5

How XCS Works: Ensuring Effective Evolutionary Pressures

The last chapter gave a concise introduction to the accuracy-based XCS classifier system. We saw that XCS is designed to evolve online a complete, maximally accurate, and maximally general solution to the problem at hand (e.g. by approximating the Q-value function). The accuracy-based approach assures that no strong overgenerals are possible since the maximally accurate classifiers receive maximal fitness.

With this knowledge in mind, we now turn to the further aspects of our facetwise LCS theory approach regarding the evolutionary pressures in the XCS classifier system. While strong overgenerals are prevented by the accuracy-based fitness approach, it still needs to be assured that fitness *guides* towards the intended solution. Second, parameter initialization and estimation needs to be most effective. Third, appropriate generalization needs to apply so that the solution becomes maximally general.

To investigate these points in XCS, we undertake a general analysis of all *evolutionary pressures* in XCS. Evolutionary pressures can be regarded as evolutionary biases that influence or bias learning in XCS. Often, the pressures influence *specificity* in the classifier population. Specificity was defined in Chapter 3. It essentially characterizes how restricted a classifier condition is. The smaller the problem subspace a classifier condition covers, the more specific it is.

The pressure analysis quantifies generalization in XCS as well as the influence of mutation. It leads to an equilibrium in population specificity if no fitness pressure applies. This interaction is quantified in the *specificity equation*, which we evaluate in detail. With the addition of fitness pressure and subsumption, the equilibrium lies exactly at the point of the desired maximally accurate and maximally general problem solution.

However, in order to ensure that fitness guides to the equilibrium, the generalization pressure needs to be overcome. This is assured by an appropriate selection mechanism leading us to the introduction of *tournament selection* for offspring selection. We show that tournament selection assures sufficiently strong fitness guidance and consequently makes XCS much more parameter as well as noise independent.

The next section introduces all evolutionary pressures, analyzes them in separation, and combines them in one general specificity equation. The results are experimentally validated in binary classification problems. Next, tournament selection is introduced and evaluated. The investigations prepare XCS to face the subsequent challenges outlined in the facetwise theory approach.

5.1 Evolutionary Pressures in XCS

Previous publications have considered the influence of fitness guidance and generalization. The principles underlying evolution in XCS were originally outlined in Wilson's *generalization hypothesis* (Wilson, 1995), which suggests that classifiers in XCS become maximally general due to niche-based reproduction in combination with population-wide deletion. Subsequently, Kovacs (1996) extended Wilson's explanation to an *optimality hypothesis*, supported experimentally in small multiplexer problems, in which he argued that XCS develops minimal representations of optimal solutions (that is, the optimal solution representation $[O]$). Later, Wilson (1998) suggested that XCS scales polynomially in problem complexity and thus is machine learning competitive. Kovacs and Kerber (2001) related problem complexity directly to the size of the optimal solution $|[O]|$. However, Butz, Kovacs, Lanzi, and Wilson (2001) showed that the $|[O]|$ measure is clearly not the only measure that influences XCS's learning behavior.

Despite these insights, Wilson's generalization hypothesis still needed to be theoretically validated and quantified. In this section, we investigate Wilson's hypothesis developing a fundamental theory of XCS generalization and learning. To avoid the additional complication of back-propagating reward in RL problems, we focus on XCS's performance in classification problems.

In particular, we analyze the evolutionary pressures present in XCS. An evolutionary pressure refers to a learning bias in XCS influencing the population structure, often with respect to specificity. The average *specificity* in a population, denoted by $\sigma[P]$, refers to the average specificity of all classifiers in the population. That is,

$$\sigma[P] = \frac{\sum_{c \in [P]} \sigma(c) \cdot c.num}{\sum_{c \in [P]} c.num}, \tag{5.1}$$

where $\sigma(c)$ refers to the specificity of classifier c. We defined specificity for the binary case in Chapter 3 as the fraction of specified attributes (zero or one) in the condition part of a classifier.

Our analysis distinguishes between the following evolutionary pressures:

1. *Set pressure*, which quantifies Wilson's generalization hypothesis;

2. *Mutation pressure*, which quantifies the influence of mutation on specificity;
3. *Deletion pressure*, which qualifies additional deletion influences;
4. *Fitness pressure*, which qualifies the accuracy-based fitness influence;
5. *Subsumption pressure*, which qualifies the exact influence of subsumption deletion.

Set pressure and mutation pressure are combined to a general *specificity equation* that is evaluated in several experiments progressively increasing the influence of deletion and fitness pressure.

5.1.1 Semantic Generalization due to Set Pressure

The basic idea behind the set pressure is that XCS reproduces classifiers in action sets $[A]$ whereas it deletes classifiers from the whole population $[P]$. The set pressure is a combination of the selection pressure produced by the GA applied in $[A]$ and the pressure produced by deletion applied in $[P]$. It was originally qualitatively proposed in Wilson's original paper (Wilson, 1995) and later further experimentally analyzed by Kovacs (1996).

The generalization hypothesis argues that since more general classifiers appear more often in action sets $[A]$, they undergo more reproductive events. Combined with deletion from $[P]$, the result is an intrinsic tendency towards generality favoring more general classifiers. Classifiers in this respect are *semantically* more general in that they are more frequently part of an action set. Classifiers that are equally often part of an action set but may be distinguished by syntactic generality are not affected by the set pressure.

To formalize the set pressure, we determine the expected specificity $\sigma[A]$ of classifiers in an action set $[A]$ with respect to the current expected specificity $\sigma[P]$ of the classifiers in population $[P]$. The specificity of the initial random population $\sigma[P]$ is directly correlated with the don't-care probability $P_{\#}$, i.e., $\sigma[P] = 1 - P_{\#}$. For our calculations, we assume a binomial specificity distribution in the population. This assumption essentially holds in the case of a randomly generated population. It enables us to determine the probability that a randomly chosen classifier in the population $cl \in [P]$ has specificity k/l as follows:

$$P(\sigma(cl) = k/l|\sigma[P]) = \binom{l}{k} \sigma[P]^k \left(1 - \sigma[P]\right)^{l-k}, \tag{5.2}$$

where cl is a classifier; l is the length of classifier conditions; and k is the number of *specified attributes* in the condition, that is, the number of attributes different from a don't-care symbol. The equation essentially is able to estimate the proportion of different specificities in a population with average specificity $\sigma[P]$.

The probability that a classifier cl matches a certain input s depends on its specificity $\sigma(cl)$. To match, a classifier cl with specificity k/l must match all k specific bits. This event has probability 0.5^k since each specific attribute matches with probability 0.5. Therefore, the proportion of classifiers in [P] with a specificity k/l that match in a specific situation is as follows:

$$\begin{aligned} &P(cl\ matches \wedge \sigma(cl) = k/l|\sigma[P]) = \\ &= P(\sigma(cl) = k/l|\sigma[P])P(cl\ matches|\sigma(cl) = k/l) = \\ &= P(\sigma(cl) = k/l|\sigma[P])\, 0.5^k = \binom{l}{k}\left(\frac{\sigma[P]}{2}\right)^k (1-\sigma[P])^{l-k} \end{aligned} \tag{5.3}$$

To derive a specificity $\sigma[M]$ of a match set $[M]$, it is first necessary to specify the proportion of classifiers in $[M]$ with specificity k/l given the population specificity $\sigma[P]$. This proportion can be derived as follows:

$$\begin{aligned} P(\sigma(cl) = k/l|cl \in [M] \wedge \sigma[P]) &= \frac{P(cl\ matches \wedge \sigma(cl) = k/l)|\sigma[P]}{\sum_{i=0}^{l} P(cl \text{ matches} \wedge \sigma(cl) = i/l)|\sigma[P]} = \\ = \frac{\binom{l}{k}\left(\frac{\sigma[P]}{2}\right)^k (1-\sigma[P])^{l-k}}{\sum_{i=0}^{l}\binom{l}{i}\left(\frac{\sigma[P]}{2}\right)^i (1-\sigma[P])^{l-i}} &= \frac{\binom{l}{k}\left(\frac{\sigma[P]}{2}\right)^k (1-\sigma[P])^{l-k}}{\left(1-\frac{\sigma[P]}{2}\right)^l} = \\ &= \binom{l}{k}\left(\frac{\sigma[P]}{2-\sigma[P]}\right)^k \left(1-\frac{\sigma[P]}{2-\sigma[P]}\right)^{l-k} \end{aligned} \tag{5.4}$$

