For Real! XCS with Continuous-Valued Inputs

Christopher Stone christopher.stone@uwe.ac.uk
Faculty of Computing, Engineering and Mathematical Sciences
University of the West of England
Bristol, BS16 1QY, United Kingdom

Larry Bull larry.bull@uwe.ac.uk
Faculty of Computing, Engineering and Mathematical Sciences
University of the West of England
Bristol BS16 1QY, United Kingdom

Abstract

Many real-world problems are not conveniently expressed using the ternary representation typically used by Learning Classifier Systems and for such problems an interval-based representation is preferable. We analyse two interval-based representations recently proposed for XCS, together with their associated operators and find evidence of considerable representational and operator bias. We propose a new interval-based representation that is more straightforward than the previous ones and analyse its bias. The representations presented and their analysis are also applicable to other Learning Classifier System architectures.

We discuss limitations of the real multiplexer problem, a benchmark problem used for Learning Classifier Systems that have a continuous-valued representation, and propose a new test problem, the checkerboard problem, that matches many classes of real-world problem more closely than the real multiplexer.

Representations and operators are compared using both the real multiplexer and checkerboard problems and we find that representational, operator and sampling bias all affect the performance of XCS in continuous-valued environments.

Keywords

checkerboard problem, continuous-valued input, interval representation, learning classifier system, real multiplexer problem, operator bias, real input, representational bias, sampling bias, XCS

1 Introduction

XCS is a Learning Classifier System (Holland, 1986) introduced by Wilson (1995) in which a classifier's fitness for the Genetic Algorithm (GA) (Holland, 1975) is based on the accuracy of its predictions rather than its ability to receive reward. The XCS algorithm is described in detail in (Butz & Wilson, 2001). Like most Learning Classifier Systems, a ternary representation is usually used with XCS. However, many real-world problems are not conveniently expressed in terms of a ternary representation and several alternate representations have been suggested to allow Learning Classifier Systems to handle these problems more readily (Ahluwalia & Bull, 1999; Lanzi, 1999; Bonarini, 2000; Bull & O'Hara, 2002).

We concentrate here on the representation for continuous-valued inputs introduced by Wilson (2000); however, we also consider Wilson's (2001a) representation for

integer data, which has also been used for function approximation with XCS (Wilson, 2001b). Both of these representations replace the standard ternary representation. The only other changes made to XCS to accommodate the new representations are in the cover, mutation and GA subsumption operators. Two-point crossover is retained.

Although the arguments presented here specifically relate to XCS with a one of m binary encoding for real numbers, many of the issues are also relevant to other Learning Classifier System architectures that may use these representations and to a floating point encoding of real numbers.

The remainder of this paper is organized as follows. Section 2 introduces interval predicates, terminology and the real multiplexer problem. Sections 3 and 4 study the two representations for continuous-valued data introduced by Wilson, Centre-Spread Representation (CSR) and Ordered Bound Representation (OBR), by examining their properties and operators. In Section 5 we introduce a new representation, Unordered Bound Representation (UBR) and analyse it in the same way. Section 6 looks at the real multiplexer problem in more detail and Section 7 extends this discussion to hyper-rectangles, the decision surfaces constructed by interval predicates. In Section 8, we introduce a new test problem, the checkerboard problem and use both this and the real multiplexer problem in Section 9 to compare representations and operators. Section 10 provides conclusions to the work.

2 Interval Predicates

2.1 Motivation

XCS has been shown to generate complete and maximally general maps (Kovacs, 1996) for ternary representations. There is evidence (Wilson, 2000; Wilson 2001a) to suggest that XCS is able to do this for continuous and integer-valued domains.

Thus, XCS approximates the function mapping $X \times A \Rightarrow P$ where X represents the environment, A is the set of possible actions and P is the payoff received for executing a particular action in an environmental state. In this paper, we consider continuous-valued environments, $X \in \Re^n$ and Boolean actions, $A \in \{0, 1\}$.

Learning Classifier Systems in general and XCS in particular, typically use a ternary representation to encode the environmental condition that a classifier matches. Bits in the condition string of a classifier are allocated to represent the state of a single environmental variable, x_i. Exact matching in this way is generally not suitable for a continuous-valued environment, where real-valued data over a range must be represented. One possibility for continuous-valued environments is to encode the environment in the form of inequalities, $x_i < \theta_i$. The decision surface represented by a classifier is then a hyperplane in the n-dimensional solution space.

The representations considered here replace the $\{0, 1, \#\}$ classifier predicate with one representing a half-open interval $[p_i, q_i)$. This interval matches the environment if $p_i \leq x_i < q_i$. The classifier condition is a vector of length n, each element of which encodes such an interval. A classifier with such a representation describes a hyper-rectangle in solution space.

2.2 Terminology

To avoid confusion and aid precision, we adopt the following notation throughout this paper.

1. Intervals in phenotype space are tagged with the subscript p, e.g., $[0, 1)_p$
2. Intervals in genotype space are tagged with the subscript g, e.g., $[0, n]_g$

3. Tuples are distinguished from intervals by the absence of a subscript.

The solution space is $[p_{\min}, q_{\max})_p^n$, where $p_{\min}$ and $q_{\max}$ are the minimum and supremum of the interval. For clarity of presentation and without loss of generality, we assume that $p_{\min}$ and $q_{\max}$ are the same for all dimensions i of the solution space.

2.3 The Real Multiplexer

The Boolean multiplexer is a standard benchmark problem for Learning Classifier System evaluation. Wilson (2000) introduced the real multiplexer as a test problem for Learning Classifier Systems with continuous-valued inputs. Each 'bit' of the Boolean multiplexer is presented as a value x_i in the $[0, 1)_p$ interval, with $x_i < \theta_i$ meaning binary 0 and $x_i \geq \theta_i$ meaning binary 1. The value θ_i is a control parameter that may be varied to provide problems of varying difficulty. By default $\theta_i = 0.5 \; \forall \; i \leq n$ and this is the threshold used in this paper.

Experiments on XCS were performed using the 6-bit real multiplexer. XCS was presented with randomly generated (6 element) vectors of real numbers in the interval $[0, 1)$. For each of these random vectors, XCS suggested a binary action representing the output value of the multiplexer. For this, it was rewarded with a payoff of 1000 for the correct action and 0 otherwise.

XCS settings used for all real multiplexer experiments in this paper were $N = 800$, $\beta = 0.2$, $\alpha = 0.1$, $\varepsilon_0 = 10$, $\nu = 5$, $\theta_{GA} = 12$, $\chi = 0.8$, $\mu = 0.04$, $\theta_{del} = 20$, $\delta = 0.1$, $\theta_{sub} = 20$, $p_I = 10$, $\varepsilon_I = 0$, $f_I = 0.01$, $\theta_{mna} = 2$, $m = 0.1$, $s_0 = 1.0$. These match the settings published in (Wilson, 2000). GA subsumption, but no action set subsumption, was used. All experimental results presented are the average of 10 runs using alternate explore and exploit trials. A 16-bit binary encoding was used for real numbers.

Wilson showed that XCS was able to solve the 6-bit real multiplexer using Centre-Spread Representation. A duplicate of these results is shown in Figure 1. This shows the *system performance*, the fraction of correct actions averaged over the previous 50 exploit trials, the *system error*, the absolute difference of the payoff and the predicted payoff averaged over the previous 50 exploit trials and the *macroclassifier fraction*, the fraction of the population that are macroclassifiers. Figure 1 also shows other information that will be referred to later in the paper.

More recently, Bull, Wyatt and Parmee (2002) have shown that XCS can solve the 11-bit real multiplexer.

3 Centre-Spread Representation

3.1 Background

To extend XCS into continuous-valued environments, Wilson (2000) introduced the Centre-Spread Representation. Centre-Spread Representation provides a form of receptive field for Learning Classifier Systems. An interval predicate, $[p_i, q_i)_p$, is represented as a tuple (c_i, s_i) where $c_i, s_i \in \Re$. c_i encodes the centre of the interval and s_i encodes the spread (or width) of the interval. The interval is decoded as follows:

$$\begin{aligned} p_i &= \min(p_{\min}, c_i - s_i) \\ q_i &= \max(q_{\max}, c_i + s_i) \end{aligned}$$

Use of Centre-Spread Representation thus involves a genotype (c_i, s_i) to phenotype $[p_i, q_i)_p$ mapping, or gene expression.

Wilson does not provide details of the encoding used for real numbers in the Centre-Spread Representation. However, given that the solution space is bounded,

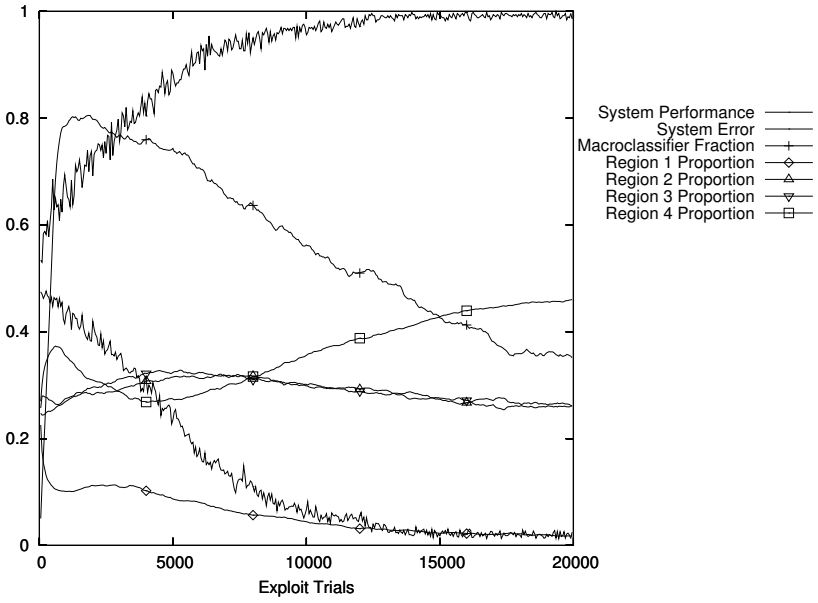

Figure 1: 6-bit real multiplexer with Centre-Spread Representation, standard cover with $s_0 = 1$, 2-point crossover and standard mutation

we assume a one of m binary encoding where the real values for both the centre and spread are encoded into binary integers of length k using the equation

$$\left\lfloor \frac{(2^k - 1)(p_i - p_{\min})}{q_{\max} - p_{\min}} \right\rfloor$$

With this scheme, the real values for centres and spreads are discretized into one of m possible values upon encoding. There are 2^{2k} possible centre-spread combinations and each possible centre-spread is represented exactly once. Because of the discretization of the phenotype, the half-open solution space $[p_{\min}, q_{\max})_p$ in phenotype space may be regarded as the closed solution space $[0, 2^k - 1]_g$ in genotype space.

The ternary representation used in most Learning Classifier Systems has an explicit 'don't care' value in the form of the '#' allele. Centre-Spread Representation does not have any explicit 'don't care' scheme. Instead, the maximally general interval $[p_{\min}, q_{\max})_p$ provides an implicit 'don't care' mechanism by matching all possible environmental inputs. An implication of this is that the proportion of maximally general intervals introduced into the population is not directly controllable by a system parameter, as is normally the case with a ternary representation.

3.2 Properties

As the solution space is half-open, the centre-spread genotype must be limited upon expression in order to restrict the range of the phenotype to the interval $[p_{\min}, q_{\max})_p$. We refer to this process as *truncation*. Truncation means that the genotype to phenotype ($g \rightarrow p$) mapping is non-linear and many to one. In short, it is possible for a phenotype to be represented by more than one genotype. As an example, consider the solution space interval $[0, 1)_p$. The phenotype $[0.5, 1)_p$ may be represented as centre-spread tuples (0.75, 0.25), (0.8, 0.3), (0.9, 0.4), (1, 0.5) or any number of other tuples.

Table 1: Phenotype frequency matrix for Centre-Spread Representation with $k = 3$

		q_i							
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
p_i	**0**	1	1	2	2	3	3	4	20
	1		1	0	1	0	1	0	4
	2			1	0	1	0	1	3
	3				1	0	1	0	3
	4					1	0	1	2
	5						1	0	2
	6							1	1
	7								1

In practice, the number of possible centre-spread tuples representing an interval is finite, due to the discretization necessary when representing real numbers. The number of possible genotypes for a particular phenotype is therefore determined by both the phenotype itself and the details of the encoding of real numbers employed.

If there were no truncation on $g \rightarrow p$ mapping (and thus a one to one $g \rightarrow p$ mapping), there would be 2^{2k} possible phenotypes. However, given the need for truncation, certain phenotypes are expressed from multiple genotypes. There are therefore less than 2^{2k} unique phenotypes with the Centre-Spread Representation. Certain phenotypes are 'missing' and we refer here to these missing phenotype to genotype ($p \rightarrow g$) mappings as *holes*.

The above two properties mean that using Centre-Spread Representation:

1. Expression of random genotypes results in increased frequency of expression of certain phenotypes over other possible phenotypes.

2. The phenotype space contains holes where certain phenotypes are missing, as they are not expressible.

To examine these phenomena in more detail we enumerated the $g \rightarrow p$ mapping for one of m binary encodings of length $2 \leq k \leq 12$. Without loss of generality, $p_{\min} = 0$ and $q_{\max} = 2^k - 1$.

