# Bounding Learning Time in XCS

Martin V. Butz, David E. Goldberg, and Pier Luca Lanzi

Illinois Genetic Algorithms Laboratory (IlliGAL)
University of Illinois at Urbana-Champaign
Urbana, IL, 61801
{butz,deg,lanzi}@illigal.ge.uiuc.edu

**Abstract.** It has been shown empirically that the XCS classifier system solves typical classification problems in a machine learning competitive way. However, until now, no learning time estimate has been derived analytically for the system. This paper introduces a time estimate that bounds the learning time of XCS until maximally accurate classifiers are found. We assume a domino convergence model in which each attribute is successively specialized to the correct value. It is shown that learning time in XCS scales polynomially in problem length and problem complexity and thus in a machine learning competitive way.

## 1 Introduction

Although the learning classifier system framework was proposed more than thirty years ago [1], the theoretical understanding is still rather sparse. Due to the complex interaction of several adaptive mechanisms, including the evolutionary learning method, the credit assignment mechanism, and the distributed problem representation, the exact analysis of the systems is hard. Recently, two classifier systems have reached most attention in the literature: the strength-based classifier system ZCS [2,3] and the accuracy-based classifier system XCS [4,5].

The XCS classifier system has shown to solve typical classification problems competitively to other machine learning algorithms [6,7]. Also the theoretic understanding is increasing [8,9,5,10]. In particular, problem bounds have been found that bound the minimal population size in order to assure that the evolutionary algorithm applies (the *covering challenge*), that necessary minimal order schemata are available (the *schema challenge*), that the found subsolutions (or schemata) have the chance to reproduce before being deleted (the *reproductive opportunity bound*), and that the different subsolutions have enough support to avoid the loss of a niche (the *support challenge*, in preparation). However, until now, no estimate has been found that approximates the time until complete knowledge is reached.

Specifically, we are interested in how many learning steps XCS needs to evolve the intended accurate, maximally general model of the applied task. Assuming that all the other challenges are met, we can estimate how long it takes to discover successively better classifiers until the maximally general, accurate classifiers are found. This paper focuses on mutation-driven evolution, taking

K. Deb et al. (Eds.): GECCO 2004, LNCS 3103, pp. 739–750, 2004.

into account time until reproduction and time until generation of the next best classifier via mutation. The experimental study confirms that XCS learning time is approximated by the derived time bound. The bound shows that learning time scales polynomially in the problem length and exponentially in the order of the problem (polynomially in problem complexity).

The next section gives an overview of the XCS classifier system and existing theory. Given that all other bounds are met, we then derive the time bound for XCS learning. The experimental study that follows shows that XCS learning scales in the specified time bound. Moreover, several operator and parameter influences are identified. Concluding remarks put the derived bound into a broader perspective of learning classifier system and machine learning research.

## 2 XCS Theory

The accuracy-based learning classifier system XCS was introduced elsewhere [4]. Due to the accuracy-based fitness approach, XCS learns not only the rules (or classifiers) that denote the best classification possible, but rather a complete situation-action-reward mapping. In short, XCS is designed to learn a complete, accurate, and maximally general *payoff map* of an environment (out of which an optimal behavioral/classification policy can be derived).

XCS knowledge is represented by a *population* of maximally $N$ condition-action-reward prediction classifiers. Each classifier essentially specifies the expected reward given the specified conditions and executing the specified action. Rule evaluation is done via a credit assignment mechanism [11] with similarities to algorithms in reinforcement learning [4,12]. Rule generation and evolution is done via a steady-state, niched genetic algorithm [13,14]. If a GA is applied in a particular learning iteration, two classifiers are reproduced using tournament selection [15] with respect to their fitness in the current action set (the subset of classifiers that contains all classifiers that specify the executed action and whose conditions are satisfied by the current problem instance). To keep the population size constant, two classifiers are deleted via proportionate selection from the whole population.