To compute $\sigma[M]$ we multiply actual specificity values, k/l, by the proportions $P(\sigma[M] = k/l|\sigma[P])$ and sum up the values to derive the resulting specificity of $[M]$. Since the action set $[A]$ has on average the same specificity as the match set $[M]$ ($\sigma[A] \approx \sigma[M]$), $\sigma[A]$ can be derived as follows:

$$\begin{aligned} &\sigma[A] \approx \sigma[M] = \\ &= \sum_{k=0}^{l} \frac{k}{l} P(\sigma(cl) = k/l|cl \in [M] \wedge \sigma[P]) = \\ &= \sum_{k=1}^{l} \frac{k}{l}\binom{l}{k}\left(\frac{\sigma[P]}{2-\sigma[P]}\right)^k \left(1-\frac{\sigma[P]}{2-\sigma[P]}\right)^{l-k} = \\ &= \sum_{k=1}^{l}\binom{l-1}{k-1}\left(\frac{\sigma[P]}{2-\sigma[P]}\right)^k \left(1-\frac{\sigma[P]}{2-\sigma[P]}\right)^{l-k} = \\ &= \frac{\sigma[P]}{2-\sigma[P]}\sum_{j=0}^{l-1}\binom{l-1}{j}\left(\frac{\sigma[P]}{2-\sigma[P]}\right)^j \left(1-\frac{\sigma[P]}{2-\sigma[P]}\right)^{l-1-j} = \\ &= \frac{\sigma[P]}{2-\sigma[P]} \end{aligned} \tag{5.5}$$

The equation can be used to determine the average expected specificity $\sigma[A]$ in an action set $[A]$ assuming a binomially distributed specificity with mean $\sigma[P]$ in the population.

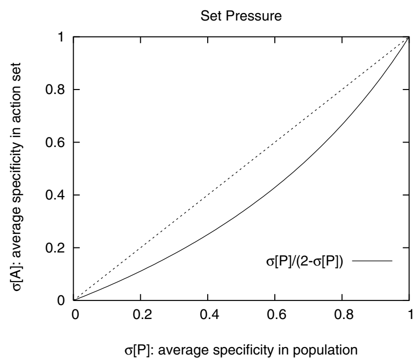


Fig. 5.1. Except at the lower and upper bounds, the expected average specificity of action sets $\sigma[A]$ is always smaller than that of the current population.

Figure 5.1 depicts Equation 5.5. Except at the lower and upper bounds, the specificity of $[A]$ is always smaller than the specificity of $[P]$. Thus, since selection takes place in the action sets but deletion occurs in the population as a whole, there is a tendency for the generality of the population to increase—in line with Wilson's generalization hypothesis. In the absence of fitness pressure, the equation provides an estimate of the difference in specificity of selected and deleted classifiers. Equation 5.5 is enhanced below accounting for mutation as well.

5.1.2 Mutation's Influence

Although usually only a low mutation probability is applied, mutation still influences specificity. In the absence of other evolutionary influences, mutation pushes the population towards a certain proportion of zeros, ones, and don't-cares. As outlined in Chapter 3, free mutation pushes towards a distribution of 1:2 general:specific. *Niche mutation*, which mutates a specified attribute always to a don't care and a don't care always to the current value of the respective attribute, pushes towards a distribution of 1:1 general:specific.

The average expected change in specificity between the parental classifier c_p and the mutated offspring classifier c_o for the niche mutation case can be written as follows:

$$\begin{aligned}\Delta_{mn}(\sigma(c_p)) &= \sigma(c_o) - \sigma(c_p) = \\ &= \sigma(c_p)(1-\mu) + (1-\sigma(c_p))\mu - \sigma(c_p) = \\ &= \mu(1-2\sigma(c_p)), \end{aligned} \tag{5.6}$$

and for free mutation as

$$\begin{aligned}\Delta_{mf}(\sigma(c_p)) = \sigma(c_o) - \sigma(c_p) = \\ = \sigma(c_p)(1 - \frac{\mu}{2}) + (1 - \sigma(c_p))\mu - \sigma(c_p) = \\ = 0.5\mu(2 - 3\sigma(c_p)). \qquad (5.7)\end{aligned}$$

As expected, the increase in specificity is higher when free mutation is applied given low parental specificity ($\sigma(c_p) < 1/2$) and specificity decrease is lower when parental specificity is high ($\sigma(c_p) > 2/3$). Applying random selection, mutation, and random deletion, mutation pushes the population towards a specificity of 0.5 applying niche mutation and 0.66 applying free mutation. The current intensity of the pressure depends on the mutation type, the current parental specificity, and on the frequency of the GA application (influenced by the parameter θ_{GA}).

5.1.3 Deletion Pressure

The probability of a classifier being deleted depends on its action set size estimate as and (depending on classifier experience) its fitness F. Due to the resulting bias towards deleting classifiers that occupy larger action sets, deletion pushes the population towards an equal distribution of classifiers in each environmental niche. With respect to specificity, classifier selection for deletion from $[P]$ is essentially random and there is no particular deletion pressure for or against general classifiers. In the absence of other biases the average expected specificity of deleted classifiers is equal to the average specificity in the population $\sigma[P]$.

A more significant effect can be observed with respect to overlapping niches. Given there are two non-overlapping, accurate niches and another accurate niche that overlaps with either one of the former, the action set size estimate of the overlapping niche will be larger than that of the non-overlapping ones. For example, given the non-overlapping niches `000***` and `01*0**` and the overlapping niche `0*00**` (this is actually the case in the multiplexer problem), and given further that all niches are represented by accurate, maximally general classifiers with a numerosity of say ten, then the action set size estimate of the overlapping classifier will stay on 20 whereas the estimate of the non-overlapping ones will approximate 15, making the deletion of the overlapping classifier more likely. Thus, apart from emphasizing equal niche support, the action-set size estimate based deletion pushes the population towards a non-overlapping solution representation.

5.1.4 Subsumption Pressure

Subsumption deletion applies only to classifiers that are accurate ($\varepsilon < \varepsilon_0$) and sufficiently experienced ($exp > \theta_{sub}$). The accuracy requirement suggests that the problem is less noisy than ε_0 since otherwise classifiers are not expected to satisfy the criterion ever.

If accurate classifiers evolve, subsumption deletion pushes towards maximal *syntactic* generality in contrast to the set pressure which pushes towards *semantic* generality. *GA subsumption deletion* prevents the insertion of offspring into $[P]$, if there is a classifier in $[A]$ that is more general than the generated offspring. Thus, once an accurate, maximally general solution was found for a particular niche, no accurate, more specialized classifier will be inserted anymore, disabling any specialization in the current niche.

To summarize, subsumption pressure is an additional pressure towards accurate, maximally syntactically general classifiers from the over-specific side. It applies only when accurate classifiers are found. Thus, subsumption pressure is helpful mainly later in the learning process once accurate classifiers are found. It usually results in a strong decrease of population size, focusing on maximally general classifiers.

5.1.5 Fitness Pressure

Until now we have not considered the effect of fitness pressure which can influence several other pressures. Fitness pressure is highly dependent on the particular problem being studied and is therefore difficult to formalize. In general, fitness results in a pressure which pushes $[P]$ from the overgeneral side towards accurate classifiers. Late in the run, when the optimal solution is mainly found, it prevents overgeneralization.

As in the case of subsumption, fitness also pushes the population towards a non-overlapping problem representation. Since fitness is derived from the relative accuracy, the accuracy-share is lower in classifiers that overlap with many other accurate classifiers. In effect, unnecessary, overlapping classifiers have a lower fitness on average, are thus less likely to reproduce, and are thus likely to be deleted from the population.