As a readily understandable example of one of these enumerations, Table 1 shows the frequency of each possible phenotype for $k = 3$, a 3-bit one of m binary encoding of real values[1].

The phenotypic frequency in Table 1 shows several interesting properties:

1. The frequency of all possible phenotype intervals is not uniform (as already discussed).

2. Certain phenotype intervals have zero frequency (as already discussed).

3. The frequency of an exact number $[p_i, p_i]_p$ is always 1.

4. The frequency of intervals of the form $[p_{\min}, q_i)_p$ increases as q_i increases and the frequency of intervals of the form $[p_i, q_{\max})_p$ increases as p_i decreases.

[1] That is, 3 bits each for centre and spread.

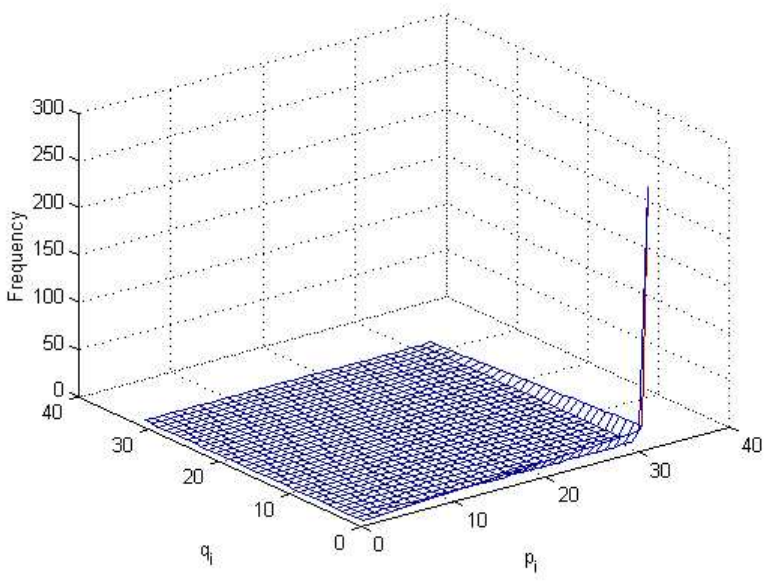

Figure 2: Phenotype frequency landscape for Centre-Spread Representation with $k = 5$

5. The frequency of the $[p_{\min}, q_{\max})_p$ interval is much greater than any other.

All encodings $2 \leq k \leq 12$ show the same patterns and properties. Only the specific frequencies vary. Space does not permit the publication of the details of each of these, but as a further example and aid to visualization, Figure 2 shows the frequency matrix for $k = 5$ plotted as a surface that may be viewed as the landscape of the $g \rightarrow p$ mapping for that particular encoding.

From these results, we can state certain properties of the Centre-Spread Representation.

Property 1: Many to one genotype to phenotype mapping. This property is the key property from which all others derive and has already been discussed in detail.

Property 2: Incomplete phenotype to genotype mapping. A corollary of the discretization of the centre and spread and the many to one $g \rightarrow p$ mapping (Property 1) is that the $p \rightarrow g$ mapping is undefined for certain phenotypes.

Holes arise because of the discretization of the encoding:

1. The centre must be located at a point that can be represented using the discrete encoding (i.e., it must be integer-valued).
2. The spread must be able to be represented using the discrete encoding (it must also be integer-valued).

For example, consider interval $[1, 2]_p$ in Table 1. This interval cannot be represented using Centre-Spread Representation and an encoding of length $k = 3$. Neither of the two requirements can be met for the interval above. The centre would need to be located at point 1.5 with a spread of 0.5.

In general, any interval where $q_i - p_i$ is odd is unable to be represented using Centre-Spread Representation. Because of this, where holes exist in the $g \rightarrow p$ mapping, they are uniformly distributed around the solution space, such that they are neighbours with a $g \rightarrow p$ mapping that has non-zero frequency (i.e., is a one-to-one or many-to-one mapping). No two holes are ever situated next to each other.

The presence of holes is undesirable in the $g \rightarrow p$ mapping, since it means that certain phenotypes cannot be expressed. This, in turn, means that the accuracy of an expressed phenotype is lower than it would otherwise be since the effective discretization of the phenotype is coarser than desired. However, the fact that holes are always located next to a phenotype that can be expressed places a lower limit on the effective discretization of the Centre-Spread Representation.

Holes are an artefact from using one of m binary encoding. With a floating point encoding, the holes would be small enough to cause no practical problems.

Property 3: The genotype to phenotype mapping of exact numbers is one to one. Intervals of the form $[p_i, p_i]_p$ are the leading diagonal of the frequency matrix and represent exact numbers or points in the solution space. These may be necessary to represent a solution in a particular problem and always have frequency one (i.e., a one-to-one $g \rightarrow p$ mapping). A corollary of this is that all possible exact numbers can be represented using Centre-Spread Representation.

Property 4: The frequency of intervals of the form $[p_{\min}, q_i)_p$ and $[p_i, q_{\max})_p$ increases as p_i decreases or q_i increases. Property 1 states that certain phenotypes can be expressed from multiple genotypes. Property 4 provides more detail on the nature of this many to one mapping.

As seen in Figure 2, the $g \rightarrow p$ landscape is characterized by a flat plateau containing the majority of $g \rightarrow p$ mappings. These exist with a frequency of either 0 or 1 (i.e., not expressible, or one-to-one mapping). All of the one-to-many $g \rightarrow p$ mappings occur in phenotypes of the form $[p_{\min}, q_i)_p$ or $[p_i, q_{\max})_p$. The special case of the $[p_{\min}, q_{\max})_p$ phenotype is discussed in Property 5.

Moreover, the frequency of mappings increases as either p_i decreases or q_i increases. Thus, wide (general) intervals 'anchored' at $p_{\min}$ or $q_{\max}$ have a greater frequency of expression than narrow (specific) intervals or those not anchored at $p_{\min}$ or $q_{\max}$.

Property 5: The frequency of the $[p_{\min}, q_{\max})_p$ interval is much greater than that of any other interval. Property 4 states that the expression frequency of intervals of the form $[p_{\min}, q_i)_p$ and $[p_i, q_{\max})_p$ increases as p_i decreases or q_i increases. A special case of this is the interval $[p_{\min}, q_{\max})_p$. The expression frequency of this interval is much larger than any other and it completely dominates the $g \rightarrow p$ landscape, as shown in Figure 2.

The $[p_{\min}, q_{\max})_p$ interval has special significance for interval-based representations, as it describes the maximally general interval predicate. A consequence of Property 5 is that randomly generated genotypes will be expressed as the maximally general interval predicate with a far higher frequency that would otherwise be expected.

To shed more light on these five properties, it is desirable to pursue a quantitative approach. In particular, we wish to understand the nature of the increased expression frequency of $[p_{\min}, q_i)_p$, $[p_i, q_{\max})_p$ and $[p_{\min}, q_{\max})_p$ intervals.

From examination of the results of enumerating the expression frequencies for real encodings of length $2 \leq k \leq 12$, we can derive equations for the total frequency of expression of all possible intervals of the form $[p_{\min}, q_i)_p$ and $[p_i, q_{\max})_p$:

$$f_{p_{\min},q_i} = 2^{2(k-1)} \; \forall \; p_{\min} \leq q_i < q_{\max}$$

$$f_{p_i,q_{\max}} = 2^{2(k-1)} \; \forall \; p_{\min} < p_i \leq q_{\max}$$

and frequency of expression of the $[p_{\min}, q_{\max})_p$ interval:

$$f_{p_{\min},q_{\max}} = 2^{2(k-1)} + 2^{k-1}$$

The number of possible $g \rightarrow p$ mappings is 2^{2k}, so the probabilities $P_{p_{\min},q_i}$ and $P_{p_i,q_{\max}}$ of expression of intervals of the form $[p_{\min}, q_i)_p$ and $[p_i, q_{\max})_p$ respectively, is given by

$$\begin{aligned} P_{p_{\min},q_i} &= \frac{2^{2(k-1)}}{2^{2k}} \\ &= \frac{2^{k-1}}{2^{k+1}} \\ &= 0.25 \end{aligned}$$

and similarly, $P_{p_i,q_{\max}} = 0.25$

The probability $P_{p_{\min},q_{\max}}$ of expression of the maximally general interval $[p_{\min}, q_{\max})_p$ is given by

$$\begin{aligned} P_{p_{\min},q_{\max}} &= \frac{2^{2(k-1)} + 2^{k-1}}{2^{2k}} \\ &= \frac{2^{k-1} + 1}{2^{k+1}} \\ \lim_{k\to\infty} &= \frac{2^{k-1}}{2^{k+1}} \\ &= 0.25 \end{aligned} \tag{1}$$

Although Equation 1 describes limiting behaviour for infinite length encoding of real numbers, actual values of $P_{p_{\min},q_{\max}}$ are close to 0.25 for values of k likely to be used in practice (8, 16 or 32 bit encodings). For example, for $k = 8$, $P_{p_{\min},q_{\max}} = 0.25195$ and for $k = 16$, $P_{p_{\min},q_{\max}} = 0.25001$.

$P_{p_{\min},q_{\max}}$ is the probability of the interval $[p_{\min}, q_{\max})_p$ being expressed on $g \rightarrow p$ mapping of a random genotype. Centre-Spread Representation thus includes a form of implicit 'don't care' mechanism, similar to that of the '#' allele in ternary representations. However, unlike ternary representations this $P_{\#}$ value is fixed at 0.25 and so is not adjustable to suit different problems.

The probability of expression of an interval of the form

$$[p_i, q_i)_p \; \forall \; p_i = p_{\min} \vee q_i = q_{\max}$$

is

$$P_{p_{\min},q_i} + P_{p_i,q_{\max}} + P_{p_{\min},q_{\max}} = 0.75$$

Table 2: Phenotype regions and their structural forms

	q_i	
p_i	Region 2 $[p_{\min}, q_i)_p$	Region 4 $[p_{\min}, q_{\max})_p$
	Region 1 $[p_i, q_i)_p$	Region 3 $[p_i, q_{\max})_p$

This value is essentially independent of encoding length.

So these 'special' phenotypes constitute 75% of all $g \rightarrow p$ mappings, yet only comprise $2^{k+1}-1$ of the 2^{2^k} possible $g \rightarrow p$ mappings. Their frequency therefore far exceeds what might be reasonably expected for a $g \rightarrow p$ mapping. In contrast, all remaining $g \rightarrow p$ mappings of the form $[p_i, q_i)_p \; \forall \; p_i > p_{\min} \wedge q_i < q_{\max}$ (the plateau in the $g \rightarrow p$ landscape) constitute only the remaining 25% of all mappings.

We can partition the phenotype space into four regions corresponding to the four different structural forms of interval predicate resulting from the properties of the encoding. Table 2 shows the four structural forms of interval predicate, together with the region number we shall, for convenience, assign to them. This diagram mimics the shape of the phenotype frequency matrix and shows the allocation of $g \rightarrow p$ mappings by region. In the diagram and the rest of this paper, unless otherwise noted,

$$p_{\min} < p_i \leq q_{\max} \;\wedge\; p_{\min} \leq q_i < q_{\max}$$

3.3 Operators

Since the real multiplexer is essentially a binary problem in disguise, solutions to the real multiplexer are expressed by an alphabet of three possible interval predicates directly corresponding to the Boolean multiplexer's {0,1,#} alphabet. For the real multiplexer, the solution interval predicates are $\{[0, \theta_i)_p, [\theta_i, 1)_p, [0, 1)_p\}$. However, these are exactly the forms of interval predicate found in regions 2, 3 and 4 of Table 2 that exhibit many to one $g \rightarrow p$ mappings and account for 75% of all $g \rightarrow p$ mappings! We may therefore expect that the choice of Centre-Spread Representation has a bearing on XCS' ability to solve the real multiplexer problem.

There are four places where the influence of the representation is felt in a Learning Classifier System: initialization, covering, crossover and mutation. We note that, for continuous-valued representations, GA subsumption is performed at the level of the phenotype and is independent of the representation in use.

3.3.1 Initialization

Where a population is generated at random by genotype, a non-uniform $g \rightarrow p$ mapping will affect the proportion of phenotypes expressed by the population. For Centre-Spread Representation, generation of random genotypes at initialization time will provide a population containing on average a proportion of

$$1 - \frac{1}{4^n} \tag{2}$$

of classifiers with one or more intervals of the correct structural form to solve the problem, i.e., those in regions 2,3 and 4. These can then be recombined by the GA to provide complete solutions. For the 6-bit real multiplexer considered here, this equates to

0.9998. In contrast, a one to one $g \rightarrow p$ mapping would provide a proportion of

$$\frac{(n-2)^2+1}{n^2}$$

For $n = 6$ this is 0.472. We note here that Figure 1 shows results obtained without an initial population.

3.3.2 Covering

In XCS' cover operator, the centre of the interval is fixed by the environmental state. For the real multiplexer problem, the environmental state is externally generated from the uniform probability distribution $U[0, 1)$. The cover operator for Centre-Spread Representation generates the spread from the uniform probability distribution $U[0, s_0)$. In Wilson's experiments $s_0 = 1$, so any spread $0 \leq s_i < 1$ is equally possible. Therefore, both centre and spread are drawn from $U[0, 1)$, so all possible centre-spreads are equally probable and the probabilities $P_{p_{\min},q_i}$, $P_{p_i,q_{\max}}$ and $P_{p_{\min},q_{\max}}$ also apply during covering. The cover operator, like initialization, thus generates classifiers with a 0.75 probability of being in region 2, 3 or 4 and that have a probability given by Equation 2 of one or more intervals of the correct structural form to solve the problem.

