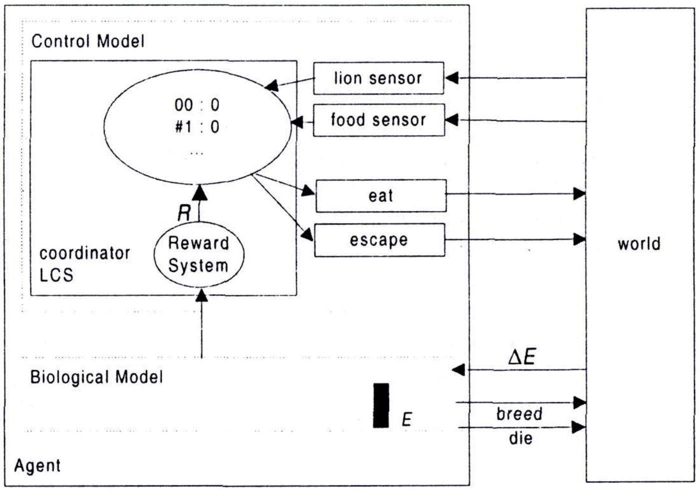


Fig. 4: LCS based antelope architecture.

The coordinator LCS is the only adaptive structure of the whole architecture, *i. e.*, the only source of individual variation in the antelope population, corresponding therefore to the guidelines stated above. Behaviours (*eat* and *escape*) and sensors (*lion sensor* and *food sensor*) are similar in all antelopes.

Within the lifetime of an antelope the coordinator LCS performs its usual cycles (performance, reward and revising), trying to find the *strategy*[2] that optimises the system payoff (*R*). Notice that although the set of reward policies existing in the population can change due to mutation when an antelope breeds, a particular reward

[2] Strategy is the set of classifiers existing in the LCS classifiers database.

policy remains fixed within an individual's lifetime assuring a coherent evaluation of the agent decisions.

3.2.2 Results

Concerning the case study results, which can be seen in further detail in (Sepúlveda 1999, 2000), the first thing worthwhile mentioning is that it was demonstrated that a population of virtual antelopes could develop good coordinator LCS reward policies. This statement is corroborated by the comparison of the antelopes performance measurements taken from the traditional "hand-designed" fitness function reward regime (eq. 1), to the measurements taken from the couple of evolutive reward regimes experimented.

$$R = C_1 * \Delta E + C_2 \tag{1}$$

This is a very important result because it allows LCS designers to provide the parameters that must be accounted for in the calculation of the system payoff, without having to specify the relationships between them. Evolution takes care of it.

Besides this result, our primary goal, the study helped us clarify some aspects regarding the choice of reward regimes. Two evolutive learning regimes were designed and evaluated.

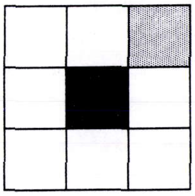
In the first one – *parameterised learning*, the numerical parameters of the coordinator LCS fitness function (C_1 and C_2, see eq.1) were coded in an 8-bit genome. This genome was mutated whenever the virtual antelopes bred, aiming, this way, to optimise the parameters values.

The second evolutive reward regime tested – *evolvable learning*, was implemented using an additional classifier system (*reward LCS*), where individual classifiers represented specific reward criterion. The classifiers condition coded the values of the parameters we thought it were important to relate to the payoff value. The reward classifiers action proposed the payoff value to deliver to the coordinator LCS (fig. 5). Both learning cycles of the reward LCS were disabled maintaining a stable reward strategy throughout an antelope life. When an antelope breeds, the reward LCS strategy is copied onto its offspring. After the classifiers copy, a fraction of the offspring reward classifiers is modified by the action of the mutation operator.