In terms of specificity, fitness pressure towards higher accuracy results usually in a specialization pressure since higher specificity usually implies higher accuracy. Certain problems, however, may mislead fitness guidance in that more general classifiers may actually have higher accuracy. This is particularly the case in problems with unbalanced class distributions and multiple classes (Butz, Goldberg, & Tharakunnel, 2003; Bernadó-Mansilla & Garrell-Guiu, 2003). Since the problem is not as severe as originally suspected, we do not investigate it any further. As suggested by our facetwise theory, though, fitness guidance needs to be ensured.

In sum, fitness pressure usually works somewhat in the opposite direction (towards higher specificity) of the set pressure. Thus, given fitness pressure in a problem, the specificity in the population is expected to decrease less or to increase dependent on the amount of fitness pressure. Fitness pressure is certainly highly dependent on the investigated problem and thus hard to quantify. The following section combines the pressure influences into the specificity equation.

5.1.6 Pressure Interaction

We now combine the above evolutionary pressures and analyze their interaction. Initially, we consider the interaction of set pressure, mutation pressure, and deletion pressure, which yields an important relationship we call the *specificity equation.* Next, we consider the effect of subsumption pressure and potential fitness influences. Finally, we provide a visualization of the interaction of all the pressures. The analyses are experimentally evaluated in Section 5.1.7.

Specificity Equation

Since mutation is only dependent on the specificity of the selected parental classifier and deletion can be assumed to be random, selection and mutation can be combined into one *specificity equation.* Essentially, set pressure generalizes whereas mutation specializes *or* generalizes dependent on the specificity of the currently selected classifier.

Since fitness pressure is highly problem dependent, we disregard fitness influences, essentially assuming equal fitness of all classifiers in our analysis. As shown later in Section 5.1.7, this assumption *holds* when all classifiers are accurate and nearly holds when all are similarly inaccurate. Despite the fitness equality assumption, deletion is also dependent on the action set size estimate *as* of a classifier. However, in accordance with Kovacs' insight on the relatively small influence of this dependence (Kovacs, 1999), we assume a random deletion from the population in our formulation. Thus, as stated above, a deletion results on average in the deletion of a classifier with a specificity equal to the specificity of the population $\sigma[P]$. The generation of an offspring, on the other hand, results in the insertion of a classifier with an average specificity of $\sigma[A] + \Delta_{mx}(\sigma[A])$ $(x \in f, n)$ dependent on the type of mutation used. Putting the observations together we can calculate the average specificity of the resulting population after one time step:

$$\sigma[P(t+1)] = \sigma[P(t)] + f_{GA}\frac{2(\sigma[A] + \Delta_{mx}(\sigma[A]) - \sigma[P(t)])}{N} \tag{5.8}$$

The parameter f_{GA} denotes the frequency of a GA application per time step assuming a constant application frequency for now. The formula adds to the current specificity in the population $\sigma[P(t)]$ the expected change in specificity calculated as the difference between the specificity of the two reproduced and mutated classifiers, that is, $\sigma[A] + \Delta_{mx}(\sigma[A])$ and $\sigma[P(t)]$. Note that although the frequency f_{GA} is written as a constant in the equation, f_{GA} actually depends on θ_{GA}, the specificity $\sigma[P(t)]$, as well as on the specificity distribution in the population. Thus, f_{GA} cannot be written as a constant in general. However, by setting θ_{GA} to zero, it is possible to force f_{GA} to be one since the average time since the last application of the GA in an action set will always be at least one.

XCS's tendency towards accurate, maximally general classifiers from the overgeneral side is not dependent on the use of subsumption. However, subsumption is helpful in focusing the population more on the maximally general representation. In fact, although the set pressure pushes the population towards more general classifiers the pressure is somewhat limited. Equation 5.8 shows that without subsumption the complete convergence of the population towards maximally accurate, maximally general classifiers is not assured. However, XCS is an online learning system that should be flexible with respect to problem dynamics so that complete convergence is usually not desired.

Another reason for a potential lack of complete convergence can be that the set pressure is not present at all. This can happen if the state space of a problem is a proper subspace of all possible representable states $\{0,1\}^l$ (as is essentially the case in most datamining applications as well as in RL problems). Subsumption can be helpful in generalizing the population further.

As mentioned above, fitness pushes towards higher specificity from the overgeneral side. Equation 5.8 assumes that the selected parental classifier has an expected average specificity of $\sigma[A]$, effectively assuming random selection in the action set. However, selection is biased towards the selection of more accurate classifiers. In effect, $\sigma[A]$ needs to be replaced by the expected offspring specificity that depends on the expected fitness distribution in the action set. Since this distribution is not only dependent on the problem but also on the selection method used and the current specificity distribution in each action set, we won't analyze the fitness influence any further. However, it should be kept in mind that fitness influences are expected to cause a specialization pressure that diminishes the generalization effect of the set pressure.

All Pressures

The interaction of all the pressures is illustrated in Figure 5.2. In particular, the fitness pressure pushes $[P]$ towards more accurate classifiers; the set pressure pushes $[P]$ towards more general classifiers; the subsumption pressure pushes $[P]$ towards classifiers that are accurate and syntactically maximally general; the mutation pressure pushes towards a fixed proportion of symbols in classifier conditions. Deletion pressure is implicitly included in the notion of set pressure. More detailed effects of deletion are not depicted. Overall, these pressures lead the population towards a population of accurate maximally general classifiers. While set pressure and mutation pressure (free mutation is represented) are independent of classifier accuracy, subsumption pressure and, of course, fitness pressure are influenced by accuracy.

5.1.7 Validation of the Specificity Equation

This section evaluates the specificity equation, formulated in Equation 5.8, and the additional evolutionary pressures identified in the previous sections.

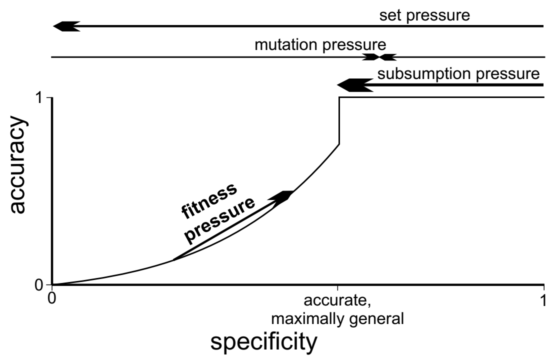


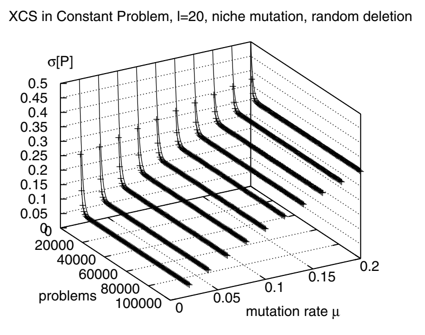
Fig. 5.2. The sketched interaction of all evolutionary pressures on imaginary specificity and accuracy axes shows how the evolutionary process in XCS is designed to evolve accurate, maximally general classifiers. If the classifiers are over-general (that is, specificity is smaller than at the accurate, maximally general point), accuracy is low (strongly dependent on the problem and XCS's accuracy equation) and fitness pushes towards the propagation of more accurate classifiers. On the other hand, if classifiers are over-specialized, accuracy is maximal but further generalization is possible. Set pressure and subsumption pressure are responsible to stress such generalizations.

The specificity equation summarizes the effect of three main evolutionary pressures in XCS: set pressure , mutation pressure, and deletion pressure.

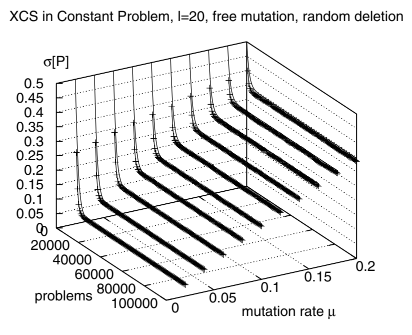
XCS is applied to Boolean strings of length $l = 20$ with different settings. The following figures show runs with mutation rates varying from 0.02 to 0.20. In each plot, solid lines denote the result from Equation 5.8; while crossed lines represent the result of actual XCS runs. Curves are averages over 50 runs. If not stated differently, the population is initially filled up with random classifiers with don't-care probability $P_{\#} = 0.5$. Niche mutation is applied. The other XCS parameters are set as follows: $N = 2000$; $\beta = 0.2$; $\alpha = 0.1$; $\varepsilon_0 = 10$; $\nu = 5$; $\theta_{GA} = 0$; $\chi = 0.8$, $\theta_{del} = 20$; $\delta = 0.1$; and $\theta_{sub} = \infty$. Note that the discount factor γ is irrelevant here since these are classification or single-step problems. Since this section is concerned with the set pressure, subsumption is turned off to prevent additional generalization influences.