3.3.3 Crossover

We have determined the probability distribution for new classifiers generated by the initialization and cover operators. As it is applied with high probability, crossover has the opportunity to affect this distribution by its production of offspring. We examine the impact of crossover on a single interval predicate, represented as a centre-spread tuple. We are only interested in the action of crossover *when it occurs* for a specific interval predicate. For crossover to alter that interval, a crossover point must occur within the interval. All the crossover operators considered in this paper that allow a crossover point within an interval restrict the crossover point to occur between the two alleles representing the interval. If the crossover point happens to occur between intervals, the interval itself survives unscathed (although it is likely to be paired with other intervals).

As mathematical analysis of crossover is difficult, we enumerate centre-spread combinations for two parents. We enumerate only those parental intervals that have at least one point in common with each other, since XCS uses a niche GA. A factor in the enumeration is the length of the real encoding used. For consistency with the results presented for mutation, we used an encoding of length $k = 8$ and crossed over a single centre and spread with a fixed crossover point between the alleles. For each combination, we noted the region(s) that the parents occupied and the region(s) occupied by their children. Enumeration of all possible centre-spread combinations implies an equal probability of parental intervals across the four regions. As already discussed, this is the case for intervals generated by initialization and covering. From this we may readily calculate the expected proportions of offspring across regions. This is shown in Table 3.

Niche crossover with Centre-Spread Representation tends to preserve the distribution of intervals across regions, with a small bias from region 1 to region 4. Note that this effect only occurs for an interval that has a crossover point within the interval predicate. For crossover points between interval predicates the interval is unchanged, and so the distribution of intervals, and hence regions, cannot alter. The probability of an interval being disrupted in this way depends on n, the number of interval predicates in the classifier's condition and the type of crossover operator used.

Table 3: Phenotype proportions for Centre-Spread Representation with $k = 8$ and crossover within an interval

Region	Form	Parent population	Offspring population
1	$[p_i, q_i)_p$	0.25	0.225
2	$[p_{\min}, q_i)_p$	0.25	0.25
3	$[p_i, q_{\max})_p$	0.25	0.25
4	$[p_{\min}, q_{\max})_p$	0.25	0.275

3.3.4 Mutation

The mutation operator for XCS with Centre-Spread Representation mutates a classifier by adding or subtracting with equal probability an amount m_i drawn from $U[0, m)$. A setting of $m = 0.1$ was used for Wilson's real multiplexer experiments.

We examine the behaviour of mutation over the four regions by studying the probability of an offspring occupying a region, given each possible parental region. To do this, we enumerate all possible mutations for an interval using a setting of $m = 0.1$ for every centre-spread combination over a range of possible values of k, the real encoding length. In the actual Learning Classifier System, the alleles corresponding to the interval's centre and spread are independently mutated with probability μ, so we examine these separately. Mutation of the centre allele shifts the centre by the amount of the mutation, viz:

$$\begin{aligned} p_i &= c_i - s_i + m_i \\ q_i &= c_i + s_i + m_i \end{aligned}$$

Mutation of the spread allele alters the width of the interval:

$$\begin{aligned} p_i &= c_i - s_i - m_i \\ q_i &= c_i + s_i + m_i \end{aligned}$$

For brevity, and since both centre or spread alleles have equal probability of mutation, we average the individual results from these enumerations to provide a picture of the effect of a single mutation on the interval predicate[2]. Mutations for values of $4 \leq k \leq 8$ were examined, and a common pattern was seen across all enumerations. As an illustration of the results found, Figure 3 shows the transition diagram of a single mutation of the interval predicate for $k = 8$. In the diagram, the states are the four possible regions that the parent could occupy and the numbers next to the arrows indicate the probability of a transition to a particular region for the offspring. As these probabilities are rounded to two decimal places, we do not distinguish here between very low probabilities (<0.01) and zero probability.

We can see from the diagram that the vast majority of all possible mutations do not produce any migration of region between the parent and offspring (where a region maps to itself). A parent tends to generate offspring that occupy the same region as itself, so parents with the correct structural form of the real multiplexer solution, $[p_{\min}, q_i)_p$, $[p_i, q_{\max})_p$ and $[p_{\min}, q_{\max})_p$, will overwhelmingly produce offspring with these characteristics. This mutation operator therefore has the effect of refining the details of an interval, but is unlikely to cause a change of region from parent to offspring.

[2]In fact, results for the centre and spread alleles differed only slightly and the combined results shown here are also indicative of the individual results.

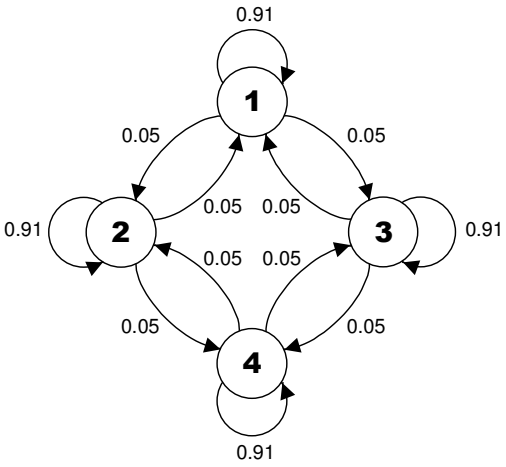

Figure 3: Region transition diagram for Centre-Spread Representation with $k = 8$ and standard mutation. Regions are represented by states and transition probabilities by arrows

It is possible for mutation to produce offspring that occupy a different region to that of the parent, but this occurs with a low probability. Migration between regions is possible via the two pathways $1 \leftrightarrow 2 \leftrightarrow 4$ and $1 \leftrightarrow 3 \leftrightarrow 4$ through which offspring are likely to progress through multiple mutation steps. These represent a partial ordering of regions. It is possible to travel in both directions along these pathways with roughly equal probability, showing that this mutation operator is broadly neutral with respect to transitions across regions.

4 Ordered Bound Representation

4.1 Background

Wilson (2001a) introduces a representation that describes an interval predicate by its lower and upper endpoints. This is used for problems requiring integer variables. However, when using a one of m binary encoding for real variables, the encoding for real numbers differs from that of integers only in the size of the alphabet used and the consequent size of the genotypic search space. There is no reason why the Ordered Bound Representation cannot also be used for continuous-valued problems and, indeed, the same representation is also used for real-valued function approximation in (Wilson, 2001b).

Ordered Bound Representation stores an interval predicate $[p_i, q_i)_p$ as a tuple (l_i, u_i) where for real-valued problems $l_i, u_i \in \Re$. l_i is the lower bound of the interval and u_i is the upper bound. We again assume that the alleles are encoded using a one of m binary encoding of length k.

One issue with Ordered Bound Representation is the ordering imposed on the alleles representing an interval predicate by the restriction that $l_i \leq u_i$. Wilson does not describe how this restriction is addressed, but essentially, any operator that could cause a situation where $l_i > u_i$ must check each interval predicate in a classifier's condition

and swap the lower and upper alleles if the ordering $l_i \leq u_i$ is violated as a result of the operation.

Apart from discretization in the real encoding, there is a direct mapping between the elements of the genotype and phenotype, so that $l_i \equiv p_i$ and $u_i \equiv q_i$. As a result, no explicit gene expression is necessary and, at the level of the representation, a one to one mapping exists for all possible phenotypes and their corresponding genotypes. No truncation occurs upon the expression since it is not possible to represent values outside the endpoints of the phenotype interval $[p_{\min}, q_{\max})_p$. The issues arising from the many to one $g \rightarrow p$ mapping with Centre-Spread Representation cannot arise with Ordered Bound Representation.

4.2 Properties

In Section 3.2 we stated certain properties that exist due to the many to one $g \rightarrow p$ mapping of Centre-Spread Representation. For reference, the equivalent properties of Ordered Bound Representation are:

Property 1: One to one genotype to phenotype mapping.

Property 2: Complete phenotype to genotype mapping.

Property 3: The genotype to phenotype mapping of exact numbers is one to one.

Property 4: The frequency of intervals of the form $[p_{\min}, q_i)_p$ and $[p_i, q_{\max})_p$ is constant for all p_i and q_i.

Property 5: The frequency of the $[p_{\min}, q_{\max})_p$ interval is the same as that of any other interval.

These are all due to the one to one $g \rightarrow p$ mapping that exists for Ordered Bound Representation. The representation stores all possible interval predicates with equal frequency and shows no bias towards certain types of interval predicate. This suggests that Ordered Bound Representation may be more suited for continuous-valued domains where the structure of the problem is *a priori* unknown.

The maximally general interval $[p_{\min}, q_{\max})_p$ is represented by a single tuple in Ordered Bound Representation, so given a random genotype, an interval predicate will be maximally general with probability

$$\frac{1}{2^{2k}}$$

As the size k of the real number encoding increases and thus its granularity becomes finer, the chances of a maximally general interval appearing in the initial population becomes exponentially lower. This is in contrast to the Centre-Spread Representation, which results in the maximally general interval being represented with an essentially fixed probability of 0.25. For this reason, Ordered Bound Representation effectively provides no implicit 'don't care' mechanism analogous to the '#' allele in a ternary representation.

4.3 Operators

We examined the effects of the interaction between Centre-Spread Representation and its operators in Section 3.3. Here we investigate Ordered Bound Representation and its associated operators.

4.3.1 Initialization

Where a population is generated at random by genotype, a uniform $g \rightarrow p$ mapping across intervals means that all possible phenotypes will exist in the population with identical probability. The frequency of expression of intervals of the form $[p_{\min}, q_i)_p$, $[p_i, q_{\max})_p$ and $[p_{\min}, q_{\max})_p$ (regions 2, 3, and 4) can be calculated as

$$\begin{aligned} f_{p_{\min},q_i} &= 2^k - 1 \; \forall \; p_{\min} \leq q_i < q_{\max} \\ f_{p_i,q_{\max}} &= 2^k - 1 \; \forall \; p_{\min} < p_i \leq q_{\max} \\ f_{p_{\min},q_{\max}} &= 1 \end{aligned}$$

The number of possible $g \rightarrow p$ mappings where $l_i \leq u_i$ is $2^{k-1}(2^k + 1)$, so the frequency of intervals of the form $[p_i, q_i)_p \; \forall \; p_i > p_{\min} \wedge q_i < q_{\max}$ (region 1) is given by

$$\begin{aligned} f_{p_i,q_i} &= 2^{k-1}(2^k + 1) - 2(2^k - 1) - 1 \\ &= (2^k - 1)(2^{k-1} - 1) \end{aligned}$$

The probability of expression of an interval of this form is

$$\begin{aligned} P_{p_i,q_i} &= \frac{(2^k - 1)(2^{k-1} - 1)}{2^{k-1}(2^k + 1)} \\ &= \frac{2^{2k-1} - 3(2^{k-1}) + 1}{2^{2k-1} + 2^{k-1}} \\ \lim_{k \rightarrow \infty} &= 1 \end{aligned}$$

Initialization therefore generates classifiers essentially exclusively in region 1. No classifiers are to be expected in a small, finite, population with the correct structural form of the solution as occurs in Centre-Spread Representation.

4.3.2 Covering

The cover operator generates a classifier containing intervals with the (l_i, u_i) tuples given by

$$\begin{aligned} l_i &= x_i - U[0, s_0) \\ u_i &= x_i + U[0, s_0) \end{aligned}$$

To match the experiments performed with Centre-Spread Representation [3] $s_0 = 1$. Note that, unlike the case with Centre-Spread Representation, the resulting interval is not generally centred on the environmental variable, x_i. This is the method adopted by Wilson, which we also use here. However, it would be trivial to alter the algorithm to emulate Centre-Spread Representation strategy by using the same random spread for both endpoints and this is examined in Section 9.2. In either case, truncation is necessary when mapping from the generated interval to the genotype. This truncation causes similar effects to those seen for Centre-Spread Representation. For example, Table 4 shows the frequency matrix for all possible intervals generated by the cover operator using a real encoding of length of $k = 3$.

This shows increased frequency of regions 2, 3 and 4 similar to that of Centre-Spread Representation. Furthermore, region 1 also shows increased mapping frequency as the interval width $q_i - p_i$ increases. Studies of such matrices for encoding

[3] (Wilson, 2001a) refers to the cover spread as r_0. For consistency we use s_0 for all representations.