The following XCS theory focuses on Boolean function problems in which each problem instance is represented by $l$ binary features and belongs to one of $n$ classes. The *specificity* of a classifier refers to the number of features that the classifier condition specifies over the total number of features $l$ (usually unspecified features are represented by the don't care symbol #).

### 2.1 Evolutionary Pressures in XCS

After Kovacs analyzed the problem of *strong overgenerals* that shows that any strength-based learning classifier system (without fitness sharing techniques) suffers from the tragedy of the common [16], recent analysis investigated the evolutionary learning progress in XCS. In [9] five major evolutionary pressures in XCS, (1) set pressure, (2) mutation pressure, (3) deletion pressure, (4) subsumption pressure, and (5) fitness pressure, were identified.

Intuitively, the *set pressure* formalizes the intrinsic generalization pressure in XCS. Since the average specificity $s[A]$ of classifiers in an action set $[A]$ is lower than the average specificity $s[P]$ in the population and classifiers are reproduced in action sets but deleted in the population, the average offspring specificity is lower than the specificity of the replaced classifiers.

Also *mutation pressure* influences specificity. Generally, a random mutation process causes a tendency towards an equal number of symbols in a population. Thus, applying random mutations, the result will be a population with an approximately equal proportion of 0, 1, and # symbols in the condition parts of classifiers in a binary LCS. Thus, mutation results in a pressure towards an equal number and an equal distribution of symbols in classifier conditions.

Combining set and mutation pressure to a general *specificity equation* [10] a general estimate of the change in the population's specificity can be derived:

$$\Delta_{spe} s([P]) = f_{ga} \frac{2\,(s([A]) + \Delta_{mut} - s([P]))}{N}. \tag{1}$$

Parameter $f_{ga}$ approximates the average frequency of GA application, $s([X])$ refers to the average specificity of the referred set $X$, $\Delta_{mut}$ quantifies the specificity change due to mutation, and $N$ specifies the population size. The formula allows an accurate prediction of specificity change and convergence over time given no fitness influence [9,10]. It was also shown that given no fitness influence, the converged specificity in the population can be roughly approximated by twice the mutation rate.

The main part of the *deletion pressure* is already included in the set pressure. In addition to deleting classifiers from the population while reproducing classifiers in action sets, deletion is biased towards deleting classifiers that populate large niches and classifiers with a fitness value which is significantly smaller than the average value of the population [17].

*Subsumption pressure* is designed to decrease the population size boiling it down to the accurate, maximally general classifiers. It applies only if the noise in a problem is lower than the error threshold $\varepsilon_0$ below which a classifier is considered for subsumption. Subsumption must be applied with care. A too low experience threshold $\theta_{sub}$ as well as a too-high value of $\varepsilon_0$ can cause fundamental loss of information in the population (subsuming accurate classifiers by a temporarily accurate, over-general classifier).

*Fitness pressure* is needed to generate a major drive towards accuracy from the inaccurate (and thus mainly over-general) side. Essentially, fitness pressure causes the reproduction of higher accurate classifiers. Thus, fitness is the major pressure that guides the evolutionary process towards higher accuracy.

Over the last years it became clear that, given the problem provides appropriate fitness guidance, fitness pressure needs to be strong enough to overcome the set pressure. Since the traditionally applied proportionate selection mechanism is highly dependent on fitness scaling, a set-size proportionate *tournament selection* mechanism was introduced to XCS [15] that results in a more robust and more problem independent XCS learning system.

## 2.2 Problem Bounds

In addition to the above evolutionary pressure in XCS, several problem bounds have been identified. The following paragraphs give an overview over the derived problem bounds.