Constant Function

We begin the validation of Equation 5.8 examining runs in a constant function, which always returns one. With these settings, all classifiers turn out to be accurate since their prediction error is always zero. Note however that a zero prediction error does not necessarily mean constant fitness values. In fact, since fitness is determined by the classifier's *relative accuracy*, fitness should still have an influence on evolutionary pressure. We investigate the influence of free mutation, niche mutation, the GA threshold θ_{GA}, and the influence of the deletion pressure.



(a)



(b)

Fig. 5.3. Solid lines represent the specificity as predicted in Equation 5.8. Marked lines represent the actual experimental specificity $\sigma[P]$. In a constant function setting, the actual specificity behaves nearly exactly as predicted by the model applying either (a) niche mutation or (b) free mutation.

Niche Mutation.

Figure 5.3a) depicts the average specificity $\sigma[P]$ in the specified settings varying the mutation rate. The deletion pressure is eliminated by deleting classifiers uniformly randomly in the population based solely on the numerosity *num*.

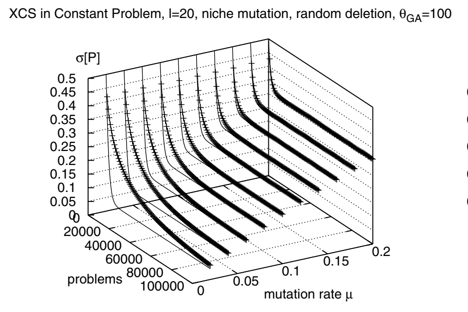
The empirical specificities match very closely to the model expressed in Equation 5.8. The initial specificity of 0.5 drops off quickly in the beginning due to the strong set pressure. Soon the specializing effect of the mutation pressure becomes visible and the specificity in the population converges as predicted. The higher the mutation rate μ, the stronger the specializing influence of mutation, as manifested in the higher convergence value in the curves with higher μ.

Free Mutation.

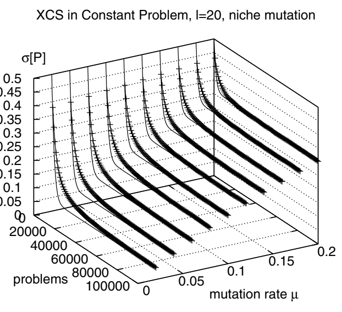
Figure 5.3b) depicts the specificity of the population [P] when free mutation is used. Besides the visibility of the mutation pressure due to the variation of μ, Figure 5.3b) confirms that free mutation has a slightly stronger influence on specificity as formulated in Equation 5.7. When directly comparing Figures 5.3a) and 5.3b) we note that the higher the parameter μ, the higher the influence of mutation pressure and thus the higher the differences in specificity due to the different mutation types.

θ_{GA} Threshold.

As noted above, the GA frequency f_{GA} in Equation 5.8 can generally not be written as a constant value. The frequency depends on the specificity $\sigma[P(t)]$



(a)



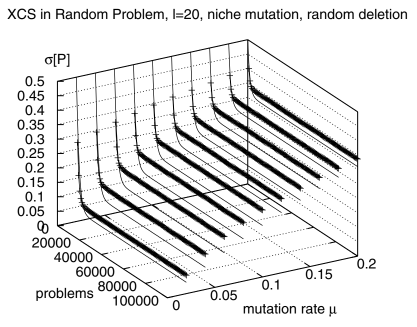
(b)

Fig. 5.4. (a) When applying a GA threshold of $\theta_{GA} = 100$, the GA frequency and consequently the specificity pressure decreases. Convergence values are not influenced. (b) Due to slow adaptation of the action set size estimate parameter, specificity convergence takes longer when action set size estimate based deletion is applied.

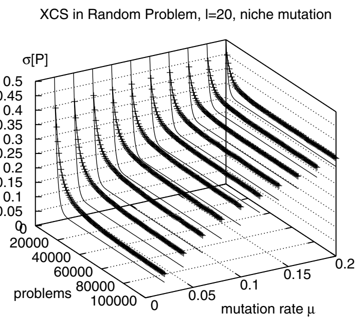
of the population, its distribution, and the problem distribution. However, the GA frequency f_{GA} equals one when the GA threshold θ_{GA} is set to zero (as has been done in the above evaluations). When setting θ_{GA} to a higher value (e.g. $\theta_{GA} = 100$), Figure 5.4a) reveals the lower GA frequency effect. Once the specificity in the population has dropped, the action set sizes increase since more classifiers match a specific state. Consequently, more classifiers take part in a GA application, more time stamps ts are updated, the average time since the last GA application in the population and in the action sets decrease, and finally the GA frequency decreases. The decrease is observable in the slower specificity decrease. As predicted by Equation 5.8, f_{GA} does not influence the convergence value.

Normal Deletion

The behavior of $\sigma[P]$ changes when we apply the usual deletion mechanism, which deletes proportional to the action set size estimate parameter *as*. Figure 5.4b) reports runs in which the usual deletion is used. Note that also in Figure 5.4b) the slopes of the curves decrease in comparison to the ones in Figure 5.3a). In the end, though, specificity of $[P]$ converges to the value predicted by the theory. The difference can only be the result of the bias in the deletion method of deleting classifiers with larger action set size estimates *as*. As the specificity of $[P]$ decreases, the action set size increases as noted before. Thus, since more general classifiers are more often present in action sets, their action set size estimate *as* is more sensitive to the change in the action set size and consequently, it is larger in more general classifiers while specificity drops. Eventually, all *as* values will have adjusted to the change



(a)



(b)

Fig. 5.5. (a) Applied to a random function, which returns zero or one equally probable for each problem instance, the specificity stays on a higher level due to the higher error variance in more specialized classifiers and parameter initialization effects. (b) Fitness-biased deletion decreases convergence of specificity due to the discussed action set size estimate influence.

and the predicted convergence value is met. This explanation is further confirmed by the fact that the difference between the actual runs and the curves given by Equation 5.8 become smaller and converge faster for higher mutation rates μ since the specificity slope is not as steep as in the curves with lower μ values.

Random Function

The results in the previous section show that the specificity equations accurately predict the behavior of XCS's average specificity. The fitness influence in XCS with a constant function proved to be rather small. Accordingly, we now apply XCS with two different deletion strategies to a much more challenging problem, that is, to a random Boolean function, which randomly returns rewards of thousand and zero. Figure 5.5a) reports the runs in which XCS with random deletion is applied to the random function. The experiments show that in the case of a random function, the fitness influences the specificity slope as well as the convergence value. In fact, the convergence value is larger than predicted by the model in Equation 5.8. The main effect is caused by the chosen initialization technique as well as by the higher error, and thus fitness variance in less experienced classifiers.

Since the possible rewards are 0 and 1000, assuming accurate parameter estimates in a classifier, classifier predictions fluctuate around 500, and consequently also the prediction errors fluctuate around 500. As in the more sensitive action set size estimates in Section 5.1.7, here the sensitivity is manifested in the prediction error ε. More specific classifiers have a less sensitive

ε and consequently a higher variance in the ε values. Since the accuracy calculation expressed in Equation 4.5 scales the prediction error to the power ν, the higher variance causes an *on average* higher accuracy and thus higher fitness.

Different parameter initialization techniques in combination with the moyenne adaptative modifiée technique enhance this influence. The more a classifier is inexperienced, the more the classifier parameters are dependent on the most recent cases. This, in combination with the scaled fitness approach, can make the effect even stronger. Since we set the experience *exp* of a new classifier to 1, XCS keeps the decreased parental parameter estimates so that fitness over-estimation is prevented. In fact, experimental runs with $exp = 0$ show that the specificity can increase to a level of even 0.2 independent of the mutation rate and type.

