Representation and Hidden Bias II: Eliminating Defining Length Bias in Genetic Search via Shuffle Crossover

Richard A. Caruana (raca@philabs.philips.com)
Larry J. Eshelman (lje@philabs.philips.com)
J. David Schaffer (ds1@philabs.philips.com)

Philips Laboratories, North American Philips Corporation, 345 Scarborough Road, Briarcliff Manor, New York 10510

Abstract

The traditional crossover operator used in genetic search exhibits a position-dependent bias called the *dcfining-length bias.* We show how this bias results in *hidden biases* that are difficult to anticipate and compensate for. We introduce a new crossover operator, *shuffle crossover,* that eliminates the position dependent bias of the traditional crossover operator by shuffling the representation prior to applying crossover. We also present experimental results that show that shuffle crossover outperforms traditional crossover on a suite of five function optimization problems.

1. Introduction

The selected knowledge representation serves as a strong learning bias: it defines the concept space to be searched by the learning algorithm [Utgoff 1983]. Additional biases are often employed to direct search towards favored regions in the space. We show that unforeseen interactions between the representation and search mechanism can result in *hidden biases* that hinder search. We present an example where position dependent search biases interact with representations of identical expressiveness to yield different search behavior. The search mechanism we investigate is the crossover operator of the genetic algorithm developed by Holland [Holland 1975].

A genetic algorithm (GA) is a powerful general-purpose search method based on mechanisms abstracted from population genetics. The GA maintains a set of trial solutions called a population. It operates in cycles called generations that produce successive populations by survival-of-the-fittest selection followed by genetic recombination. Trial solutions are represented as strings called chromosomes that are usually coded with a binary character set.

The two most commonly employed genetic search operators are *crossover* and *mutation.* Crossover produces offspring (new trial solutions) by recombining the information from two parents in the manner illustrated in Figure 1. It is the major exploratory mechanism of the GA. Mutation prevents convergence of the population by flipping a small number of randomly selected bits to continuously introduce variation.

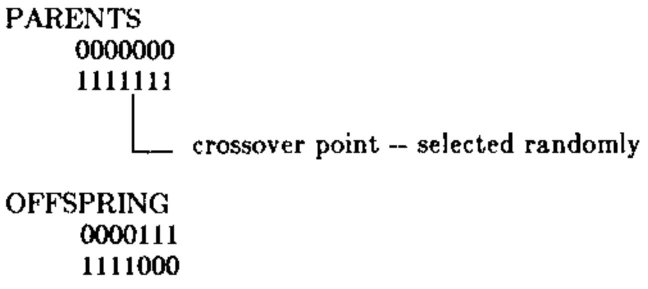

Figure 1. The crossover recombination operator

Since its introduction, the GA has been the focus of research attempting to better understand its power, overcome its weaknesses, and apply it to problems calling for efficient and flexible search [Grefenstette 1985, Grefenstette 1987, Schaffer 1989]. Much of this research has centered on representational issues; the effectiveness of the GA depends heavily on the chosen representation. In this paper we discuss one such issue, the positional dependence of string representations when traditional crossover is used, and present a new crossover operator, *shuffle crossover,* that eliminates it. We present experimental results that indicate shuffle crossover is superior to traditional crossover. We believe that the position dependent bias of the traditional crossover operator is an instance of a general knowledge representation problem: unanticipated search bias emerging from an interaction between the representation and search heuristics.

2. Shuffle Crossover

Traditional one-point crossover operates by (1) randomly pairing two individuals, (2) selecting a random position along the string, and (3) then exchanging the segments to the right of this position (see Figure l). A side effect of this operation is that interacting bits which are relatively far apart on the string are more likely to be disrupted (separated) by crossover than bits which are relatively close together. Conversely, non-interacting bits which are close together on the string are more likely to be treated by crossover as related components (i.e., not separated) than bits which are far apart.

Traditional crossover's bias in favor of short schemata (small clusters of neighboring bits) can be exploited by placing interacting bits next to each other on the chromosome when designing the representation for a particular problem. There are, however, problems with this solution. First, it may not be known which bits are related and which are not. In fact, the GA is most useful for large, complex, poorly understood search spaces where there is little or no a priori knowledge about which bits interact. Second, a linear string may not allow all interacting bits to be placed close to each other if some of them also interact with other bits. Third, the task is not simply to place interacting bits close to each other, but also to place non-interacting bits as far apart as possible. Unfortunately, there is usually no way to position bits on a one-dimensional string so that their relative locations reflect their degree of expected interaction even when this is known.