Table 4: Phenotype frequency matrix for Ordered Bound Representation with $k = 3$ and standard cover with $s_0 = 1$

	q_i								
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
	0	8	15	21	26	30	33	35	120
	1		1	2	3	4	5	6	35
	2			1	2	3	4	5	33
p_i	**3**				1	2	3	4	30
	4					1	2	3	26
	5						1	2	21
	6							1	15
	7								8

lengths of $2 \leq k \leq 8$ all showed the same effects. From these we can derive the frequencies of expression of the region 2, 3 and 4 intervals:

$$\begin{aligned} f_{p_{\min},q_i} &= \frac{2^{3k} - 2^k}{3} \\ f_{p_i,q_{\max}} &= \frac{2^{3k} - 2^k}{3} \\ f_{p_{\min},q_{\max}} &= \frac{(2^k+1)^3 - 2^k - 1}{6} \end{aligned}$$

and the probabilities $P_{p_{\min},q_i}$, $P_{p_i,q_{\max}}$ and $P_{p_{\min},q_{\max}}$ for the cover operator with Ordered Bound Representation:

$$\begin{aligned} P_{p_{\min},q_i} &= \frac{2^{3k} - 2^k}{3(2^{3k})} \\ &= \frac{1}{3} - \frac{1}{3(2^{2k})} \\ \lim_{n\to\infty} &= \frac{1}{3} \end{aligned}$$

and similarly, $P_{p_i,q_{\max}} = \frac{1}{3}$

$$\begin{aligned} P_{p_{\min},q_{\max}} &= \frac{(2^k+1)^3 - 2^k - 1}{6(2^{3k})} \\ \lim_{n\to\infty} &= \frac{1}{6} \end{aligned}$$

thus $P_{p_i,q_i} = \frac{1}{6}$

So, even though Ordered Bound Representation has no intrinsic bias, the truncation necessary when using the cover operator introduces bias. This is shown in Figure 4 for $k = 5$.

Note that the amount of bias is determined by the setting of the s_0 parameter, as instances of covering where truncation is not necessary cannot introduce bias. The

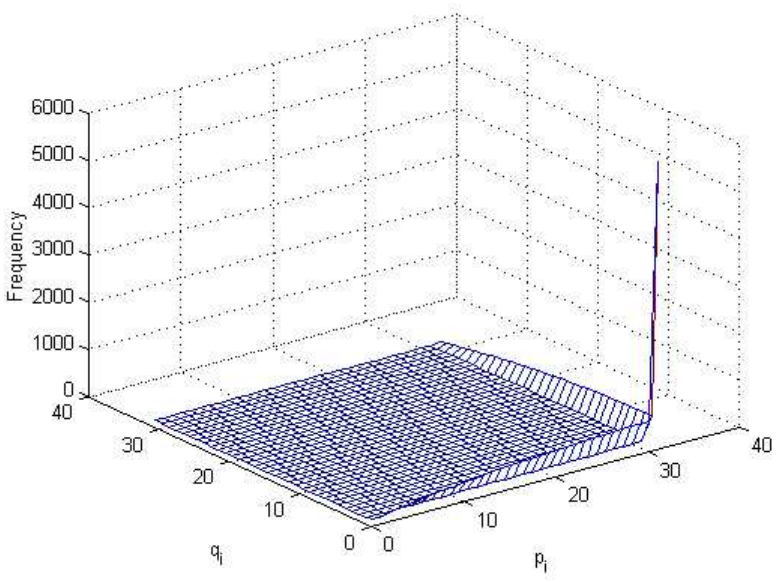

Figure 4: Phenotype frequency landscape for Ordered Bound Representation with $k = 5$ and standard cover with $s_0 = 1$

Table 5: Phenotype proportions for Ordered Bound Representation with $k = 8$ and crossover within an interval

Region	Form	Parent population	Offspring population
1	$[p_i, q_i)_p$	0.25	0.984
2	$[p_{\min}, q_i)_p$	0.25	0.008
3	$[p_i, q_{\max})_p$	0.25	0.008
4	$[p_{\min}, q_{\max})_p$	0.25	< 0.001

setting of $s_0 = 1$ provides a good bias for the real multiplexer problem since it generates classifiers with one or more intervals of the correct structural form (i.e., those in regions 2, 3 and 4) with a probability of

$$1 - \frac{1}{6^n}$$

For the 6-bit real multiplexer, this probability is 0.99998.

4.3.3 Crossover

Analysis of crossover with Ordered Bound Representation for an encoding of length $k = 8$ as described in Section 3.3.3 yields the results shown in Table 5.

Crossover under Ordered Bound Representation tends to produce offspring in region 1 at the expense of those in regions 2, 3 and 4. This affects the probability distribution of the population across regions and tends to remove offspring in regions 2, 3 and especially 4 (the maximally general interval) from the population. As such, crossover appears to provide the same bias as that of initialization for Ordered Bound Repre-

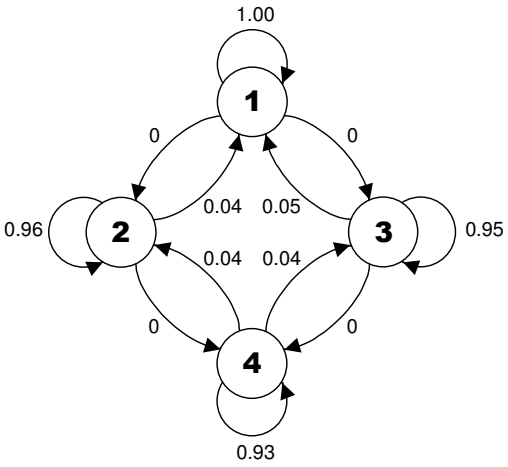

Figure 5: Region transition diagram for Ordered Bound Representation with $k = 8$ and standard mutation. Regions are represented by states and transition probabilities by arrows

sentation (Section 4.3.1). For simplicity, the analysis assumes a parent population with a uniform distribution across regions, which is not generally the case. But, although exact details will vary, the general trends seen here should still apply.

4.3.4 Mutation

Mutation for Ordered Bound Representation was studied in the same way as for Centre-Spread Representation (Section 3.3.4). The transition diagram for $k = 8$ is shown in Figure 5.

This displays similar characteristics to those of Centre-Spread Representation. Most mutations cause no change of region from parent to offspring, but simply refine the details of the interval within the region. When a transition does occur, the transition probabilities for Ordered Bound Representation essentially only allow transitions away from anchored (region 2, 3 and 4) intervals.

5 Unordered Bound Representation

5.1 Background

Ordered Bound Representation provides a one to one $g \rightarrow p$ mapping, but the $l_i \leq u_i$ ordering restriction is unnecessary. If this restriction is lifted, the phenotype can still be directly encoded using the endpoints of the interval, but without an ordering requirement. Thus, an interval $[p_i, q_i)_p$ may be encoded as the tuples (p_i, q_i) or (q_i, p_i) $\forall\ p_i \neq q_i$. There are thus exactly two equivalent genotypes for each phenotype except where $p_i = q_i$ when there is exactly one genotype for each phenotype. In other words, the $g \rightarrow p$ mapping is normally two to one, except for exact numbers, which have a one to one mapping. We do not expect the resulting small bias in favour of intervals over exact numbers to substantially affect the performance of a Learning Classifier System using Unordered Bound Representation. Indeed, the desire for an interval-based phenotype

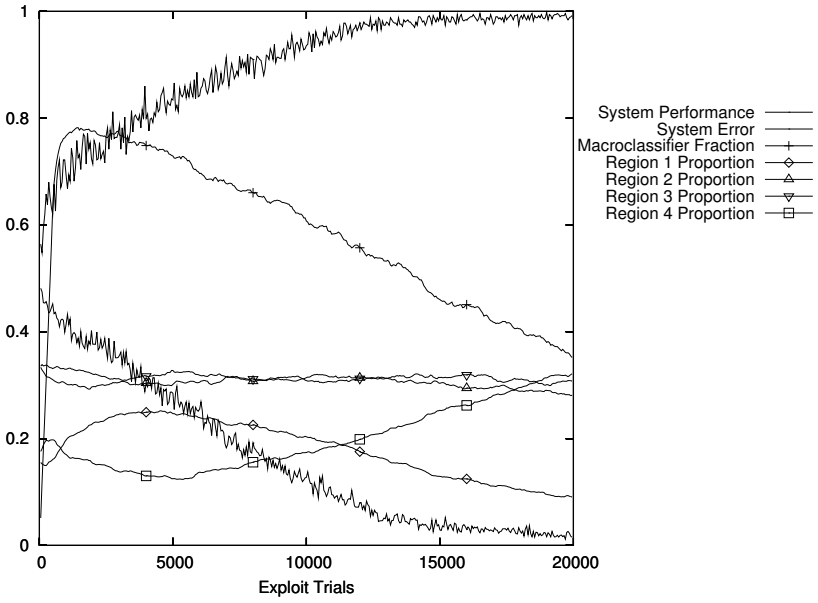

Figure 6: 6-bit real multiplexer with Unordered Bound Representation, standard cover with $s_0 = 1$, 2-point crossover and standard mutation

suggests that the solution to a problem using an interval-based representation is best expressed in the form of a vector of intervals, rather than simple inequalities, so any resulting slight performance differences compared to Ordered Bound Representation should, if anything, be advantageous. In any event, the bias induced by the Unordered Bound Representation's $g \rightarrow p$ mapping is negligible compared to the major disparities in phenotype expression frequency seen using Centre-Spread Representation.

The advantage of Unordered Bound Representation over Ordered Bound Representation is that it avoids the additional operator complexity associated with swapping the endpoints of an interval if the $l_i \leq u_i$ ordering restriction is violated. Although this may seem trivial, the presence of the ordering restriction constitutes a form of epistasis between the l_i and u_i alleles, as their values are mutually dependent. A resulting swap may generate great change in a particular locus when viewed before and after the operation that caused the swap to occur. This cannot occur using Unordered Bound Representation, since no ordering of endpoints exists for the interval predicate at the level of the genotype.

Figure 6 shows the results of using Unordered Bound Representation on the 6-bit real multiplexer problem.

5.2 Properties

Properties of Unordered Bound Representation are:

Property 1: Two to one genotype to phenotype mapping for intervals (but not exact numbers)

Property 2: Complete phenotype to genotype mapping.

Property 3: The genotype to phenotype mapping of exact numbers is one to one.

Property 4: The frequency of intervals of the form $[p_{\min}, q_i)_p$ and $[p_i, q_{\max})_p$ is constant for all p_i and q_i.

Property 5: The frequency of the $[p_{\min}, q_{\max})_p$ interval is the same as that of any other interval.

The maximally general interval $[p_{\min}, q_{\max})_p$ may be represented by two possible tuples in Unordered Bound Representation, so given a random genotype, an interval will be maximally general with probability

$$\frac{1}{2^{2k-1}}$$

Thus, Unordered Bound Representation, like Ordered Bound Representation, provides no implicit 'don't care' mechanism.

5.3 Operators

In this section we investigate Unordered Bound Representation and the operators adapted for it.

5.3.1 Initialization

The frequency of expression of intervals of the form $[p_{\min}, q_i)_p$, $[p_i, q_{\max})_p$ and $[p_{\min}, q_{\max})_p$ (regions 2, 3, and 4) can be calculated as

$$\begin{aligned} f_{p_{\min},q_i} &= 2^{k+1} - 3 \;\forall\; p_{\min} \leq q_i < q_{\max} \\ f_{p_i,q_{\max}} &= 2^{k+1} - 3 \;\forall\; p_{\min} < p_i \leq q_{\max} \\ f_{p_{\min},q_{\max}} &= 2 \end{aligned}$$

The number of possible $g \rightarrow p$ mappings is 2^{2k}, so the frequency of intervals of the form $[p_i, q_i)_p \;\forall\; p_i > p_{\min} \wedge q_i < q_{\max}$ (region 1) is given by

$$\begin{aligned} f_{p_i,q_i} &= 2^{2k} - 2(2^{k+1} - 3) - 2 \\ &= (2^k - 2)^2 \end{aligned}$$

The probability of expression of an interval of this form is

$$\begin{aligned} P_{p_i,q_i} &= \frac{(2^k - 2)^2}{2^{2k}} \\ &= \frac{2^{2k} - 2(2^{k+1}) + 4}{2^{2k}} \\ \lim_{n\to\infty} &= 1 \end{aligned}$$

The implication of this is that, like Ordered Bound Representation, initialization generates classifiers essentially exclusively in region 1.

5.3.2 Covering

The cover operator for Unordered Bound Representation is the same as that for Ordered Bound Representation, with the addition that, to avoid unnecessary bias, it encodes the endpoints of the generated interval in a random order. This does not affect its operation, so the results presented for Ordered Bound Representation in Section 4.3.2 also apply here.

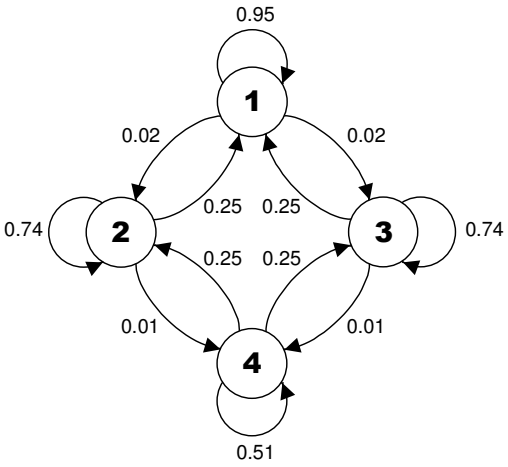

Figure 7: Region transition diagram for Unordered Bound Representation with $k = 8$ and standard mutation. Regions are represented by states and transition probabilities by arrows

5.3.3 Crossover

Analysis of crossover with Unordered Bound Representation for an encoding of length $k = 8$ as described in Section 3.3.3 yields the same results as those shown in Table 5. The comments made in Section 4.3.3 for crossover with Ordered Bound Representation therefore also apply to Unordered Bound Representation.

5.3.4 Mutation

Mutation for Unordered Bound Representation was studied in the same way as for Centre-Spread Representation (Section 3.3.4). The region transition diagram for $k = 8$ is shown in Figure 7.