When applying the usual deletion strategy, based on *as* and the fitness estimate F, deletion causes an increase in the specificity of $[P]$ early on as shown in Figure 5.5b). This longer convergence time is attributable to the bias on *as* as already observed in Figure 5.4b). The additional fitness bias causes hardly any observable influence.

Overall, it can be seen that in a random function fitness causes an intrinsic pressure towards higher specificity. This pressure is due to the parameter initialization method and the higher variance in more specific classifiers. The on-average higher fitness in more specific classifiers causes fitness pressure and deletion pressure to favor those more specific classifiers; thus the resulting undirected slight pressure towards higher specificity. Note that the specificity change and the convergence to a particular specificity level observed in Figure 5.5b) should essentially take place in all problems that are similar to a random function. This is particularly the case if classifiers are overgeneral and the investigated problem provides no fitness guidance from the overgeneral side.

Steady State Specificity Distribution

The performance evaluations showed that the specificity equation is correct. Initialization influences and variance effects usually result in a slightly higher specificity than predicted.

From Equation 5.8 it is possible to derive the specificity a population is expected to converge to (regardless of the initial specificity) assuming no fitness influence. Setting the difference $(\sigma[A]) + \Delta_{mx}(\sigma[A]) - (\sigma[P])$ to zero, we derive for free mutation

$$\begin{gathered}\sigma[A] + \Delta_{mf}(\sigma[A]) = \sigma[P] \\ \frac{\sigma[P]}{2-\sigma[P]} + \frac{\mu}{2}\left(2 - 3\frac{\sigma[P]}{2-\sigma[P]}\right) = \sigma[P],\end{gathered} \tag{5.9}$$

solving for $\sigma[P]$:

$$\sigma[P] = \frac{1 + 2.5\mu - \sqrt{6.25\mu^2 - 3\mu + 1}}{2}, \tag{5.10}$$

solving for μ:

$$\mu = \frac{\sigma[P] - \sigma[P]^2}{2 - \frac{5}{2}\sigma[P]}. \tag{5.11}$$

Similarly, we derive for niche mutation

$$\begin{gathered} \sigma[A] + \Delta_{mn}(\sigma[A]) = \sigma[P] \\ \frac{\sigma[P]}{2 - \sigma[P]} + \mu \left(1 - \frac{2\sigma[P]}{2 - \sigma[P]}\right) = \sigma[P], \end{gathered} \tag{5.12}$$

solving for $\sigma[P]$:

$$\sigma[P] = \frac{1 + 3\mu - \sqrt{9\mu^2 - 2\mu + 1}}{2}, \tag{5.13}$$

solving for μ:

$$\mu = \frac{\sigma[P] - \sigma[P]^2}{2 - 3\sigma[P]}. \tag{5.14}$$

The above equations enable us to determine the expected specificity in the population given a fixed mutation probability. On the other hand, given a desired specificity in the population, we can determine the mutation probability necessary to achieve that specificity.

Table 5.1 shows the resulting specificities in theory and empirically determined in a random function (that is, a Boolean function that returns uniformly randomly either zero or one for each problem instance) and a constant function (that is, a Boolean function that returns always one). Evaluated are niche mutation and free mutation. Empirical runs were carried through using either proportionate selection or tournament selection, which is introduced in Section 5.2.3. In the random function, the empirical results show slightly higher values than predicted by the theory. We note that the resulting specificity values can be roughly approximated by twice the value of mutation. The higher specificity values are mainly due to offspring initialization effects in conjunction with the application frequency of classifiers. In the case of low mutation rates, the fitness decrease in offspring classifiers is quickly overruled by higher relative accuracies. In the case of high mutation rates, the fitness decrease causes lower consequent specificities. These effects are enhanced by the much stronger pressure induced by tournament selection. Figure 5.6 shows the convergence values of the table in graphical form.

Table 5.2 shows necessary mutation rates for desired specificities in theory and derived from experiments. Due the specialization effects caused by offspring initialization, the mutation rate needs to be set slightly lower than

Table 5.1. Converged specificities from empirical results in a constant and a random function with free mutation and niche mutation with proportionate selection and tournament selection as well as in theory.

	Constant Function				Random Function					
	Free Mu.		Niche Mu.		Free Mu.		Niche Mu.		Theory	
μ	Prop.S.	Tour.S.	Prop.S.	Tour.S.	Prop.S.	Tour.S.	Prop.S.	Tour.S.	Free Mu.	Niche Mu.
0.01	0.02091	0.10846	0.02098	0.10773	0.03396	0.16846	0.03276	0.16456	0.01990	0.01980
0.02	0.04034	0.11512	0.04022	0.11358	0.06615	0.19667	0.06447	0.18864	0.03959	0.03918
0.04	0.07846	0.12953	0.07720	0.12641	0.12856	0.23261	0.12300	0.21903	0.07830	0.07668
0.06	0.11418	0.14594	0.11119	0.14076	0.18233	0.25920	0.17180	0.24384	0.11606	0.11240
0.08	0.15109	0.16560	0.14509	0.15790	0.22668	0.27971	0.21360	0.26364	0.15279	0.14629
0.10	0.18357	0.18712	0.17554	0.17743	0.25995	0.29965	0.24340	0.28007	0.18839	0.17830
0.12	0.21762	0.21116	0.20539	0.19864	0.28872	0.31810	0.26615	0.29549	0.22280	0.20841
0.14	0.24856	0.23706	0.23055	0.22128	0.31172	0.33538	0.28641	0.30855	0.25592	0.23661
0.16	0.28025	0.26089	0.25655	0.24380	0.33549	0.35106	0.30432	0.32357	0.28769	0.26293
0.18	0.30818	0.28485	0.27956	0.26548	0.35761	0.36812	0.32255	0.33633	0.31803	0.28740
0.20	0.33648	0.30964	0.30244	0.28554	0.37750	0.38494	0.33820	0.34983	0.34689	0.31010

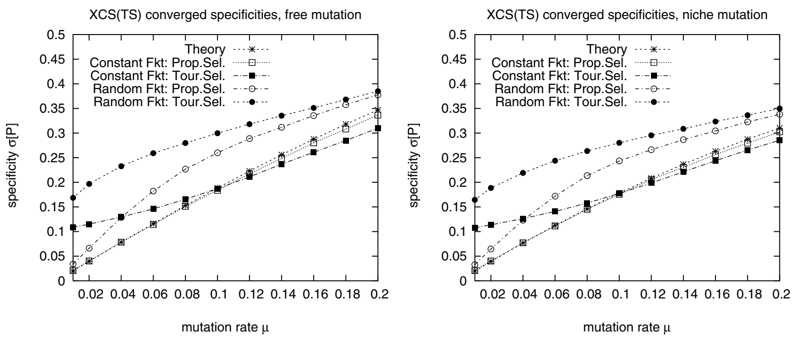


Fig. 5.6. Specificity is strongly dependent on mutation rate and mutation type. In a constant function, the theory nearly matches the resulting specificities. Fitness differences due to offspring initialization and frequency of evaluation change the specificity distribution, especially when tournament selection is applied.

suggested by the theory. Tournament selection enhances this effect since it focuses selection pressure, considering even small fitness differences significant. Note that apart from its effect on specificity, if mutation probability is set too high, mutation may be disruptive, destroying important parental structural information. The result would be an undesired random problem search.

5.2 Improving Fitness Pressure

The previous section has shown that XCS has an intrinsic generalization pressure that needs to be overcome by a sufficiently strong fitness pressure in order to evolve accurate classifiers. Selection in XCS and other LCSs has always

Table 5.2. Mutation settings for desired specificities. Empirical results are derived from simulations in a random function, which returns zero or one equally probable for each problem instance.