Hypothesizing that the positional bias of one-point crossover is more likely to work against the GA than to aid it, we developed an alternative form of crossover, shuffle crossover, to eliminate the bias . Shuffle crossover is similar to one-point crossover except that it randomly shuffles the bit positions of the two strings in tandem before crossing them over and then unshuffles the strings after the segments to the right of crossover point have been exchanged. Thus, crossover no longer has a single, consistent positional bias because the positions are randomly reassigned each time crossover is performed.

Shuffle crossover is similar, but not identical, to Ackley's "uniform" crossover operator [Ackley 1987]. Uniform crossover exchanges bits rather than segments, i.e., for each bit position, bits are exchanged between the paired individuals with fixed probability p. The number of bits exchanged with uniform crossover follows a binary distribution. Shuffle crossover can also be viewed as exchanging bits whose shuffled positions are to the right of the crossover point, however, the distribution of the number of bits exchanged is uniform, just as it is with traditional one-point crossover. This means that shuffle crossover preserves the same number of schemata as traditional crossover (although, of course, these will usually not be the same schemata). Moreover, like uniform crossover, shuffle crossover overcomes the positional bias of traditional one-point crossover, but without introducing an additional parameter to define the bit exchange probability.

In order to exhibit the qualitative difference between traditional crossover and shuffle crossover, we devised two special functions which we call the Plateau and Trap functions . Both functions consist of a series of thirty independent parameters, each represented by a 4-bit gene. The evaluation of the chromosome is the sum of the scores of the individual genes. In the Plateau function each gene is given a score of four if all four bits are 0^7s, otherwise, a score of zero. In the Trap function each gene is given a score of four if all four bits are O's, otherwise its score is determined by assigning one half point for each 1. Thus, the Plateau function is a series of independent all-or-nothing subtasks, whereas the Trap function is a series of independent bimodal subtasks with partial credit leading away from the global optimum for the subtask.

These functions were designed so that the building blocks, the 4-bit genes, would be independent of each other, but so that the bits within a gene would interact. In order to exhibit the effect of positional bias, two representations for these problems were studied: *adjacent,* in which the 4 bits in each gene are placed next to each other on the chromosome, and *distributed,* in which the 4 bits in each gene are maximally separated (i.e., the 4 bits of gene one were at loci 1,31,61 and 91, etc.)

Each of these functions was searched 50 times with both the adjacent and distributed representations using a population of 30 chromosomes, a mating rate (crossover rate) of 0.95 and a mutation rate of 0.01. These are values previously suggested as giving generally good search over a range of tasks [Grefenstette 1986]. Each search was allowed a maximum of 2000 trials (chromosome evaluations). The results of these experiments are shown in Tables 1 and 2. Note that the values in these tables reflect inverting (because our GA minimizes) and scaling the functions, so that the worst possible

1. Holland [Holland 1975) proposed the *inversion* operator to ameliorate the positional bias, but its use requires genes that can be interpreted independent of position. Other crossover operators that have less positional bias have also been proposed, including two-point crossover [De Jong 1975] and punctuated crossover |Schaffer and Morishima 1987|. See |Eshelman, Caruana, and Schaffer 1989] for a more complete discussion of the positional bias of various crossover operators.

2. Similar to Ackley's functions of the same name |Ackley 1987].

score is one and the best possible score is zero.

TABLE 1. The 4-bit Plateau Problem'

Operator	Representation Adjacent	Distributed
1 point crossover	.291	.347
shuffle crossover	.247	.258

ª Each cell contains the mean of the best individuals from 50 genetic searches, each to 2000 trials. The standard error of the mean for all cells was less than 0.009.

TABLE 2. The 4-bit Trap Problem[b]

Operator	Representation Adjacent	Distributed
1 point crossover	.261	.319
shuffle crossover	.289	.290

Each cell contains the mean of the best individuals from 50 genetic searches, each to 2000 trials. The standard error of the mean for all cells was less than 0.006.

As expected, the GA using traditional one-point crossover does significantly worse using the distributed representation than it does using the adjacent representation because of the bias against long schemata. Shuffle crossover, however, yields the same performance on both tasks with both representations; it has no length bias.

One interesting difference between the two functions is that traditional one-point crossover outperforms shuffle crossover on the Trap problem using the adjacent representation, but shuffle crossover is better on the Plateau problem than traditional crossover using either representation. Although the Trap function shows that it is possible for a task to be. constructed in such a way that the bias of the GA with traditional crossover works to advantage, we find that this is difficult to do in practice.