This shows that the mutation operator for Unordered Bound Representation acts to provide a strong pressure away from region 4. For region 4 intervals, there is only a 0.51 probability of staying in region 4 upon mutation, with a transition to regions 2 or 3 equally likely. Similarly for a region 2 or 3 interval, a transition to region 1 is possible with a probability of 0.25.

6 The Real Multiplexer Revisited

6.1 Solving the Real Multiplexer

We have seen how Centre-Spread Representation with the operators and parameter settings being used make it especially suited to solving the real multiplexer problem. Wilson (2000) states, "notice how the system has 'sculpted' the predicates and is in effect finding the thresholds. Most predicates either show ranges between 0.0 and 0.5, 0.5 and 1.0 or are 'don't cares' ". We suggest that it is the combination of Centre-Spread Representation, the operators and their parameter settings that provides pressure for these effects. Initialization and/or covering generate classifiers containing intervals

of the correct structural form, $[p_{\min}, q_i)_p$, $[p_i, q_{\max})_p$ and $[p_{\min}, q_{\max})_p$(regions 2, 3, and 4). Crossover and mutation then refine these by discovering the correct thresholds, θ_i to solve the problem. Having one of the endpoints of an interval correct upon initial generation of a classifier allows a much simpler genetic search compared to having to discover both ends of an interval concurrently. In this way, the representation and/or operators relieve the other mechanisms of XCS from much of the burden of solving the real multiplexer problem because the solution to the problem happens to match the nature of the classifiers being generated. But, for an arbitrary problem, this may not be the case.

6.2 Sampling Bias

This hypothesis suggests that the time taken to solve the real multiplexer problem should be independent of the threshold, θ_i. We repeated the real multiplexer experiment in (Wilson, 2000) where $\theta_i = 0.75$ and, like Wilson, were unable to solve the problem in 20,000 exploit trials. However, we found that if XCS is allowed to run for 50,000 trials, it does solve the problem. In fact, XCS takes approximately 2.5 times longer to solve the real multiplexer with $\theta_i = 0.75$, than when $\theta_i = 0.5$, even if $\theta_i = 0.75$ does not alternate across values of i. Further experimentation revealed that this is because $[0, 0.75]_p$ intervals are sampled with three times the frequency than that of $[0.75, 1)_p$ intervals. If both intervals are sampled with equal frequency, then XCS solves the problem in the same number of trials as for when $\theta_i = 0.5$ (not shown). This is the explanation that Wilson suggests. Importantly, the difference in performance solely arises due to sampling bias and not from any representation or operator bias present.

6.3 Relationship to Integer Results

Even with a neutral representation, such as Ordered Bound Representation or Unordered Bound Representation, the present cover operator still generates classifiers containing intervals with the correct structural form with an increased frequency. The Random-Data2 and Random-Data9 test problems (Wilson, 2001a) exhibit the same characteristics as described for the real multiplexer; that is, solutions to the problem are of the form $[p_{\min}, q_i)_p$, $[p_i, q_{\max})_p$ and $[p_{\min}, q_{\max})_p$(regions 2, 3, and 4). It would appear from Wisconsin Breast Cancer results[4] that this problem also has these characteristics. Wilson asks why XCS solves the Random-Data9 problem within a factor of 10 of the simpler Random-Data2 problem when the input space is exponentially larger (10^9 versus 10^2). We hypothesize that the covering bias described plays a part in this anomaly by generating classifiers containing intervals of the correct structural form. It is not unreasonable then to assume that the additional effort for crossover and mutation to refine these is better than exponential. Further work is necessary to validate this hypothesis with the above test problems.

6.4 Interval Predicates and the Real Multiplexer

We suggest that the real multiplexer problem is a poor choice of test problem for Learning Classifier Systems operating with continuous-valued data and interval predicates, since its solutions all have one endpoint in common with the maximally general interval in the solution space. Because of this, it is not representative of the broader class of problem where solutions are not, in general, closely aligned to the representation of the 'don't care' state.

[4] Figure 5 in (Wilson, 2001a).

Figure 8: Rectangle centred in a 2-dimensional solution space. The decision surface is shaded

The benefits of interval predicates are that they are able to represent hyper-rectangular decision surfaces in solution space. These benefits only accrue if (i) the problem solution requires a hyper-rectangle, rather than a hyperplane decision surface or (ii) the form of the problem solution is not known *a priori*. The real multiplexer problem can be solved using a hyperplane decision surface since all of the hyper-rectangles needed for the solution are anchored at a boundary of the solution space. It does not strictly require the presence of interval predicates to represent the solution and consequently cannot adequately test the general operation and performance of representations that use interval predicates.

We argue that the real benefit of the use of interval predicates is their ability to represent arbitrary intervals in solution space. This provides a richer expressive power that cannot be achieved using hyperplane Decision Surfaces and potentially allows a broader class of problem to be solved. In many real-world problems, the form of the solution is unknown *a priori* and test problems for Learning Classifier Systems using interval predicates must be flexible enough to explore all aspects of their operation and performance. This is not the case with the real multiplexer problem.

7 Hyper-Rectangles

7.1 Full Environmental Map

XCS attempts to build a full environmental map of the problem in order to cover the solution space with classifiers. The map takes the form of the population of classifiers, with individual classifiers representing portions of the map.

Classifiers using an interval-based representation construct hyper-rectangular decision surfaces in solution space. For all of the problems discussed in Section 6, the decision surface can be represented by a hyperplane and so one of the faces of the hyper-rectangle is always at the boundary of the solution space. This face simply serves to specify the direction of the inequality otherwise represented by the hyperplane.

A hyper-rectangle can approximate more complex decision surfaces than a hyperplane. In this case the decision surface is closed and will have faces that are not at solution space boundaries. This would seem to be a disadvantage for representation and operator combinations that provide bias towards the solution space boundaries. However, since XCS builds a complete map of the solution space, for each classifier representing a closed decision surface, there are multiple classifiers representing the solution space outside the closed decision surface. For example, consider a rectangle centred in a 2-dimensional solution space (Figure 8).

There are at least four other rectangles outside this rectangle mapping the solution space. Each of these touches the bounds of the solution space and presumably gains benefit from any bias of the representation and operators towards solution space boundaries. In general, the balance of this benefit will depend on the shape of the decision surface.

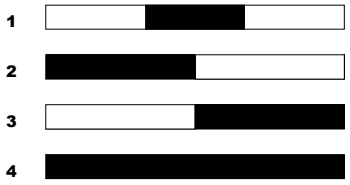

Figure 9: The four types of decision surface representing the solution interval (shaded) possible in a 1-dimensional solution space

7.2 1-Dimensional Solution Space

A single closed decision surface can be created in a 1-dimensional solution space by dividing the solution space into non-overlapping hyper-rectangles in four ways (Figure 9).

In case 1 three hyper-rectangles (i.e., interval predicates) must be constructed to cover the solution space. Two of these have their faces at the solution space boundary (the unshaded rectangles in the diagram). This is the general problem $\theta_l \leq x_i < \theta_u$, where θ_l and θ_u are the lower and upper bounds of the solution interval.

For cases 2 & 3 the solution space is covered by two hyper-rectangles, both of which have one of their faces at the solution space boundary. These are the cases for the real multiplexer and the other experiments discussed in Section 6.

Case 4 shows a hyper-rectangle covering all of one dimension of the solution space, with both faces at the boundary of the solution space. This represents the maximally general 'don't care' interval.

Notice that the four cases shown correspond to the four structural forms (regions) of interval predicate previously described.

Thus, for a single closed decision surface representing the solution interval, there are always more hyper-rectangles requiring faces at the solution space boundary than those that do not. This is because XCS 'fills in' the missing parts of the solution space when building its complete environmental map. Strictly, this closed decision surface is all that is required for a classifier in a traditional (non-accuracy based) classifier system. However, XCS also generates the other hyper-rectangles to complete the map.

Even if a dimension of the solution space is divided into multiple Decision Surfaces representing the solution interval, then excluding the maximally general interval (case 4 above), there will always be exactly two hyper-rectangles with faces at the solution space boundary. The number of hyper-rectangles without faces at solution space boundaries exceeds those with faces at solution space boundaries only when a dimension of the solution space is divided into a total of five or more hyper-rectangles. Thus, it would seem reasonable to assume that a bias towards hyper-rectangles with faces at solution space boundaries is an advantage when a 1-dimensional solution space is divided into less than five hyper-rectangles.

7.3 Multidimensional Solution Space

Of course, in a multidimensional solution space, the influence of the hyper-rectangle complexity of all dimensions must be taken into account. Assuming that each dimension, n, of the solution space is divided into the same number of hypercubes, n_d, the proportion of hypercubes with one or more faces at the solution space boundary is

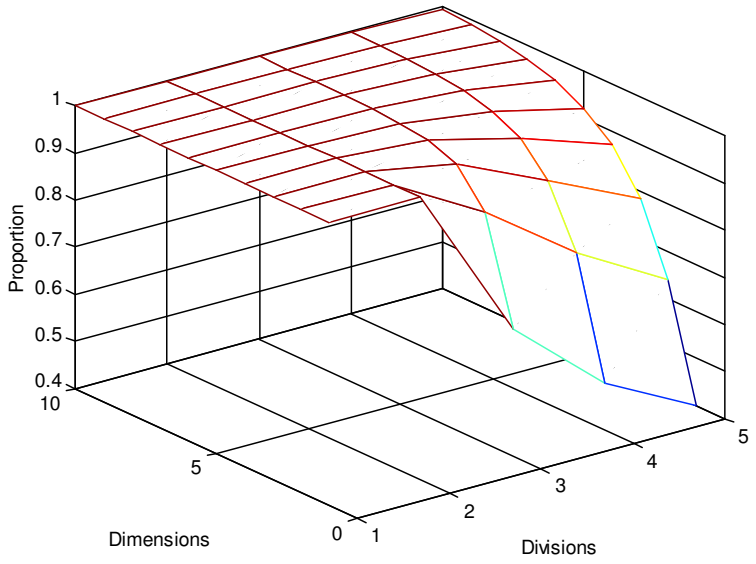

Figure 10: Proportion of hypercubes with one or more faces at the solution space boundary for problems of dimension n, with each dimension divided into n_d hypercubes

given by

$$\frac{n_d^n - (n_d - 2)^n}{n_d^n} \; \forall \; n_d \geq 2$$

This is plotted in Figure 10, which shows that the proportion of hypercubes with one or more faces at the solution space boundary depends primarily on the dimensionality of the problem and only secondarily on the number of hypercubes into which each dimension is divided. For almost all problems, this proportion is greater than 0.5, while for problems with several dimensions (i.e., most real-world problems) the number of hypercubes with no face at the solution space boundary becomes insignificantly small. As a result, these hypercubes are likely to have little influence on the performance of XCS when constructing its environmental map. Therefore, even problems where the solution is of the form $[p_i, q_i)_p \; \forall \; p_i > p_{\min} \wedge q_i < q_{\max}$ (region 1) should benefit from the representation and operator bias studied here, as the solution to the problem is dominated by the search for intervals in regions 2, 3 and 4 – precisely those for which bias exists.

8 The Checkerboard Problem

8.1 Description

To circumvent the limitations of the real multiplexer problem, we use a new abstract single-step test problem, the checkerboard problem. This problem divides up the n-dimensional solution space into equal sized hypercubes. Each hypercube is assigned a 'colour' black or white, with the colours alternating in all dimensions. For $n = 2$ the solution space takes on the appearance of a chess or checkers board. The problem diffi-

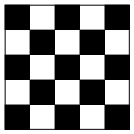

Figure 11: 2-dimensional checkerboard with $n_d = 5$

culty is controlled by both the dimensionality of the solution space, n and the number of divisions of each dimension of the solution space, n_d. To allow the colours to alternate in all dimensions, n_d must be an odd number. Figure 11 shows a 2-dimensional checkerboard with $n_d = 5$.

On each trial, the Learning Classifier System is presented with a vector of n random real numbers in the interval $[0, 1)_p$, representing a point in the solution space. The Learning Classifier System then attempts to assign an action, 0 or 1 depending on whether the point is contained in a black (0) or white (1) hypercube. The classifiers generated by the Learning Classifier System thus correspond directly to hypercubes in the solution space.

The solution to the checkerboard problem, as presented, requires no maximally general intervals due to the presence of alternating hypercubes. Although we do not use it here, a controlled number of maximally general intervals may be added to the checkerboard problem by making black entire hyper-rows and hyper-columns of the checkerboard. The number of hyper-rows and hyper-columns generalized in this way is controlled by a parameter, n_g with the maximally general intervals being allocated uniform randomly among dimensions and divisions of the problem.

The checkerboard problem is analogous to the test suite for ternary representations detailed in (Kovacs & Kerber, 2001).

8.2 Checkerboard with Initial Population

Figure 12 and Figure 13 show the performance of Centre-Spread Representation and Unordered Bound Representation on the checkerboard problem with $n = 3$ and $n_d = 3$. The solution to this problem consists of 27 hypercubes, so XCS needs 54 classifiers to construct a full map. In these experiments, an initial population of 2000 classifiers was used. Other settings are as for the real multiplexer experiments. We did not test the performance of Ordered Bound Representation due to its similarity to Unordered Bound Representation.