		0.05	0.1	0.15	0.2	0.25	0.3	0.4	0.5	0.6
	theory	0.0253	0.0514	0.0785	0.1067	0.1364	0.1680	0.2400	0.3333	0.4800
free mut.	prop.sel.	0.02	0.03	0.05	0.07	.10	0.14	$> .2$	$> .2$	$> .2$
	tour.sel.	< 0.01	< 0.01	0.01	0.02	0.06	0.10	$> .2$	$> .2$	$> .2$
	theory	0.0257	0.0529	0.0823	0.1143	0.1500	0.1909	0.3000	0.5000	–
niche mut.	prop.sel.	0.02	0.03	0.05	0.08	0.11	0.16	$> .2$	$> .2$	$> .2$
	tour.sel.	< 0.01	< 0.01	0.01	0.03	0.07	0.14	$> .2$	$> .2$	$> .2$

been done by means of proportionate selection. As we have seen in Chapter 2, though, proportionate selection results in a fitness pressure that is strongly dependent on fitness scaling and the fitness distribution in the population (Baker, 1985; Goldberg & Deb, 1991; Goldberg & Sastry, 2001; Goldberg, 2002).

This section shows that XCS can actually suffer from the pitfalls of proportionate selection observed in the GA literature. Moreover, we show that tournament selection with tournament sizes proportional to the action set size can solve the problem resulting in a strong, stable, and reliable fitness pressure towards more accurate classifiers.

5.2.1 Proportionate vs. Tournament Selection

Proportionate selection was applied and analyzed in Holland's original GA work (Holland, 1975). However, proportionate selection strongly depends both on fitness scaling (Baker, 1985; Goldberg & Deb, 1991) as well as on the current fitness distribution in the population. The smaller the fitness differences in the population the smaller the fitness pressure. Goldberg and Sastry (2001) show that evolutionary progress stalls when a population comes close to convergence since the fitness differences are not sufficiently strong anymore.

Fitness of XCS classifiers is derived from the scaled, set-relative accuracy. Although fitness scaling usually works well and proportionate selection is applied in the current action sets and not in the whole population, the more similarly accurate classifiers are, the less fitness pressure due to proportionate selection is expectable. In effect, similar accuracy of all classifiers in an action set should decrease or even annihilate fitness pressure.

Tournament selection, on the other hand, does not care about the current relative fitness differences. What matters is fitness rank. Thus, tournament selection does not suffer from fitness scaling nor from very small differences in accuracy. As long as there are significant differences in accuracy, tournament selection detects them and propagates the higher accurate classifier.

The next section shows that proportionate selection does not only suffer from cases in which classifiers are expected to have similar fitness values, but fitness pressure can actually be insufficiently strong. We propose set-size relative tournament selection as the remedy and confirm its superior performance in the exemplar multiplexer problem.

5.2.2 Limitations of Proportionate Selection

To reveal the limitations of proportionate selection, we apply XCS to the multiplexer problem with various parameter settings or with additional noise in the problem. We show that learning in XCS with proportionate selection is disrupted if the learning parameter β is set too low or if problem noise is set too high. The multiplexer problem is introduced in Appendix C.[1]

Figure 5.7 reveals the strong dependence on parameter β. Decreasing the learning rate hinders XCS from evolving an accurate problem solution. The problem is that initially overgeneral classifiers occupy a big part of the population. Better offspring often lose against the overgeneral parents since the fitness of the offspring only increases slowly (due to the low β value). Small differences in the fitness F only have small effects when using proportionate selection. Altering the slope of the accuracy curve by changing parameters α and ε_0 does not have any positive learning effect.

Figure 5.8 reveals XCS's dependence on initial specificity. Increasing $P_\#$ (effectively decreasing initial specificity) impairs the learning speed of XCS, since fitness does not cause sufficient specialization pressure. Decreasing the mutation rate μ also has a detrimental effect strongly delaying learning progress. The generalizing set pressure appears to be often stronger than the specializing fitness pressure so that the chances of reaching higher accurate classifiers are significantly decreased.

Additionally, we detected significant parameter initialization effects when using proportionate selection. If prediction p and reward prediction error ε are set to the parental values, learning speed is slightly decreased. Since in our problems only zero or thousand reward is possible and reward prediction is set directly to either one of the values, if the classifier is accurate, its error will decrease faster and fitness will increase faster.

In addition to the dependency on parameters β, $P_\#$, μ, and initialization, we can show that XCS with proportionate selection is often not able to solve noisy problems. We added two kinds of noise to the multiplexer problem: (1) Gaussian noise with a standard deviation σ is added to the payoff provided by the environment; (2) The payoff is swapped with a certain probability, termed *alternating noise* in the remainder of this work. Figures 5.9 and 5.10 show that XCS's performance is strongly degraded when adding only a small

[1] Unless stated otherwise, all results in this section are averaged over 50 experimental runs. Performance is assessed by test trials in which no learning takes place and the better prediction array value is chosen as the classification. During learning, classifications are chosen at random. Parameters are set as follows: $N = 2000$, $\beta = 0.2$, $\alpha = 1$, $\varepsilon_0 = .001$, $\nu = 5$, $\theta_{GA} = 25$, $\chi = 1.0$, $\mu = 0.04$, $\theta_{del} = 20$, $\delta = 0.1$, $\theta_{sub} = 20$, and $P_\# = 0.6$.

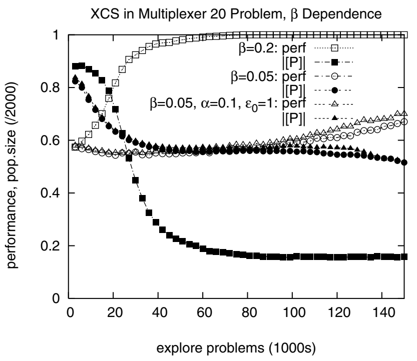


Fig. 5.7. A lower learning rate β decreases XCS learning performance. Accuracy function parameters have no immediate positive influence.

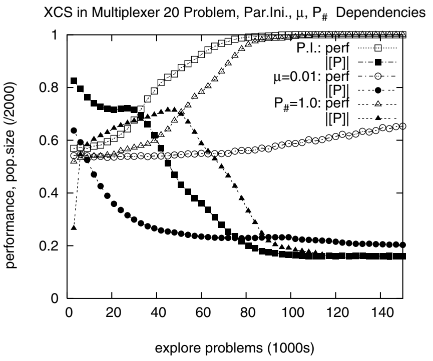


Fig. 5.8. Proportionate selection is strongly dependent on offspring initialization, initial specificity, and mutation rate.

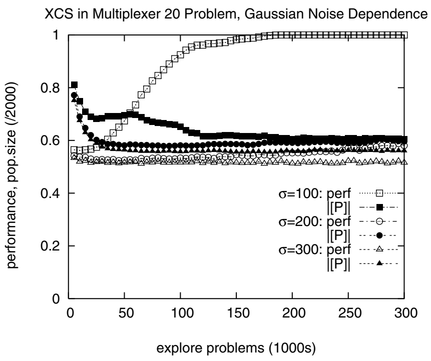


Fig. 5.9. Adding Gaussian noise to the payoff function of the multiplexer problem significantly deteriorates performance of XCS.

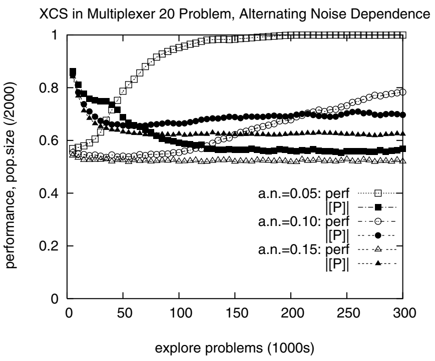


Fig. 5.10. Also alternating noise, in which a percentage of examples is assigned to the incorrect class, significantly degrades performance of XCS.

amount of either noise. Similar observations were made in Kovacs (2003), where Gaussian noise was added to the reward in some of the output classes.

In general, the more noise is added, the smaller the fitness difference between accurate and inaccurate classifiers. Thus, selection pressure decreases due to proportionate selection and the population starts to drift at random. Lanzi (1999c) proposed an extension to XCS that detects noise in environments and adjusts the error estimates accordingly. This approach, however, does not solve the parameter dependencies nor problems in which noise is not equally distributed over the problem space.