*Covering Challenge.* The first bound was formulated in [8] requiring a minimal population size to assure that the genetic algorithm actually applies and is not blocked by a continuous covering-deletion cycle. Given a current specificity of $s[P]$ in the population and assuming a problem in which any possible problem instance is equiprobable, the covering probability is determined by

$$P(cover) = 1 - \left[1 - \left(\frac{2 - s([P])}{2}\right)^l\right]^N, \tag{2}$$

where $l$ specifies the number of binary features in the problem. In order to keep the covering probability initially sufficiently high to ensure the start of the evolutionary process, the initial specificity needs to be set sufficiently low (controlled by the don't care probability $P_{\#}$).

*Schema Challenge.* In addition to the assurance of input covering, it also needs to be assured that classifiers represent particular schemata. To characterize such a classifier, Holland's schema notion is used [13]. A *representative* of a particular schema of order $o$ must have at least all $o$ positions correctly specified. For successful evolution, the presence of a representative of a schema of order $o$ needs to be highly probable. The probability of the existence of a representative can be determined by

$$P(representative) = 1 - \left[1 - \frac{1}{n}\left(\frac{s([P])}{2}\right)^o\right]^N, \tag{3}$$

assuming a binomial distribution of the specificity in the population. Parameter $n$ denotes the number of classes. While the previous specificity measure is mainly relevant for the beginning of the run, the current specificity of the population directly affects the probability of the availability of a representative.

*Reproductive Opportunity.* To ensure successful evolution, it is necessary to assure gradual evolution ensuring *reproductive opportunities* for the better classifiers. Existing or generated higher-accurate classifiers need to have reproductive opportunities before being deleted. To ensure this, the expected time until a reproductive opportunity should be shorter than the expected time until deletion. This constraint effectively results in a population size bound since only a larger population size can increase the time until deletion in XCS [5].

$$N > n 2^{k_d + (l - k_d) s([P]) + 1} \tag{4}$$

This bound ensures that classifiers necessary in a problem of order of minimal schema order $k_d$ get reproductive opportunities. Once the bound is satisfied, existing representatives of an order $k_d$ schema have a high probability of reproduction. Thus, with a high probability, XCS will evolve a more accurate population.

Note that this population size bound is actually exponential in minimal schema order $k_d$ and in string length times specificity $ls([P])$. However, it was shown that the necessary specificity in $[P]$ decreases with larger population sizes [5]. In particular, requiring that a representative of a particular schema order $k_d$ exists, it can be shown that the required minimal specificity is bounded by

$$O(\left(\frac{n}{N}\right)^{\frac{1}{k_d}}). \tag{5}$$

Considering this, a general *reproductive opportunity bound (ROP-bound)* can be derived that shows that population size grows as

$$O(l^{k_d}). \tag{6}$$

The bound essentially determines that the populations size grows polynomially in the problem length $l$ and exponentially in the problem difficulty. Thus, the computational complexity grows similar to any inductive machine learning algorithm such as for example the inductive decision tree learner C4.5 [18]. The specificity bound and population size bound will also be relevant for the following derivation of the learning time.

## 3 Bounding Learning Time in XCS

To derive our learning time bound, we estimate the time until reproduction of the current best classifier as well as the time until creation of the next best classifier via mutation given a reproductive event of the current best classifier. The model assumes a completely general initial population. First specializations are randomly introduced via mutation. Problem-specific initialization techniques or a higher initial specificity in the population may speed-up learning time (as long as the covering challenge is not violated). Further assumptions are that the current best classifier is not lost (assured by the ROP-bound) and that it is selected as the offspring when it is part of an action set (assured by the selection mechanism). The time model assumes domino convergence [19] in which each attribute is successively specified. This means that only once the first attribute is correctly specified in a classifier, then the second attribute influences fitness and so forth.

With the above assumptions, we can bound the learning time in the following way. First, we estimate the probability that mutation correctly specifies the next attribute

$$P(\text{perfect mutation}) = \mu(1-\mu)^{l-1} \tag{7}$$

where $l$ specifies the number of attributes in a problem instance (i.e. condition length). This probability can be relaxed in that we only require that the $k$ already correctly set features are not unset (changed to don't care), the next feature is set, and we do not care about the others:

$$P(\text{good mutation}) = \mu(1-\mu)^k \tag{8}$$

Equation 7 specifies the lower bound on the probability that the next best classifier is generated whereas Equation 8 specifies an optimistic bound.