Here, the initial proportions of intervals in each region of the population match well the theoretically predicted values for initialization. In these experiments, proportions of intervals in each region are measured with reference to the macroclassifier population.

By observing the proportions of intervals in the population occupying the four regions, it is possible to gain some insight into the dynamics occurring as XCS solves the problem. Notice in Figure 12 (Centre-Spread Representation) how the proportion of each region diverges from the initial value of 0.25. Compare this to Figure 13 (Unordered Bound Representation), where the proportions converge from values of 0 (regions 2 and 3) and 1 (region 1).

The expected proportions of each region may be calculated for the checkerboard problem with $n = 3$ and $n_d = 3$ by counting the number of hypercubes at the corners, edges, faces and centre of the solution space. Each of these types of hypercube

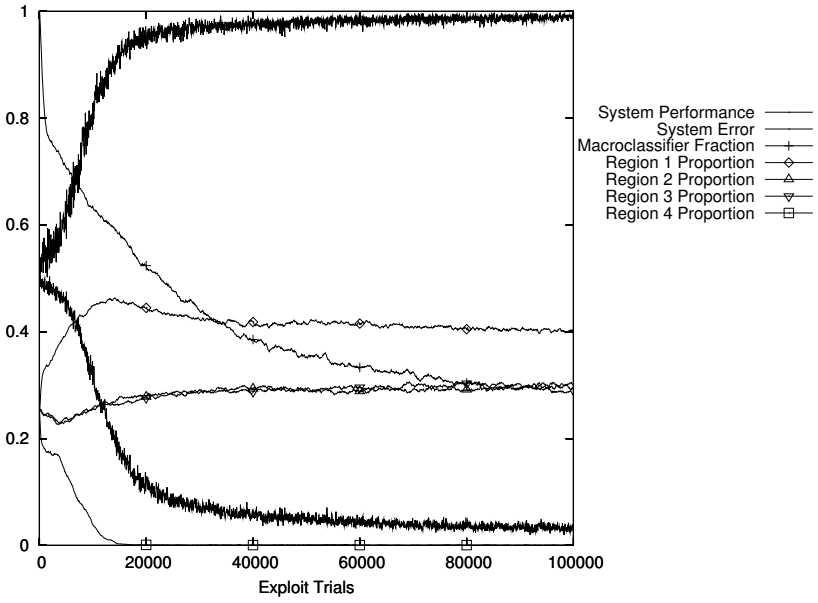

Figure 12: Checkerboard problem with Centre-Spread Representation, an initial population, standard cover with $s_0 = 1$, 2-point crossover 'within' and standard mutation

is represented by a specific combination of interval regions (for example, a hypercube at a corner of the solution space is represented by three region 2 or 3 intervals, while a hypercube at an edge is represented by two region 2 or 3 intervals and one region 1 interval). From this, we find that the expected proportion of each of region 1, 2 and 3 classifiers is $\frac{1}{3}$. It is clear from Figure 13 that although a solution to the problem appears to have been found, the proportion of region 1 classifiers is too high, whereas the proportion of region 2 and 3 classifiers is too low. This is because, apart from a low probability of mutation, the only pressure towards generalization is that provided by the environment via cover spread. With an environment that presents uniform random values, as classifiers become more general (i.e., have wider intervals), the probability of encountering an environmental input that is outside of an existing interval's range becomes lower and asymptotically approaches zero. Generalization pressure is thus variable and diminishes as XCS gets closer to solving the problem.

Although the dynamics of execution may differ, there is no great difference in System Performance and System Error between representations. We found that the presence of an initial population tended to mask the differences between representations. For this reason, we now focus on comparing representations using experiments without an initial population.

8.3 Checkerboard with no Initial Population

Figure 14 and Figure 15 show the same experiments with no initial population. Again, these results show initial proportions of intervals in each region close to the predicted values for the cover operator, given the small sample size due to the empty initial population.

It is immediately apparent that XCS with Centre-Spread Representation makes no

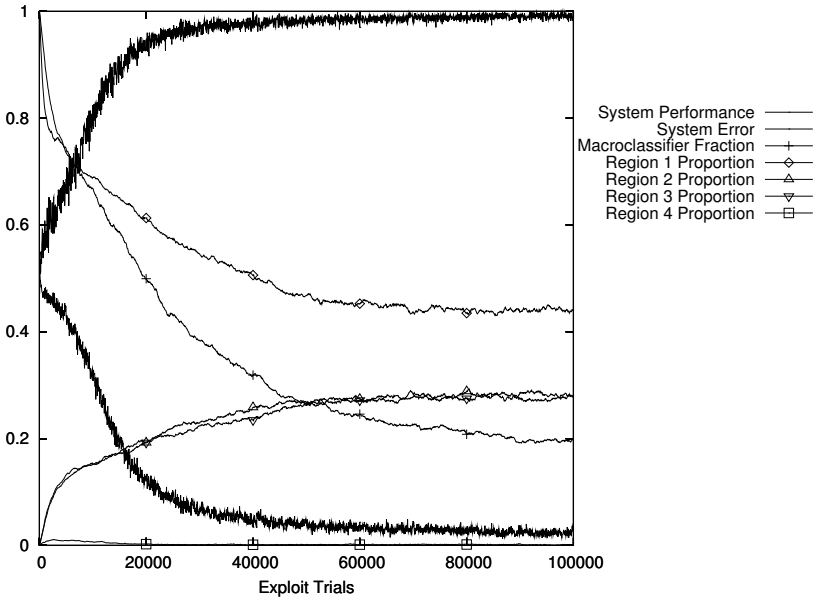

Figure 13: Checkerboard problem with Unordered Bound Representation, an initial population, standard cover with $s_0 = 1$, 2-point crossover 'within' and standard mutation

inroads towards solving the problem, whereas XCS with Unordered Bound Representation comes much closer. This is due to the abnormally high number of region 4 intervals and low number of region 1 intervals in the population with Centre-Spread Representation. For Centre-Spread Representation, covering is used only during the first 50 trials, during which time the number of region 4 (maximally general) intervals rises in the population. For the remaining trials the region 4 intervals in the population cover all environmental inputs and covering is unnecessary. These intervals take over the population and stall the search. In contrast, the search using Unordered Bound Representation makes progress from the start, correctly promoting region 1 intervals at the expense of those in region 4. In addition, the proportion of region 2 and 3 intervals after 100,000 exploit trials (0.27) is similar to the expected proportion of 0.33, suggesting that many of the cubes at the boundaries of the solution space have been identified. As suggested in Section 7.3, it appears that the bias of the Unordered Bound Representation operators and parameter settings better match the type of intervals needed to solve the problem than those of Centre-Spread Representation. In fact, covering generates region 2 and 3 intervals with a probability of $\frac{1}{3}$, which is exactly the right proportion required by the solution to the problem.

9 Comparing Representations and Operators

9.1 Background

We have seen that the choice of representation can make a large performance difference even when using the same system parameters. This difference can only arise from the action of the operators working on the representation. In order to isolate the reasons for any performance differences, we must systematically constrain operators to behave

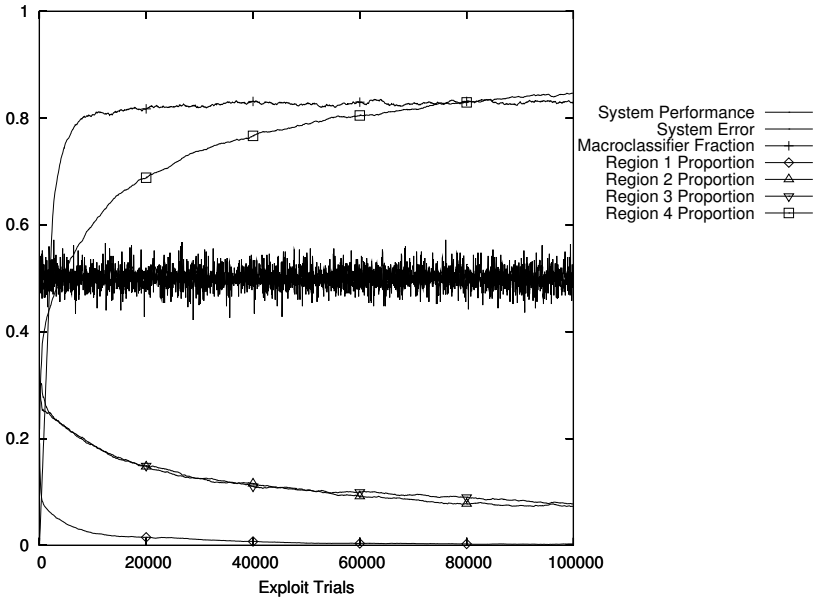

Figure 14: Checkerboard problem with Centre-Spread Representation, standard cover with $s_0 = 1$, 2-point crossover 'within' and standard mutation

identically for both representations.

Without an initial population, the operators responsible for any performance differences are covering, crossover and mutation. GA subsumption is performed at the level of the phenotype, so no performance differences can arise from this operator.

For Unordered Bound Representation, truncation during covering and mutation occurs on the alleles representing the lower and upper bound of the interval. This means that an interval in Unordered Bound Representation is limited to $[p_{\min}, q_{\max})_p$. When Centre-Spread Representation is used, it is the centre and spread alleles that are truncated during covering and mutation. Therefore, for Centre-Spread Representation, intervals in the underlying population are in the range $[2p_{\min} - q_{\max}, 2q_{\max} - p_{\min})_p$ and further truncation must be applied upon expression to limit these intervals to $[p_{\min}, q_{\max})_p$. The cover, crossover and mutation operators all work at a genotypic level, so in the case of Centre-Spread Representation, it is possible for intervals to be maintained in the population that are outside the range of the phenotype, but which are available for crossover and mutation to manipulate, and from which 'useful' intervals within range may subsequently emerge. This feature is not available with Unordered Bound Representation, where all intervals in the population are restricted to the range of the phenotype.

To design operators with identical characteristics for both representations, we need to limit the range of intervals in the population to that of the solution space. We thus refer to these as *restricted* operators. To test the restricted operators and verify that XCS behaves identically when using them, we ran experiments using restricted cover, no crossover and restricted mutation with both representations on the 6-bit real multiplexer and $n = 3$, $n_d = 3$ checkerboard problem. These showed the same results and population dynamics for both representations (not shown).

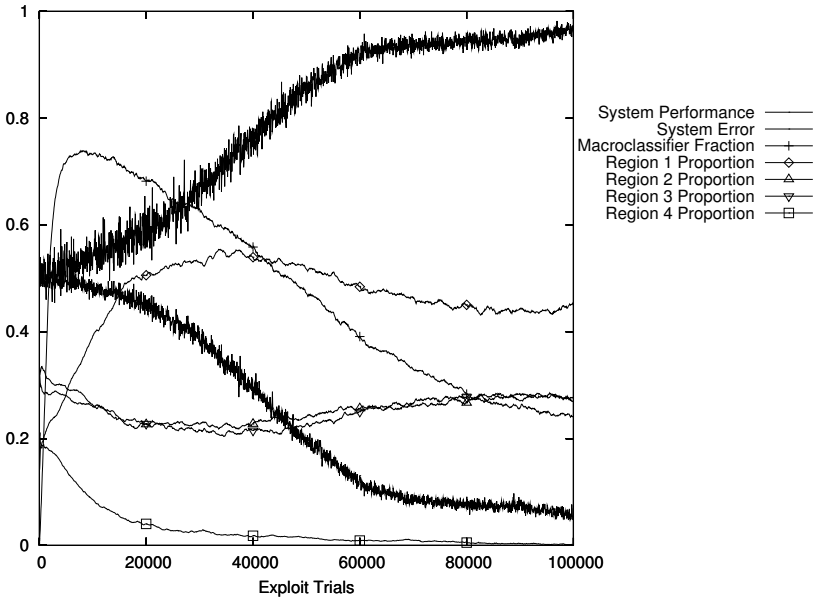

Figure 15: Checkerboard problem with Unordered Bound Representation, standard cover with $s_0 = 1$, 2-point crossover 'within' and standard mutation

To compare the effects of different operator choices, we performed extensive experimentation using the Centre-Spread Representation and Unordered Bound Representation operators and variants. These are listed in Table 6 and described in more detail in the following sections. Space precludes detailed examination of every combination of representation, operator and problem, so we focus instead only on general trends and results of particular interest.

9.2 Cover

We compared three variants of cover operator. Standard cover is the cover operator already described. This differs between Centre-Spread Representation and Unordered Bound Representation in two ways:

1. The Centre-Spread Representation cover operator is symmetric, since by definition, the spread must be equal on both sides of the centre. The Unordered Bound Representation cover operator, as presented, is asymmetric.

2. The Centre-Spread Representation cover operator generates intervals in the range $[2p_{\min} - q_{\max}, 2q_{\max} - p_{\min})_p$ while the Unordered Bound Representation cover operator generates intervals in the range $[p_{\min}, q_{\max})_p$.