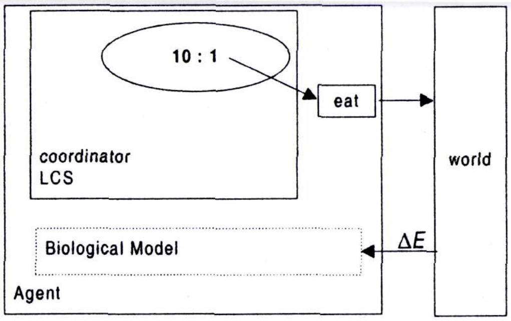
$$\underbrace{00}_{\text{IF}E\in[0.0,100.0],}\quad \underbrace{11}_{\Delta E\in[0.0,10.0]}\ :\ \underbrace{10}_{\text{THEN}R=5.0}$$

Fig. 5: Example of a reward LCS classifier.

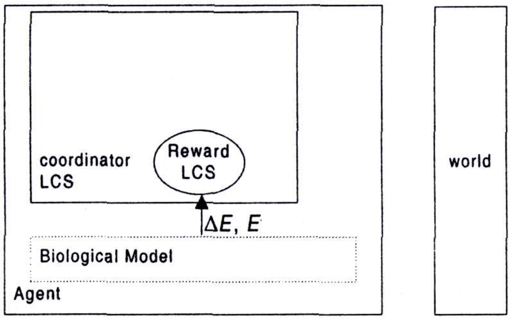
The interaction between the coordinator LCS and the reward LCS can be best understood following an example:



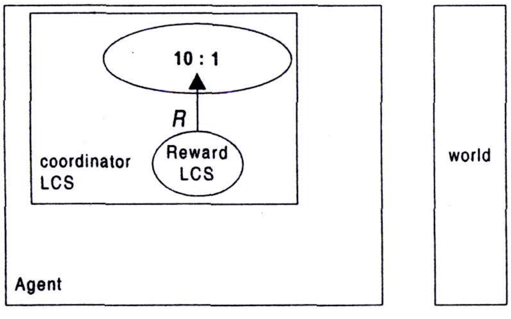
1. An antelope detects a food cell in its surroundings. The presence bits are computed and the corresponding message is sent to the coordinator LCS



2. The coordinator LCS selects the classifier determining the antelope behaviour. The eating behaviour is performed and the food is collected, ending the coordinator LCS performance cycle.



3. The coordinator LCS reward cycle starts. E and ΔE are encoded in reward LCS messages.



4. A reward classifier is selected. The payoff value expressed in the selected classifier action is assigned to the coordinator classifier that triggered the eat behaviour.

The results showed that the reward regime producing better results was the evolvable learning regime. This regime, besides producing better coordination LCS strategies, evolved quicker than the parameterised learning regime.

The determinant factor behind these results is the *granularity* provided by the reward LCS. Granularity can be viewed as a property reflecting the degree of independence between each reward criterion. In this sense, continuous fitness functions (like eq. 1) are non-granular, since they force a relationship between payoff values corresponding to situations that might not be correlated.

4 Questions to answer

The results provided by our prior work seem to have validated the use of an evolutionary approach aimed to generate LCS reward policies. But our case study was very simple, leaving yet many questions unanswered. On the next sub-sections we will try to clarify some of the issues that were not sufficiently addressed in previous papers and that underlie many of the approach relevant questions.

4.1 Reward Regimes

As we have emphasized earlier, the reward regime that seems to guarantee a better learning efficiency is the evolvable learning reward regime. From the results gathered, we assume that a reward regime with some degree of granularity is preferable to a continuous (or global) one.

We think however that it is not necessary to define an additional LCS to support a granular reward regime, like we did in the case of the evolvable learning regime. We did it because, from our point of view, it was easier to introduce an additional LCS than other kind of data structure. The LCS is not needed because none of the so-called learning cycles are used, since it is not required to change the reward policy during an individual lifetime.

Apparently, we just need a data structure able to support a table of independent reward criterion. But, if we want to define a suitable evolutive reward regime, there are evidences pointing to the inclusion of some LCS features in a well-designed granular reward regime.