The probability of reproduction of a classifier is mainly influenced by the probability of being part of an action set. The probability of being part of an action set again, is determined by the current specificity of a classifier. Given a classifier which specifies $k$ attributes, the probability of reproduction is

$$P(\text{reproduction}) = \frac{1}{n}\frac{1}{2}^{k} \tag{9}$$

where $n$ denotes the number of actions in a problem. The best classifier has a minimal specificity of $k/l$. With respect to the current specificity in the population $s([P])$, the specificity of the best classifier may be expected to be $k+s([P])(l-k)$ assuming a uniform specificity distribution in the other $l-k$ attributes. Taking this expected specificity into account, the probability of reproduction is

$$P(\text{reproduction in [P]}) = \frac{1}{n}\frac{1}{2}^{k+s([P])(l-k)}. \tag{10}$$

Since the probability of a successful mutation assumes a reproductive event, the probability of generating a better offspring than the current best is determined by

$$P(\text{generation of next best cl.}) = P(\text{reproduction in [P]})\ P(\text{good mutation}) = \frac{1}{n}\frac{1}{2}^{k+s([P])(l-k)} \mu(1-\mu)^{l-1}. \tag{11}$$

Since this is a geometric distribution (memoryless property, each trial has an independent and equally probable distribution), the expected time til the generation of the next best classifier is

$$E(\text{time until generation of next best cl.}) = 1/P(\text{generation of next best cl.}) = \frac{1}{\frac{1}{n}\frac{1}{2}^{k+s([P])(l-k)}\mu(1-\mu)^{l-1}} = \frac{n2^{k+s([P])(l-k)}}{\mu(1-\mu)^{l-1}} \leq \frac{n2^{k+s([P])l}}{\mu(1-\mu)^{l-1}}. \tag{12}$$

Given now a problem in which $o$ features need to be specified and given further the domino convergence property in the problem, the expected time until the generation of the next best classifier can be summed to derive the time until the generation of the global best classifier:

$$E(\text{time until generation of maximally accurate cl}) = \sum_{k=0}^{o-1}\frac{n2^{k+s([P])l}}{\mu(1-\mu)^{l-1}} = \frac{n2^{s([P])l}}{\mu(1-\mu)^{l-1}}\sum_{k=0}^{o-1}2^k < \frac{n2^{o+s([P])l}}{\mu(1-\mu)^{l-1}}. \tag{13}$$

This time bound shows that XCS needs an exponential number of evaluations in the problem difficulty $o$. As argued above, the specificity and consequently also mutation needs to be decreased indirect proportional to the string length $l$. In particular, since specificity $s([P])$ grows as $O((\frac{n}{N})^{\frac{1}{k_d}})$ (Equation 5) and population size grows as $O(l^{k_d})$ (Equation 6), specificity essentially grows as $O(\frac{n}{l})$. Using the O-notation and plugging this behavior into Equation 13 we derive the following adjusted time bound.

$$O\left(\frac{l2^{o+n}}{(1-\frac{n}{l})^{l-1}}\right) = O\left(\frac{l2^{o+n}}{e^{-n}}\right) = O\left(l2^{o+n}\right) \quad (14)$$

Thus, learning time in XCS is bound mainly by the order of problem difficulty $o$ and the number of problem classes $n$. It is linear in the problem length $l$. This derivation essentially also validates Wilson's hypothesis that XCS learning time grows polynomially in problem complexity as well as problem length [20]. The next section experimentally validates the derived learning bound.

# 4 Experimental Validation

In order to validate the derived bound, we evaluate XCS performance on an artificial problem in which domino convergence is forced to take place. Similar results are expected in typical Boolean function problems in which similar fitness guidance is available, such as in the layered multiplexer problem [4,5]. In other problems, additional learning influences may need to be considered such as the influence of crossover or the different fitness guidance in the problem [5].