Restricted cover is symmetric and generates intervals in the range $[p_{\min}, q_{\max})_p$ for both representations. The properties of restricted cover are the same as those described in Section 3.3.2. In the case of Unordered Bound Representation, the only change to the covering algorithm is to apply the same random spread to both sides of the environmental variable being covered. The algorithm for the Centre-Spread Representation restricted cover operator is more complex:

$$s_i = U[0, s_0)$$

Table 6: Operators used during comparison of representations and operator variants

Operator	Variant	Characteristics
Cover	Standard	Symmetric (CSR), Asymmetric (UBR)
	Restricted	Symmetric
	Unbiased	Symmetric
Crossover	Standard 1-point	Between predicates
	Standard 2-point	Between predicates
	Standard Uniform	Between predicates
	Restricted 1-point	Within predicates
	Restricted 2-point	Within predicates
	Restricted Uniform	Within predicates
Mutation	Standard	
	Restricted	

$$
\begin{aligned}
l &= EncodeandTruncate(x_i - s_i) \\
u &= EncodeandTruncate(x_i + s_i) \\
s &= \frac{(u-l)}{2} + (u-l) \bmod 2 \\
c &= l + s
\end{aligned}
$$

The algorithm generates an interval as a lower and upper bound so that truncation occurs as for Unordered Bound Representation. It then converts the encoded interval back to an encoded centre and spread as needed for Centre-Spread Representation. The spread is incremented by one if it is an odd number to ensure that region 4 intervals are generated in the correct proportion. This is necessary because using a one of m binary encoding, the range of the maximally general interval $[p_{\min}, q_{\max})_p$ is always odd. It cannot be represented in Centre-Spread Representation without truncation, as only even ranges can be represented.

Unbiased cover is simply a variant of standard cover with a symmetric spread that is limited to $U[0, \min(x_i - p_{\min}, q_{\max} - x_i))_p$. This avoids the need for truncation, as the spread is limited to the bounds of the solution space.

We found performance differences between standard Unordered Bound Representation (asymmetric) cover and restricted (symmetric) cover with certain combinations of problem, operators and parameter settings. It is possible that such variation in performance arises simply because of the differing nature of the bias of the two types of cover, as seen in Section 3.3.2, Section 5.3.2 and below. Alternatively, it could be related to a bias caused by the fact that asymmetric cover chooses a different random spread for each side of the environmental state, whereas symmetric cover produces an interval that is (excepting truncation) centred on the environmental input. Further work is necessary to understand these performance differences in more detail.

We also examined the effect of variations of the cover spread parameter, s_0. We used a value of $s_0 = 0.5$ for these experiments. This value allows all possible intervals to be generated with the exception of the maximally general interval, but results in a minimal amount of truncation. Table 7 and Table 8 show the phenotype frequency matrices for symmetric and asymmetric cover with $s_0 = 0.5$. It can be seen that although there are no region 4 intervals generated, for both operators there is still an increased probability of region 2 and 3 intervals. Moreover, the most frequent many

Table 7: Phenotype frequency matrix for symmetric cover (Centre-Spread Representation and Unordered Bound Representation) with $k = 3$ and $s_0 = 0.5$

		q_i							
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
p_i	**0**	1	1	2	2	2	1	1	0
	1		1	0	1	0	1	0	1
	2			1	0	1	0	1	1
	3				1	0	1	0	2
	4					1	0	1	2
	5						1	0	2
	6							1	1
	7								1

Table 8: Phenotype frequency matrix for asymmetric cover (Unordered Bound Representation) with $k = 3$ and $s_0 = 0.5$

		q_i							
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
p_i	**0**	4	7	9	10	6	3	1	0
	1		1	2	3	4	3	2	1
	2			1	2	3	4	3	3
	3				1	2	3	4	6
	4					1	2	3	10
	5						1	2	9
	6							1	7
	7								4

to one $g \rightarrow p$ mappings in region 2 and 3 occur around the median values of p_i and q_i with the frequencies ramping up to these values from the solution bounds. This means that covering is more likely to generate region 2 and 3 intervals with ranges around the median than those with very large or small ranges. In addition, the asymmetric cover operator shows a similar effect for region 1 intervals, which does not occur with the symmetric cover operator.

For both representations, the smaller cover spread was an advantage for the checkerboard problem, but produced poorer performance on the real multiplexer problem. This difference arises because of the need for maximally general intervals in the real multiplexer problem that is not present in the checkerboard problem. If the cover operator is able to generate intervals in region 4, this aids XCS in solving the real multiplexer problem. In contrast it is a handicap for the checkerboard problem, where no region 4 intervals are necessary to solve the problem.

The performance difference obtained by simply altering the cover spread parameter can be quite spectacular. Figure 16 shows the performance of XCS on the checkerboard problem with Centre-Spread Representation and $s_0 = 0.5$. A comparison of these results with those of Figure 14 reveals a major difference in performance, yet the only parameter change was to alter the cover spread from 1 to 0.5. The two sets of results show very different dynamics with respect to the evolution of the proportions of inter-

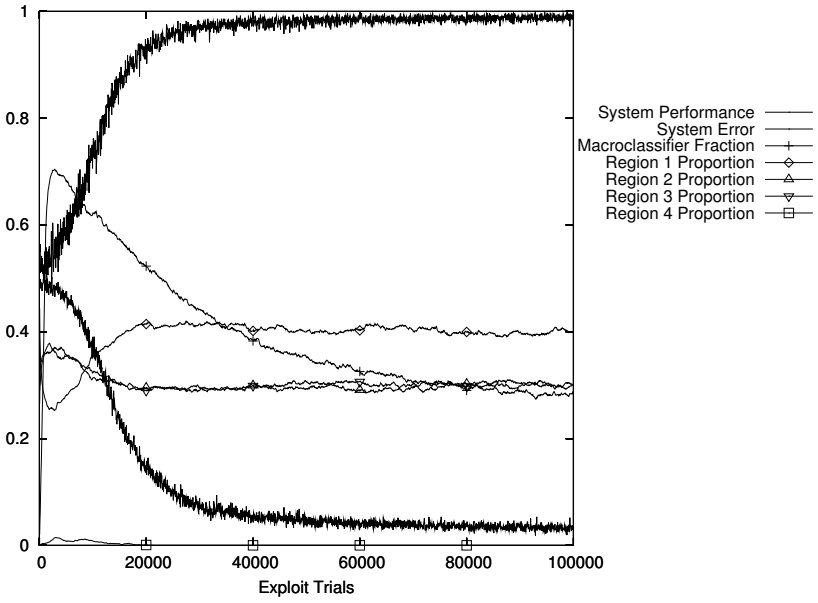

Figure 16: Checkerboard problem with Centre-Spread Representation, standard cover with $s_0 = 0.5$, 2-point crossover 'within' and standard mutation

vals of each region in the population. Although the performance differences between different values of cover spread are not always so great, the value of the cover spread does have a significant effect on system dynamics and, ultimately, on performance. For reference and comparison with Figure 15, Figure 17 shows the results for Unordered Bound Representation with $s_0 = 0.5$.

Unbiased cover showed similar effects with both representations. Whilst its performance on the checkerboard problem (Figure 18) was better than covering with $s_0 = 1$, it proved totally unsuitable for the real multiplexer problem (Figure 19). In Figure 18 it is possible to see how the proportion of region 1 intervals starts at 100% and then decreases as region 2 and 3 intervals are discovered. This happens only very slowly for the real multiplexer. Here, the proportion of region 2, 3 and 4 intervals needed to solve the problem is very low, even after 20,000 runs, when the problem would have been solved with a biased cover operator (Figure 1).

9.3 Crossover

As crossover is so tightly coupled with the representation, it is difficult to provide a restricted crossover operator that behaves identically for both Centre-Spread Representation and Unordered Bound Representation. However, if crossover operates only between predicates, it manipulates entire intervals and the underlying representation should be irrelevant. In this case, no performance difference is to be expected between representations. We refer to this as crossover *between* predicates. The standard crossover operators for Centre-Spread Representation and Unordered Bound Representation work *within* predicates, where the crossover point may be between any two alleles. As well as minimizing any differences due to representation, crossover between predicates allows us to see the benefits or otherwise compared to crossover within pred-

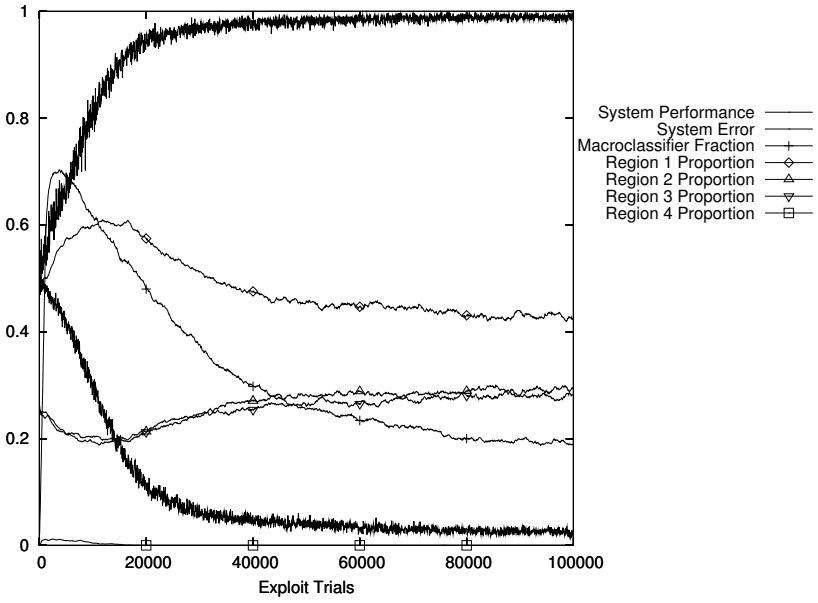

Figure 17: Checkerboard problem with Unordered Bound Representation, standard cover with $s_0 = 0.5$, 2-point crossover 'within' and standard mutation

icates.

We experimented with 1-point, 2-point and uniform crossover operators, both within and between predicates. These experiments were performed with restricted cover and restricted mutation to minimize differences between representations due to cover and mutation. Experiments were performed with both $s_0 = 1$ and $s_0 = 0.5$.

In general, we found little to choose between Centre-Spread Representation and Unordered Bound Representation, except on the checkerboard problem with $s_0 = 1$, where Unordered Bound Representation crossover within predicates produced consistently better results than Centre-Spread Representation crossover within predicates. We attribute this to the nature of the intervals produced by covering and the bias of crossover within intervals. With $s_0 = 1$, a relatively high proportion of region 4 intervals are introduced into the population. Crossover within predicates for Centre-Spread Representation does not materially affect this proportion (Section 3.3.3) and the proportion of region 4 intervals remains high. In contrast, the bias of crossover within predicates for Unordered Bound Representation (Section 5.3.3) reduces the proportion of region 4 intervals in the population to a small amount so that classifiers with maximally general intervals are unable to dominate action sets. Inspection of the results shows that the proportion of region 4 intervals is higher for Centre-Spread Representation (Figure 20) than that of Unordered Bound Representation (Figure 21) and that this is at the expense of the proportion of region 1, 2 and 3, which are needed to solve the problem.

We found that crossover between predicates tended to produce better results than crossover within predicates for the real multiplexer problem, but that the converse was true for the checkerboard problem (not shown). These results occurred for both Centre-Spread Representation and Unordered Bound Representation with settings of $s_0 = 0.5$

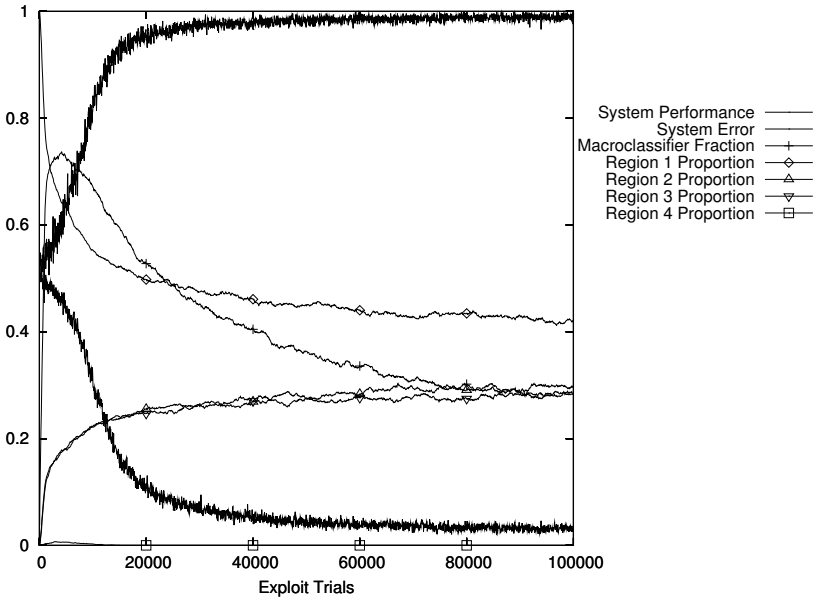

Figure 18: Checkerboard problem with Centre-Spread Representation, unbiased cover, 2-point crossover 'within' and standard mutation

and $s_0 = 1$. In all cases examined, performance correlated with the ability of the operators to generate or remove from the population region 1 and 4 intervals as needed by the problem. Proportions of region 2 and 3 intervals appear to be less critical to performance. It is possible that the recombination of centres and spreads is disruptive, as centre and spread alleles are mutually dependent, and further work is necessary to understand these results more fully.

9.4 Mutation

We used two types of mutation for the experiments. The first was the standard mutation operator already described. Although this is essentially the same algorithm for both representations, the alleles undergoing mutation differ for the two representations and the details of truncation differ between representations:

1. Mutation for Centre-Spread Representation creates a shift of the centre or a change in the size of the spread. Mutation for Unordered Bound Representation changes the value of the lower or upper bound.