5.2.3 Tournament Selection

In contrast to proportionate selection, tournament selection is independent of fitness scaling (Goldberg & Deb, 1991). In tournament selection parental classifiers are not selected proportional to their fitness, but tournaments are held in which the classifier with the highest fitness wins (stochastic tournaments are not considered herein). Participants for the tournament are usually chosen at random from the population in which selection is applied. The size of the tournament controls the selection pressure. Fixed tournament sizes are generally used in GAs.

Compared to standard GAs, the GA in XCS is a steady-state, niche GA. Only two classifiers are selected in each GA application and selection is restricted to the classifiers in the current action set. Thus, some classifiers might not get any reproductive opportunity at all before being deleted from the population. Additionally, action set sizes can vary significantly. Initially, action sets are often over-populated with overgeneral classifiers. Thus, a relatively strong selection pressure, which adapts to the current action set size, appears to be necessary.

Thus, effective tournament selection in XCS holds tournaments of sizes dependent on the current action set size $|[A]|$ choosing a subset of size $\tau|[A]|$ ($\tau \in (0, 1]$) of the classifiers in $[A]$.[2] Instead of proportionate selection, two independent tournaments are held in which the classifier with the highest fitness is selected. The tournament selection procedure is described in algorithmic form in Appendix B.

The action set size proportionate tournament size assures that the current best classifier (assuming only one copy) is selected at least once with probability $1 - (1 - \tau)^2$. For example, if the tournament size is set to $\tau = 0.4$ of the population, we assure that the maximally accurate classifier is part of at least one of the two tournaments with a probability of 0.64. On average, $2\tau^2 + 2(\tau(1 - \tau)) = 2\tau$ optimal classifiers are selected. The more copies of the best classifier exist, the higher the probability. This derivation is impossible when fixed tournament sizes are used since the action set size continuously varies and consequently the probability of selecting the best classifier continuously varies as well.

XCSTS in the Previous 20 Multiplexer Settings

Figures 5.11 and 5.12 show that XCS with tournament selection, referred to as *XCSTS*, can solve the 20 multiplexer problem even with a low parameter value β, a low parameter value μ, or a high parameter value $P_{\#}$. XCSTS is also more independent from initial parameter settings. The much stronger and stable fitness pressure overcomes the generalizing set pressure even without the help of mutation pressure. A reliable and stable performance increase is observable.

[2] If not stated differently, τ is set to 0.4 in the subsequent experimental runs.

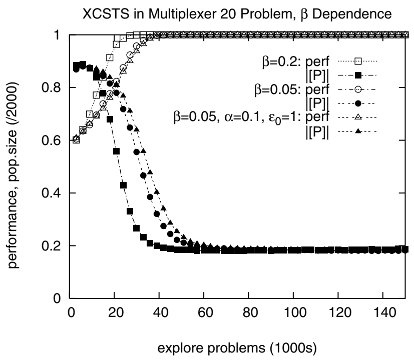


Fig. 5.11. XCSTS is only marginally influenced by a decrease in learning rate β. As in the proportionate selection case, accuracy parameters have nearly no learning influence.

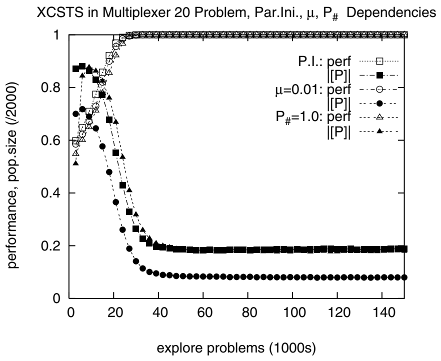


Fig. 5.12. Other parameter variations such as a lower initial specificity ($P_{\#} = 1.0$), a low mutation rate ($\mu = 0.01$), or a different offspring initialization ($P.I.$) hardly influences XCSTS's learning behavior.

Figures 5.13 and 5.14 show that XCSTS is much more robust in noisy problems as well. XCSTS solves the same Gaussian noise 20 multiplexer problem with nearly no performance degradation (Figure 5.13). Despite the noisy parameter estimation values, tournament selection detects the more accurate classifiers generating a sufficiently strong fitness pressure. The decrease in the differences of accuracy due to the additional noise hardly affects XCSTS. Also in the alternating noise case, XCSTS reaches a higher performance level (Figure 5.14). Note that as expected, the population sizes do not converge to the sizes achieved without noise, since subsumption does not apply. Nonetheless, in both noise cases the population sizes decrease indicating the detection and convergence to maximally accurate classifiers.

Tournament Selection with Fixed Size

XCS's action sets vary in size and in distribution. Dependent on the initial specificity in the population (controlled by parameter $P_{\#}$), the average action set size is either large or small initially. As was shown above, the average specificity in an action set is on average smaller than the specificity in the whole population. Replication in action sets and deletion from the whole population results in an implicit generalization pressure that can only be overcome by a sufficiently large specialization pressure. Additionally, the distribution of the specificities depends on initial specificity, problem properties, the resulting fitness pressure, and learning dynamics. Thus, an approach with fixed tournament size is dependent on the particular problem and probably not flexible enough. Kovacs used fixed tournament sizes to increase fitness pressure in his comparison to a strength-based XCS version (Kovacs, 2003).

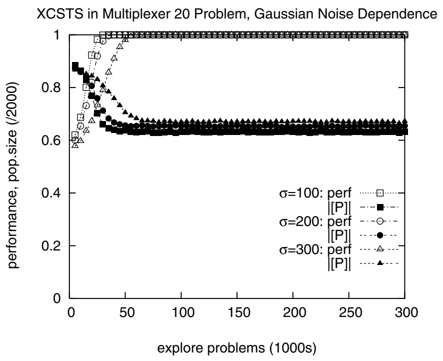


Fig. 5.13. XCSTS performs much better than XCS in noisy problems. Performance is hardly influenced when adding Gaussian noise of up to $\sigma = 300$.

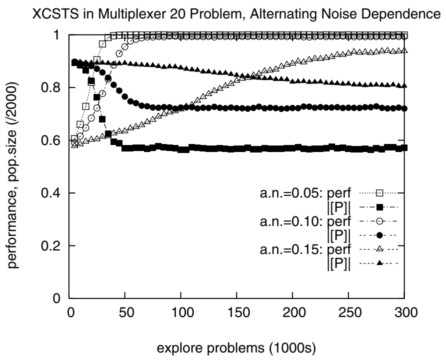


Fig. 5.14. Also in the alternating noise case, XCSTS's learning behavior is faster and more reliable.

In Figure 5.15 we show that XCSTS with fixed tournament size τ_f only solves the multiplexer problem with the large tournament size of $\tau_f = 12$. Since the population is usually over-populated with overgeneral classifiers early in a run, action set sizes are large so that a small tournament size mainly causes competition only among overgeneral classifiers. Thus, not enough fitness pressure is generated. When adding noise, an even larger tournament size is necessary (Figure 5.16). A tournament size of $\tau_f = 32$, however, does not allow any useful recombinatory events anymore since the action set size itself is usually not much bigger than that. Thus, fixed tournament sizes are inappropriate for XCS's selection mechanism.

Different Tournament Sizes

A change in the relative tournament size effectively changes the strength of the selection pressure applied. While a tournament size $\tau = 0$ corresponds to random selection, $\tau = 1$ corresponds to a deterministic selection of the classifier with the currently highest fitness in the action set, and thus the strongest selection pressure. As can be seen in Figure 5.17 and Figure 5.18, XCSTS is able to generate a complete and accurate problem representation for a large range of τ. However, if selection pressure is too weak, learning may not take place at all or may be delayed. On the other hand, if selection pressure is very strong, crossover never has any effect since identical classifiers are crossed. The lack of effective recombinations hardly influences XCS performance in the multiplexer problem as shown in Figure 5.18. However, in other problems ineffective crossover may strongly impair XCS's learning capabilities. Thus, τ may not be set to 1. In practice, a value of 0.4 proved to be robust.