To force domino convergence, instead of using the usual Widrow-Hoff delta rule to update classifier estimates, we set the reward prediction error directly to a fixed value according to the current specificity of the classifier. Given a problem of problem difficulty $o$, the prediction error of a classifier is set to $500(o-k)/o$ where $k$ denotes the number of successive relevant attributes specified in the classifier. Thus, given a problem of length $l=6$ and $o=3$ (and assuming a left-to-right order with the first three features being relevant), the classifier 1#1111 would be assigned an error of 333 whereas classifier 011#1# would be assigned an error of 0.

If not stated differently, the XCS classifier system is applied with a GA threshold $\theta_{GA}=0$ (the GA is always applied), error instead of fitness-based selection, tournament selection with a action-set proportionate tournament size of $\tau=0.4$, niche mutation, no action mutation, no crossover, an initial completely general population ($P_{\#}=1.0$) and GA subsumption ($\theta_{sub}=0, \epsilon_0=1$). The results are averaged over 20 experiments. The error-based selection approach eliminates the additional evolutionary influence due to fitness sharing.

## 4.1 Time Bound Validation

To validate the time bound, we monitor the specificity of the relevant attributes. According to the domino convergence theory, the system should successively detect the necessary specialization of each relevant attribute eventually converging

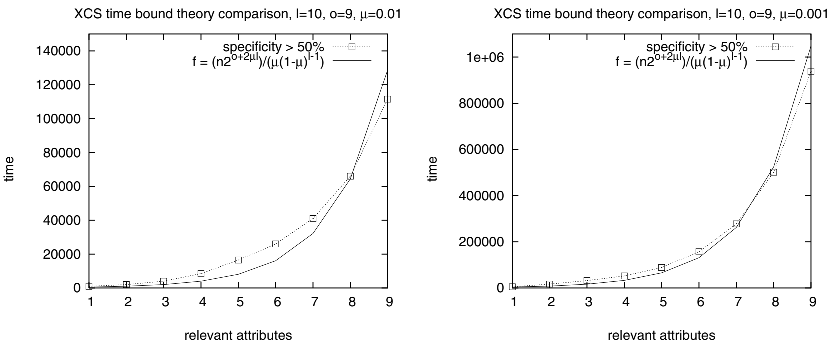


**Fig. 1.** The theory comparison shows that the time until the specificity of the successive attributes has reached 50% is approximated by the theoretical bound. Maximum population size $N$ is set to 32000.

to a specificity of nearly 100%. The time bound estimates the expected time until all relevant attributes are detected. To evaluate the bound, we record the number of steps until the specificity of a particular attribute reaches 50%. This criterion indicates that the necessary specificity is correctly detected but it does not require full convergence.

Figure 1 shows the time until 50% specificity is reached in the successive attributes in the setting with $l = 10$ and $o = 9$. The comparison with the theoretical bounds matches approximating the specificity in the population $s([P])$ with $2\mu$ which has been shown to be approximately correct [5]. As predicted by the theory, decreasing the mutation rate (Figure 1, right-hand side) increases the time until the required specificity is reached. Although nearly all interactions between the different niches are prevented by disallowing the mutation of the action part and by applying niche mutation only, the specificity in the later attributes still is learned slightly faster than predicted by the theory. Two-stage interactions might occur in which mutation first overgeneralizes a highly accurate classifier and then specializes it in another niche.

The second concern is the influence of the number of irrelevant attributes. Figure 2 shows that also in this case the theory closely matches the empirical results. Since a higher mutation rate results in a higher specificity, the influence of the number of irrelevant attributes is more significant in the setting with a mutation rate of $\mu = 0.01$. In the low mutation case, the bound is approximated in the settings with larger string length. Hereby, the specificity is approximated by twice the mutation rate. Using a smaller population size can delay or stall the evolutionary process due to the reproductive opportunity bound.