2. The Centre-Spread Representation mutation operator generates intervals in the range $[2p_{\min} - q_{\max}, 2q_{\max} - p_{\min})_p$ while the Unordered Bound Representation mutation operator generates intervals in the range $[p_{\min}, q_{\max})_p$. This difference is apparent when the neutrality of the Centre-Spread Representation mutation operator with respect to region (Figure 3) is compared with the bias of the Unordered Bound Representation mutation operator (Figure 7).

To allow comparison between representations, we also implemented a restricted mutation operator. This mutates the effective centre or spread as per Centre-Spread Representation, but limits the resulting lower and upper bounds as per Unordered

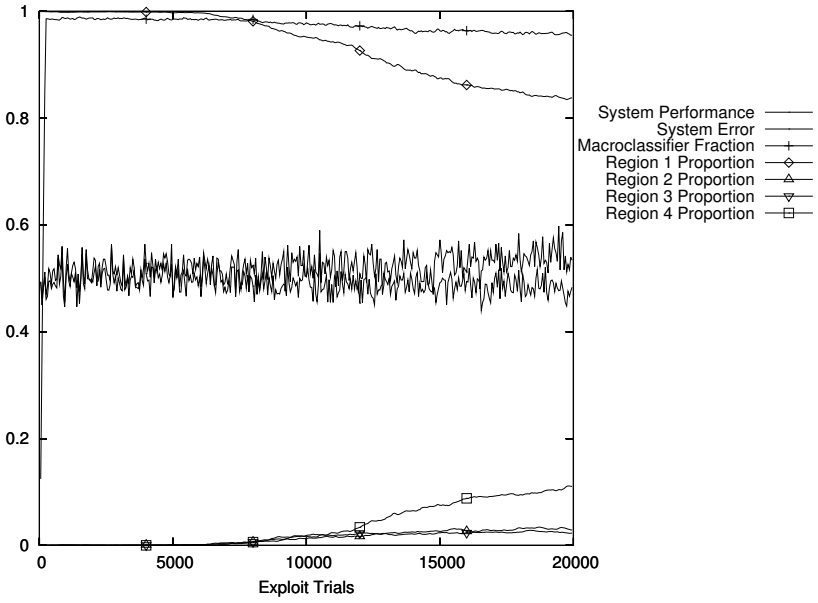

Figure 19: 6-bit real multiplexer with Centre-Spread Representation, unbiased cover, 2-point crossover 'within' and standard mutation

Bound Representation. For Unordered Bound Representation, this means that both alleles in an interval predicate are altered upon each mutation. Restricted mutation was used only to allow meaningful comparisons between variants of cover and crossover and was not intended for comparison with the standard mutation operator.

We compared the standard mutation operators for Centre-Spread Representation and Unordered Bound Representation. In these experiments, mutation operated in conjunction with the restricted cover and restricted crossover operators. However, we found no evidence suggesting that one mutation operator was superior to the other.

10 Conclusions

We showed that the Centre-Spread representation has a many to one $g \rightarrow p$ mapping that affects the proportions of intervals in the population. As a result, operators typically used with this representation provide bias in the intervals they generate. This bias is caused by the need to truncate both the interval itself (during gene expression) and the alleles representing the interval (within operators) to allow only legal ranges to be produced. If the solution space was unbounded, truncation would be unnecessary and no such bias would exist. We have not yet experimented with unbounded solution spaces.

Ordered Bound Representation has a one to one $g \rightarrow p$ mapping, but the need for truncation in its operators still causes bias. The ordering requirement within tuples with this representation motivated us to introduce a new representation, Unordered Bound Representation, which obviates problems caused by the ordering requirement, yet retains all the desirable features of Ordered Bound Representation.

We hypothesized that such representational and operator bias aids the solution of the real multiplexer problem because the intervals favoured by the bias correspond

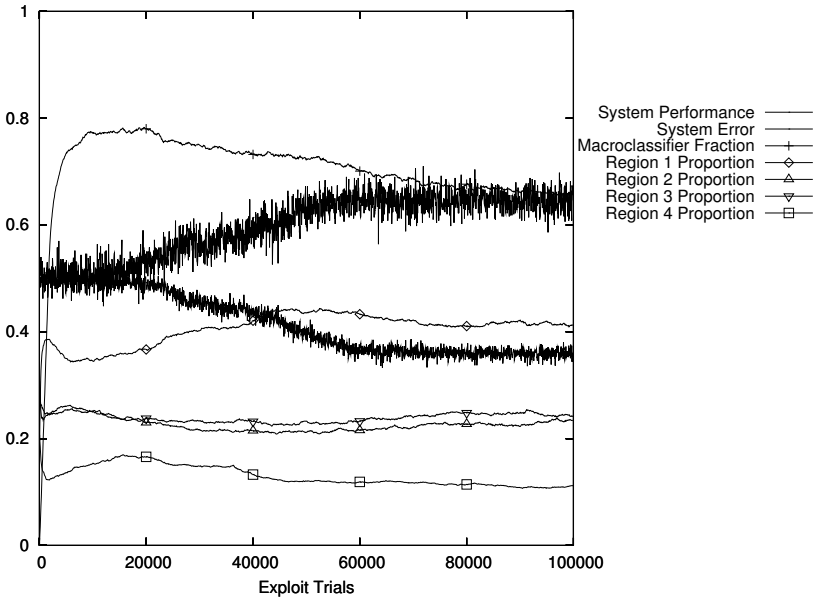

Figure 20: Checkerboard problem with Centre-Spread Representation, restricted cover with $s_0 = 1$, 2-point crossover 'within' and restricted mutation

closely to those needed for the solution to the real multiplexer problem. Consequently, we introduced a new test problem for continuous-valued domains, the checkerboard problem, which has a solution that is not closely correlated with the biased intervals and which matches more closely that of real-world problems. The checkerboard problem typically showed performance differences between operators and representations better than the real multiplexer. We will be experimenting with adding maximally general intervals to the checkerboard problem in the near future.

Testing with two representations and different variants of the standard cover operator showed that the type and amount of bias introduced by the representation and operators used does affect the performance of XCS. In particular, the spread parameter of the cover operator can make a huge difference in performance, because this parameter acts as a control over the distribution of intervals introduced into the population. One idea that we have not yet tried is to augment the cover and mutation operators with an explicit mechanism to introduce maximally general intervals into the population in a probabilistic manner similar to that used by a ternary representation. This may allow more control over this aspect of the distribution of intervals in the population.

In general, our experimental results support the hypothesis that representation and operators aid the performance of XCS by generating intervals that are useful to solve the problem. As a result, representation and operators must be matched to the problem at hand in order to achieve the best results. If this does not occur, XCS may not be able to solve the problem. These results have similarities with those reported in (Butz et al, 2002) for XCS with a discrete representation with respect to the impact of generalization upon system performance.

We also found that sampling bias affects system performance. It is possible to solve the real multiplexer problem when $\theta_i = 0.75$ in the same number of trials as for $\theta_i =$

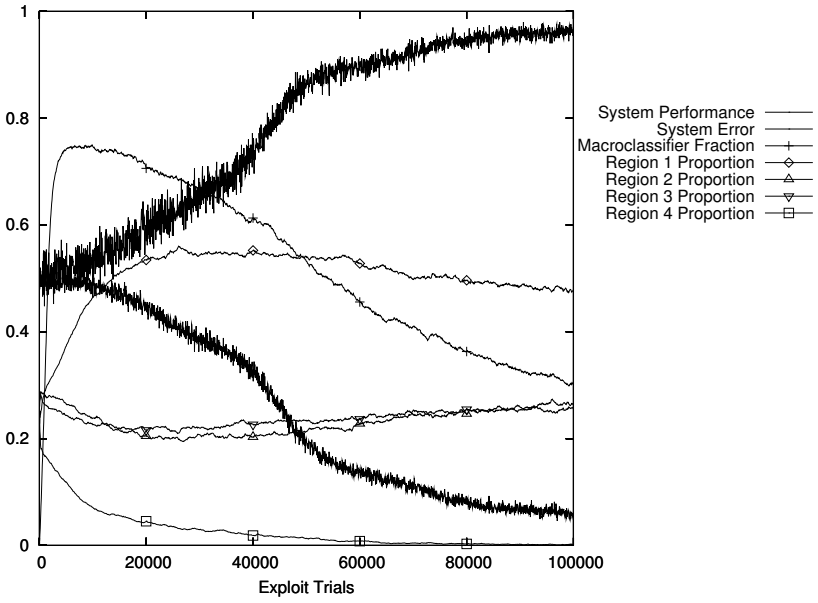

Figure 21: Checkerboard problem with Unordered Bound Representation, restricted cover with $s_0 = 1$, 2-point crossover 'within' and restricted mutation

0.5 by sampling solution intervals with equal frequency. Bias caused by unbalanced training examples is a well-known problem in machine learning (Breiman *et al*, 1984).

Although we specifically examined XCS in this paper, many of the results and conclusions also apply to other Learning Classifier System architectures using the representations studied. In particular, all of the analysis of representation and operator bias is applicable to other architectures. However, the outcome of these biases with architectures that do not build a complete environmental map may not correspond to those seen here for XCS. This is because the arguments presented in Section 7 relating to the proportion of hyper-rectangles at the solution boundary do not apply unless a complete map is built. In architectures where this does not occur, the relative desirability of intervals will differ from that seen for XCS and a complete map.

References

Ahluwalia, M. & Bull, L. (1999). A Genetic Programming-based classifier system. In W. Banzhaf et al. (eds.), *GECCO-99: Proceedings of the Genetic and Evolutionary Computation Conference. San Francisco, CA: Morgan Kaufmann*, pages 11–18.

Bonarini, A. (2000). An introduction to Learning Fuzzy Classifier Systems. In P. L. Lanzi, W. Stolzmann and S. W. Wilson (eds.), *Learning Classifier Systems. From Foundations to Applications, LNAI-1813*, Berlin: Springer, pages 83–104.

Breiman, L., Friedman, J. H., Olshen, R. A., Stone, C. J. (1984). *Classification and Regression Trees*. New York, NY: Chapman & Hall.

Bull, L. & O'Hara, T. (2002). Accuracy-based neuro and neuro-fuzzy classifier systems. In W. B. Langdon et al. (eds.), *GECCO-2002: Proceedings of the Genetic and Evolutionary Computation Conference*. San Francisco, CA: Morgan Kaufmann, pages 905–911.

Bull, L., Wyatt, D. & Parmee, I. (2002). Initial Modifications to XCS for use in Interactive Evolutionary Design. In J. J. Merelo et al. (eds.), *Parallel Problem Solving from Nature - PPSN VII*, Berlin: Springer, pages 568–577.

Butz, M. V. & Wilson, S. W. (2001). An algorithmic description of XCS. In P. L. Lanzi, W. Stolzmann and S. W. Wilson (eds.), *Advances in Learning Classifier Systems. Proceedings of the Third International Workshop (IWLCS-2000), LNAI-1996*, Berlin: Springer, pages 253–272.

Butz, M. V., Kovacs, T., Lanzi, P. L. & Wilson, S. W. (2002). Theory of Generalization and Learning in XCS. Technical Report No. 2002011, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign, May, 2002.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems.* The University of Michigan Press, Ann Arbor, MI. Republished by MIT Press, 1992.

Holland, J. H. (1986). Escaping brittleness: the possibilities of general-purpose learning algorithms applied to parallel rule-based systems. In R. S. Michalski, J. G. Carbonell & T. M. Mitchell (eds.), *Machine Learning, an Artificial Intelligence Approach. Volume II*. Los Altos, California: Morgan Kauffmann, pages 593–623.

Kovacs, T. (1996). *Evolving optimal populations with XCS classifier systems*. Master's thesis, School of Computer Science, University of Birmingham, U. K. Also Technical Report CSR-96-17 and CSRP-96-17, School of Computer Science, University of Birmingham, U. K.

Kovacs, T. & Kerber, M. (2001). What makes a problem hard for XCS? In P. L. Lanzi, W. Stolzmann and S. W. Wilson (eds.), *Advances in Learning Classifier Systems. Proceedings of the Third International Workshop (IWLCS-2000), LNAI-1996*, Berlin: Springer, pages 80–99.

Lanzi, P. L. (1999). Extending the representation of classifier conditions, part II: from messy coding to s-expressions. In W. Banzhaf et al. (eds.), *GECCO-99: Proceedings of the Genetic and Evolutionary Computation Conference*. San Francisco, CA: Morgan Kaufmann, pages 345–352.

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

Wilson, S. W. (2000). Get real! XCS with continuous-valued inputs. In P. L. Lanzi, W. Stolzmann and S. W. Wilson (eds.), *Learning Classifier Systems. From Foundations to Applications, LNAI-1813*, Berlin: Springer, pages 209–219.

Wilson, S. W. (2001a). Mining oblique data with XCS. In P. L. Lanzi, W. Stolzmann and S. W. Wilson (eds.), *Advances in Learning Classifier Systems. Proceedings of the Third International Workshop (IWLCS-2000), LNAI-1996*, Berlin: Springer, pages 158–174.

Wilson, S. W. (2001b). Function approximation with a classifier system. In L. E. Spector et al (eds.), *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO-2001*, San Francisco, CA: Morgan Kaufmann, pages 974–981 .