One is the ability to express clusters of rules. Although it is advantageous to have several reward criteria not related with each another, there are cases in which different events might be payoff-equivalent. A classifier expresses this situation easily by means of the LCS alphabet universal symbols.

Other is the LCS ability to cover inputs. If the reward policy is supposed to provide a payoff when a range of events occurs, then there either must exist criterion covering the whole event range, or it must exist a mechanism (like the LCS cover operator) able to generate the reward criterion that may be missing.

Concluding the discussion concerning the reward regimes, we can say that a promising reward regime should be granular, able to express fitness regularities and that it should be provided a reward regime cover operator. Fitness functions do not seem to provide efficient reward regimes, because they require the system designer to provide a clue about the actual expression of the function (continuous or non-continuous, linear, quadratic, etc). Other aspects related to reward regimes deserve more analysis. In particular, one of the most pertinent is the study of the relationships that seem to exist between the agent biological model and the reward regime.

4.2 Evolution Parameters

The paths followed by the evolution process are intimately linked to the parameterisation an individual biological aspects. We adopted the simpler evolutionary model guaranteeing us the viability of our approach. The evolutionary model main design options were: asexual reproduction, mutation and natural selection.

The choice between sexual or asexual reproduction determines in great extent the genetic operators to be used. We feel that sexual reproduction can bring benefits because this option enables the application of the crossover operator to parents reward policies.

Another relevant aspect related also with reproduction is the way to initialise the internal model of a new LCS agent. One hypothesis is to initialise it randomly; the traditional way of initialising a LCS. Another hypothesis is to copy the parent internal model to its offspring, consubstantiating this way a process of social transmission, which can be viewed like a kind of cultural or social legacy. In our case study, it was used the social transmission hypothesis. The impact caused in the experiment results, by taking this option, was not assessed.

From the evolutive point of view, the most important attributes of an individual are the ones conditioning natural selection. Many of these attributes are very dependent on the application domain and therefore it is hard to provide general guidelines to their parameterisation. But if we intend to induce the emergence of a reward policy, there is a dimension of the LCS agent "life" deserving to be carefully analysed: time.

The implications of time on the LCS agent "life" are related to the poor performance exhibited by LCS agents during the first simulation cycles after their generation. The low performance level exhibited during the LCS agent "infancy", corresponds to the time span needed for a LCS to find a set of classifiers optimising the system payoff (the

LCS *learning inertia*). The agent still has not learned (through the reward cycle) and discovered (through the revising cycle) the rules, which optimise the system payoff. Therefore, there is no point in allowing the agents to breed as soon as they are born since the performance that they might exhibit is not related to the "quality" of their leaning process. So, when we think on establishing the number of cycles that a LCS agent must perform before it is able to breed (its *sexual maturity*), the LCS learning inertia is a factor to be considered.

Regardless of the design decisions concerning evolution we have briefly described, there is still much work to be done in order to tune the parameterisation of the genetic operators. If the reward system is granular, how should mutation be applied? For instance, in our case, where the reward regime was based in LCS, we have mutated a fraction of the reward LCS classifiers when an antelope bred, promoting this way the exploration of the reward policy search space. But other parameterisation criteria could be conceived. Regarding crossover (we did not use it in our case study), there is an additional factor to deal - maintaining the reward policy coherent, *i.e.*, joining reward criteria covering the same inputs leaves without response other type of situations. One idea to overcome this situation is to provide a semantic for the reward regime and then to swap equivalent reward policy "pieces".

5 Extending the approach to XCS

One point that we wished to attend when we performed our case study was to assess the application of our approach to other types of classifier systems and, in particular, to XCS. The interest on XCS is due to the body of theoretic work developed for these systems, which can help us solving many of the problems we have mentioned so far.

We think that the extension of our approach to XCS agents would not present any trouble (we hope to do it in the near future). The XCS definition of utility, which is different form the one defined in the scope of standard LCS, does not bother us, since it does not affect the output and structure of the system reward policy. The structure of the XCS reward system is similar to the standard LCS reward system, although the modifications performed in the XCS reward system description (including explicitly the requirement of the reward system to provide an end-of-problem flag). The emphasis on the XCS fitness landscape regularities even seems to suit better our approach.