Figure 3 shows the behavior of the specificities of the six relevant positions and several of the other positions (that all behave similarly). It can be seen that complete convergence is delayed if population size is set not high enough. The reproductive opportunity bound slowly comes into play. With a string length of more than 150, evolution partially stalls completely since the overspecialized

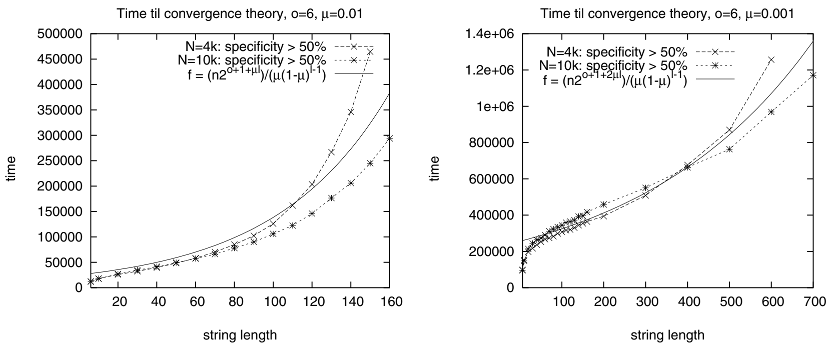


**Fig. 2.** The influence of problem length is properly bound by the theory. The reproductive opportunity bound increasingly outweighs the time bound when the population size is set too low.

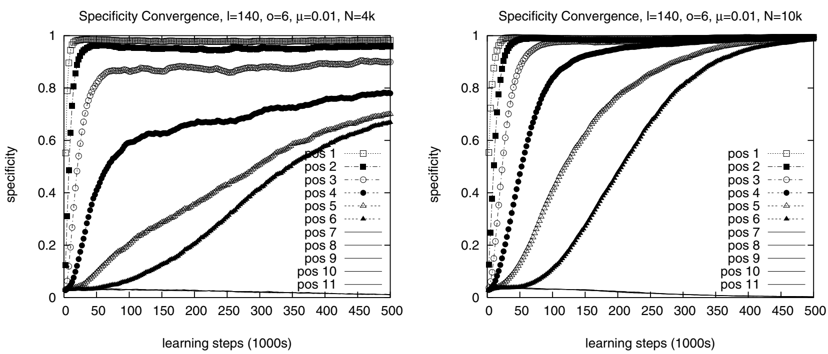


**Fig. 3.** Increasing the problem length further, the reproductive opportunity bound increasingly affects convergence stalling learning by preventing the reproduction of more-accurate classifiers. Increasing the population size remedies the reproductive opportunity bound giving more-accurate classifiers more time for reproduction.

irrelevant attributes prevent sufficient reproductive opportunities (see Section 2.2 and [5]). With a higher population size, the reproductive opportunity bound vanishes and all six specificities converge to one without delay. Similar behavior is found for the case with a lower mutation rate and larger string length as indicated in Figure 2 (right-hand side).

### 4.2 Parameter Influences

In addition to the above bound, we investigate the effects of several parameters and additional mechanisms in XCS. Figure 4 reveals dependences on several XCS mechanisms in the setting with a string length $l = 10$ and the number of relevant attributes $o = 4$ (left-hand side) and $o = 8$ (right-hand side). In the setting with four relevant attributes, we can see that the disallowance of action mutation as