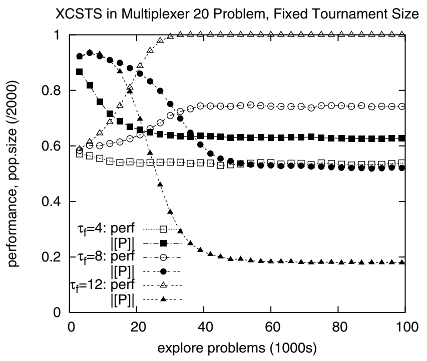


Fig. 5.15. Due to fluctuations in action set sizes and distributions as well as the higher proportion of more general classifiers in action sets, fixed tournament sizes are inappropriate for XCS selection.

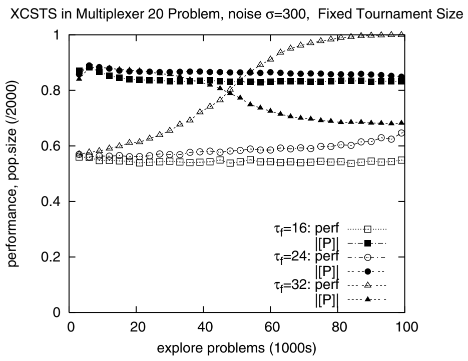


Fig. 5.16. Tournament selection with fixed tournament size requires large sizes to solve noisy problems. Large action sets prevent sufficiently strong selection pressure when tournament sizes are too small.

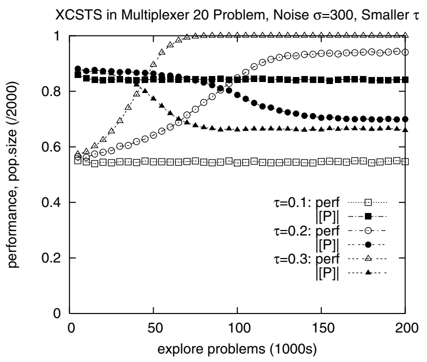


Fig. 5.17. The strength of the selection pressure can be manipulated by the tournament size proportion. Very small proportions result in an insufficiently strong pressure.

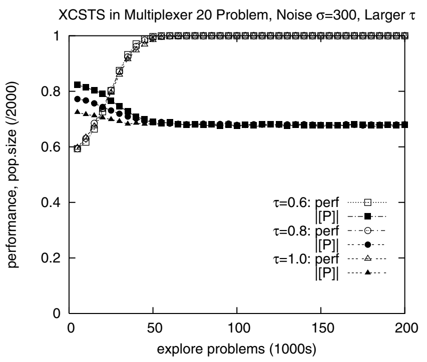


Fig. 5.18. Very large tournament set size proportions may prevent effective recombination. In the multiplexer problem, this restriction results in hardly any performance influence.

Specificity Guidance Exhibited

As theorized above, the two selection methods differ in their dependence on the fitness estimate and fitness distribution. The fitness pressure, resulting from proportionate selection, depends on fitness scaling and in particular on the relative amount of fitness difference. Tournament selection on the other hand only depends on the fitness difference itself and not on the amount of the difference. That is, as long as the more accurate classifiers have a higher

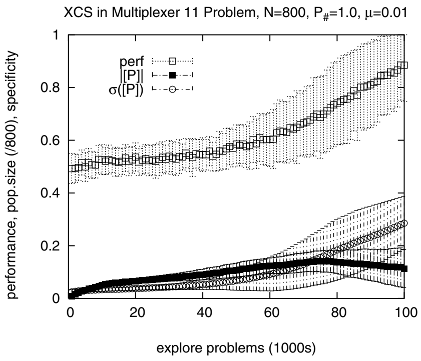


Fig. 5.19. Starting with an overgeneral population and low mutation rate, XCS is not able to pick up the accuracy signal reliably.

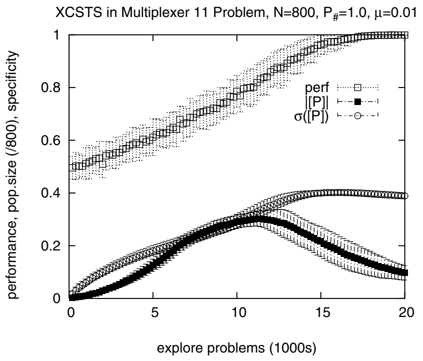


Fig. 5.20. XCSTS immediately pushes the population towards higher specificity and thus higher accuracy.

fitness estimate (regardless of how much higher the estimate is), tournament selection causes fitness pressure towards higher accuracy.

To exhibit this pressure, we monitor the average specificity in the population as well as the average standard deviation of the specificity. Figures 5.19 and 5.20 show the change in specificity when starting with a completely general population ($P_\# = 1.0$) in the 11 multiplexer problem. Figure 5.20 shows that XCSTS immediately identifies higher accurate classifies causing the specificity to rise. XCS, on the other hand, stalls at the overgeneral level apparently relying on a lucky guess for successful learning (Figure 5.19).

Another indicator can be found in the performance increase showing also the standard deviation over the experiments. Since XCSTS detects the fitness guidance in the problem immediately, the standard deviation between the runs remains small and the performance is hardly affected by a higher initial generality. Performance of XCS with proportionate selection, on the other hand, is much less reliable and strongly depends on initial generality. If there is no accurate classifier generated initially, proportionate selection has problems generating a sufficiently strong fitness guidance.

5.3 Summary and Conclusions

This chapter has shown how XCS evolves a complete, accurate, and maximally general problem solution. Several *evolutionary pressures* guide the initial classifier population to the solution.

1. *Fitness pressures* is the main pressure towards higher accuracy.
2. *Set pressure* causes classifier generalization towards higher *semantic* generality.

3. *Mutation pressure* causes diversification searching in the syntactic neighborhood of currently best subsolutions. Mutation also has an effect on specificity.
4. *Deletion pressure* emphasizes the maintenance of a complete solution.
5. *Subsumption pressure* propagates accurate classifiers that are *syntactically* more general.

Combining set pressure and mutation pressure, we derived a *specificity equation*, which is able to predict the expected specificity change in an XCS population as long as no additional fitness pressure applies. In conjunction with fitness pressure and subsumption pressure, the pressures push towards an equilibrium that coincides with the desired complete, maximally accurate, and maximally general problem solution.

However, to reach this equilibrium, fitness pressure needs to be strong enough to reliably overcome the generalizing set pressure. We showed that proportionate selection may not be sufficiently strong due to parameter influences as well as problem influences such as a noisy reward function. *Tournament selection* with tournament sizes proportional to the current action set size solves these drawbacks. Since the tournament size is set proportionate to the current action set size, the minimal probability of selecting the most accurate classifier is fixed so that the minimal fitness pressure is also fixed. We showed that XCS with tournament selection is able to solve the investigated problems reliably, confirming the theorized reliable fitness pressure towards higher accuracy.

Note however that tournament selection for deletion is actually inappropriate. Tournament selection strives to evolve the best classifiers and converges to these classifier fast. This is appropriate for reproduction because only one maximally accurate classifier or a few accurate, overlapping classifiers should evolve in each action set. Since deletion is applied population wide and XCS evolves a distributed problem solution, proportionate selection is the right choice in this case since it naturally maintains a distributed problem solution. Horn, Goldberg, and Deb (1994) provides a detailed analysis on the suitability of population wide proportionate selection with fitness sharing.

With respect to the facetwise theory approach for LCSs, we can now assure the first major aspects of the approach: (1) Due to the addition of tournament selection, fitness guides reliably to the intended solution. The accuracy-based fitness approach prevents strong overgenerals; (2) Parameters are estimated appropriately using adapted Q-learning and the moyenne adaptative modifiée technique; (3) Appropriate generalization applies as quantitatively analyzed in the *specificity equation*. Other influences with respect to generalization cause (crossover) or slight additional specificity influence (deletion, parameter initialization, accuracy determination) but no disruption. Subsumption pushes towards maximally syntactically general, accurate classifiers as long as complete accuracy can be reached in the problem (reward prediction error ε drops below ε_0).

With the first main aspect of our facetwise LCS theory understood and satisfied, we are now ready to face the second aspect. The next chapter consequently investigates the computational effort necessary to ensure solution growth and sustenance.