One of the questions that might be answered using the XCS framework is the one presented earlier regarding agent sexual maturity. Due to the similarity between Q-learning and XCS, there is hope that in the near future it will be found the number of cycles that a XCS will take to converge to a solution. This result would solve the problem regarding the definition of a XCS agent sexual maturity and would certainly provide a guideline, regarding the parameterisation of this variable in other types of LCS based agents.

6 Conclusion

In this paper we reviewed the evolutionary approach we proposed in previous papers regarding the emergence of a LCS reward policy. We formalized some aspects of our approach, proposing a general model for an LCS based agent. Additionally, we identified areas that are worthwhile pursuing in terms of research; one is the area concerning the search for reward regimes supporting the evolution a LCS reward policy; other is concerned with the parameterisation of the evolution process itself. In both areas the main problems were analysed and, either specific solutions or general research guidelines were proposed.

Finally, we discussed the extension of this approach to XCS agents and defended that the extension not only can be performed straightforwardly, but also that it can provide answers to questions that remain open, like the definition of the sexual maturity of an evolutive agent.

Although the infancy of the approach, we think, based on the results gathered so far, that the use of evolution to tune a LCS reward policy is a path worthwhile to explore. We hope that this paper becomes a map for those who want to follow it.

7 Acknowledgments

The work we presented was sponsored by the PRAXIS XXI program (BD/9375), of the "Fundação para a Ciência e Tecnologia".

8 References

Colombetti Marco and Marco Dorigo (1993). "Learning to Control an Autonomous Robot by Distributed Genetic Algorithms", Proceedings of From Animals to Animats, Second International Conference on Simulation of Adaptable Behavior (SAB92), pp. 305-312, J.A. Meyer, H. Roitblat and S. Wilson, MIT Press.

Dorigo Marco and Marco Colombetti (1994). "The Role of the Trainer in Reinforcement Learning", Proceedings of MLC-COLT '94 Workshop on Robot Learning, pp. 37-45, S.Mahadevan et al. Ed.

Holland John (1992), *Adaptation in Natural and Artificial Systems*, 1992 Edition. MIT Press.

Mitlohner Johann (1996). "Classifier Systems and Economic Modeling", APL Quote Quad, 26(4), pp.77-86.

Sanza Cédric, C. Destruel and Y. Duthen (1999), "Learning in Real-time Environment Based on Classifier Systems", Proceedings of the 7th International conference in Central Europe on Computer Graphics, Visualization and Interactive Digital Media'99, Vol. 2, pp. 440-447, V. Skala Ed..

Sepúlveda Tiago and Mário Rui Gomes (1999). "Evolving a Classifier Reward Policy", Proceedings of the 5th International Symposium on Artificial Life and Robotics, , Vol. 1, pp. 264-267, M. Sugisaka and H. Tanaka Ed.

Sepúlveda Tiago and Mário Rui Gomes (2000). "Am I Right? Emergence of a Reward Policy for a Learning Classifier System", Proceedings of 15th European Meeting on Cybernetics and Systems Research – 2000, Vol. 2, pp. 484-488, Robert Trappl Ed..

Smith Robert, B. A. Dike, B. Ravichandran, A El-Fallah and R. K. Mehra (1999) "The Fighter Aircraft LCS: A case of Different LCS Goals and Techniques", Learning Classifier Systems - From Foundations to Applications, LNCS Vol. 1813, pp. 283-300, Springer.

Wilson Stewart (1994). "ZCS: A Zeroth Level Classifier System", Evolutionary Computation, Vol. 2, 1, pp. 1-18.

Wilson Stewart (1999). "State of XCS Classifiers System Research", Learning Classifier Systems - From Foundations to Applications, LNCS Vol. 1813, pp. 63-80, Springer.