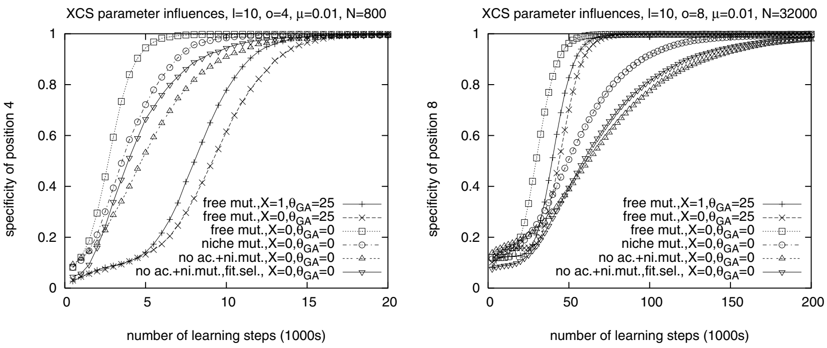


**Fig. 4.** Several mechanisms influence the learning speed: action mutation and free mutation facilitate the additional knowledge exchange between different problem niches and subsolutions. A higher GA threshold delays evolution especially given a more generalized population. Fitness-based selection slightly speeds-up learning. More relevant attributes and thus a more specialized population show similar performance influences (right-hand side).

well as the restriction to niche mutation decreases performance. Allowing action mutation or free mutation, subsolutions in one problem niche can propagate much easier to another niche (by mutating action 1 to 0 or specified attribute 1 to 0 or vice versa). Increasing the GA threshold $\theta_{GA}$ delays the evolutionary process. Uniform crossover has an additional beneficial effect enhancing the possibility of knowledge exchange between different niches. Fitness-based selection also slightly speeds-up learning. Due to the fitness decrease in offspring (fitness is set to 10% of the parental fitness), a slight generalization pressure [21] is generated that decreases specificity (in particular the specificity of the irrelevant attributes) and thus facilitates learning. In the setting with eight relevant attributes (right-hand side), the performance decrease due to restricted mutation overshadows the decrease due to a higher GA threshold.

## 5 Summary and Conclusions

This paper introduced a first learning time bound to the XCS classifier system. Assuming a domino convergence model in which each relevant attribute converges successively, we showed that learning time grows polynomially in problem complexity and linearly in the problem length. The provided experiments validated the basic assumptions in the derivations hold. The results confirm Wilson's original learning time estimation [20]. Additional learning mechanisms were shown to improve learning speed enabling a better knowledge exchange between different subsolutions or niches such as free mutation or crossover. Satisfying all problem bounds in XCS and given a problem structure that allows domino convergence, we can now assure that XCS learns a problem in time polynomial in problem complexity and problem length.

The research points out the strong dependence of successful learning on the underlying problem structure. Several problem difficulty measures were detected. The most trivial one is the problem length in which XCS scales polynomially. The second one is the order of the problem, which specifies the minimal number of attributes that need to be specified to be maximally accurate. We showed that XCS learning time scales exponentially in the problem order and thus polynomially in problem complexity. Finally, previously we showed that the minimal number of attributes that need to be specified to reach higher accuracy is a third problem bound (discussed in Section 2.2). Given a problem with a particular order of problem difficulty, we are now able to estimate the required population size and estimate the required learning time to solve the problem successfully.

The current time bound is applied in problems in which the domino convergence property holds. That is, each attribute progressively reduces the error estimate of a classifier and thus increases its accuracy and fitness. In problems in which this property does not hold, the convergence time may vary and other operators may be necessary to achieve successful learning. In particular, different substructures of accurate subsolutions may need to be recombined in a proper way applying intelligent recombination operators. Also, fitness guidance may be violated and a bilateral fitness approach may be necessary, as investigated in [5]. Nonetheless, this learning time bound shows that XCS is able to learn in a machine-learning competitive way. Future research will show the ways the bound can be modified and enhanced to account for recombinatory events as well as for other problem types.

**Acknowledgment.** We are grateful to Xavier Llorà, Kei Onishi, Kumara Sastry, and the whole IlliGAL lab for their help and the useful discussions. The work was sponsored by the Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grant F49620-03-1-0129. The US Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon. Additional funding from the German research foundation (DFG) under grant DFG HO1301/4-3 is acknowledged. Additional support from the Computational Science and Engineering graduate option program (CSE) at the University of Illinois at Urbana-Champaign is acknowledged. 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 or the U.S. Government.

## References

1. Holland, J.H.: Processing and processors for schemata. In Jacks, E.L., ed.: Associative Information Techniques, New York, American Elsevier (1971) 127–146
2. Wilson, S.W.: ZCS: A zeroth level classifier system. Evolutionary Computation **2** (1994) 1–18
3. Bull, L., Hurst, J.: ZCS redux. Evolutionary Computation **10** (2002) 185–205

4. Wilson, S.W.: Classifier fitness based on accuracy. Evolutionary Computation **3** (1995) 149–175
5. Butz, M.V., Goldberg, D.E., Tharakunnel, K.: Analysis and improvement of fitness exploitation in XCS: Bounding models, tournament selection, and bilateral accuracy. Evolutionary Computation **11** (2003) 239–277
6. Bernadó, E., Llorà, X., Garrell, J.M.: XCS and GALE: A comparative study of two learning classifier systems and six other learning algorithms on classification tasks. In Lanzi, P.L., Stolzmann, W., Wilson, S.W., eds.: Advances in learning classifier systems: Fourth international workshop, IWLCS 2001 (LNAI 2321). Springer-Verlag, Berlin Heidelberg (2002) 115–132
7. Dixon, P.W., Corne, D.W., Oates, M.J.: A preliminary investigation of modified XCS as a generic data mining tool. In Lanzi, P.L., Stolzmann, W., Wilson, S.W., eds.: Advances in learning classifier systems: Fourth international workshop, IWLCS 2001 (LNAI 2321). Springer-Verlag, Berlin Heidelberg (2002) 133–150
8. Butz, M.V., Kovacs, T., Lanzi, P.L., Wilson, S.W.: How XCS evolves accurate classifiers. Proceedings of the Third Genetic and Evolutionary Computation Conference (GECCO-2001) (2001) 927–934
9. Butz, M.V., Pelikan, M.: Analyzing the evolutionary pressures in XCS. Proceedings of the Third Genetic and Evolutionary Computation Conference (GECCO-2001) (2001) 935–942
10. Butz, M.V., Kovacs, T., Lanzi, P.L., Wilson, S.W.: Toward a theory of generalization and learning in XCS. IEEE Transactions on Evolutionary Computation **8** (2004) 28– 46
11. Holland, J.H., Reitman, J.S.: Cognitive systems based on adaptive algorithms. In Waterman, D.A., Hayes-Roth, F., eds.: Pattern directed inference systems. Academic Press, New York (1978) 313–329
12. Lanzi, P.L.: Learning classifier systems from a reinforcement learning perspective. Soft Computing: A Fusion of Foundations, Methodologies and Applications **6** (2002) 162–170
13. Holland, J.H.: Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor, MI (1975) second edition 1992.
14. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, Reading, MA (1989)
15. Butz, M.V., Sastry, K., Goldberg, D.E.: Tournament selection in XCS. Proceedings of the Fifth Genetic and Evolutionary Computation Conference (GECCO-2003) (2003) 1857–1869
16. Kovacs, T.: Towards a theory of strong overgeneral classifiers. Foundations of Genetic Algorithms 6 (2001)
17. Kovacs, T.: Deletion schemes for classifier systems. Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-99) (1999) 329–336
18. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann, San Francisco, CA (1993)
19. Thierens, D., Goldberg, D.E., Pereira, A.G.: Domino convergence, drift, and the temporal-salience structure of problems. In: Proceedings of the 1998 IEEE World Congress on Computational Intelligence, New York, NY, IEEE Press (1998) 535–540
20. Wilson, S.W.: Generalization in the XCS classifier system. Genetic Programming 1998: Proceedings of the Third Annual Conference (1998) 665–674
21. Bull, L.: Investigating fitness sharing in a simple payoff-based learning classifier system. Technical Report UWELCSG03-009, University of Western England, Learning Classifier System Group, Bristol, UK (2003)