3. Empirical Tests

Tests with the Plateau and Trap functions show how traditional crossover's position dependent bias can influence search performance. They also demonstrate that shuffle crossover eliminates this bias and can yield improved performance. Both of these results, however, were obtained with functions and representations devised to exhibit specific characteristics. In this section we attempt to determine if traditional crossover's defining length bias hinders search in practice, and whether shuffle crossover eliminates this bias without otherwise hampering search. To do this, we compare the performance of traditional crossover and shuffle crossover on the minimization of five scalar-valued functions used by De Jong [De Jong 1975] to test the GA's performance on search spaces with a variety of characteristics. The test functions are summarized in Table 3. To compare the two methods, we adopted an arduous but, we hope, unbiased methodology. The objective of this methodology is to determine the parameter settings that should be used to compare the two crossover operators on the test problems.

TABLE 3 Test Functions

Fcn	Dimensions	Size	Description
f1	3	10^9	parabola
f2	2	10^6	Rosenbrock's saddle
f3	5	10^{15}	step function
f4	30	10^{72}	quadratic with noise
f5	2	10^{10}	Shekel's foxholes

Grefenstette [Grefenstette 1986] found a set of GA parameters (crossover rate, mutation rate, population size, and scaling window) that were optimal or near-optimal on fl-f5 using traditional one-point crossover. Unfortunately, we are unable to use the parameters he found because our experiments differ from his in four important ways. First, Caruana and Schaffer [Caruana and Schaffer 1988] demonstrated the superiority of Gray coding to binary coding for these functions; we now use Gray coding in all of our experiments . Second, we use an improved selection procedure devised by Baker [Baker 1987] that was not available at the time Grefenstette ran his experiments. Third, Grefenstette compared *online* performance at a fixed number of evaluations (5000). Gray coding

3. See |Schaffer et al. 1989] for a discussion of how genetic search is influenced by changes in the control parameters.
4. For an extension of this work that shows mixing binary and Gray representations is better yet, see [Caruana, Schaffer, and Eshelrnan 1989].
5. Online performance is the average performance of all trials explored by the search. It is appropriate when every trial must be "paid for" during a search [De Jong 1975|

improves performance enough that comparison at 5000 evaluations is no longer interesting: on some of the functions, the searches consistently find the optimum before 5000 evaluations. Rather than try to find a new fixed number of evaluations at which to compare performance, we use the mean number of evaluations required to find the global optimum as our criterion [Ackley 1987]. To locate an optimal or near-optimal parameter set for this criterion for the suite of problems, one must compensate for the fact that the mean and variance of this performance measure will be different for each of the five functions; one must normalize for these differences.

We compensated for the relative difficulty of the different functions by first estimating how well a GA *could* do on each one *independently.* To do this, we used a meta-GA [Grefenstette 1986] to find the best parameters for each of the two crossover schemes on each of the five functions. This required 10 meta-searches. Each meta-search considered mutation rates from 0.0 to 0.2, crossover rates from 0.0 to 1.0, and populations of 5 to 200 individuals. (We used the elitist strategy, scaling window = 1, and a modified GA that performs *restarts* - reinitializing the population with random strings — when 5 generations in a row fail to produce a new individual.) For each function we selected the performance of the crossover operator that worked best. The minimum average number of trials to find the optimum for functions fl-f5 were 600, 8872, 1310, 1944, and 1375 respectively.

To locate the best single set of parameters for each crossover operator for the whole suite of functions, we performed a second set of meta-searches using a single performance measure that consisted of a weighted sum of the performance on each of the test functions:

$$\textit{Total Performance} = \sum_{i=1}^{i=5} \log_2 \left(\frac{perf_i}{weight_i} \right) \qquad [1]$$

where *perf*$_i$ is the performance of the GA on *function*$_i$ (i.e., the number of evaluations required to find the optimum of that function) and *weighty* is the average number of evaluations required by the best GA for that function (above). This measure is attractive because it equalizes the importance of all functions by scaling them by their relative difficulty and by compensating for unequal variances. Total performances near zero indicate that the GA is able to do as well on the test functions using a single crossover operator and parameter set as it could do using the best operator with a parameter setting optimized for each function. Surprisingly, the two meta-GAs found the same best parameter set for both crossover operators: population size 10, mutation rate 0.023, and crossover rate 0.6.

Using these parameter values, each crossover operator was used to search each function 50 times. Table 4 shows the average number of function evaluations required to find the global optimum of each of the five test functions as well as the mean total performance for each crossover operator. Lower mean values suggest faster, more effective search. The table also contains the results of two-tailed t-tests of the means obtained with the two crossover operators. The significantly better performers are shown in bold print in the table.

Table 4 Performance of 1pt and Shuffle Crossover

	Traditional 1pt		Shuffle	
Fcn	Mean	Std.Err.	Mean	Std.Err.
f1	907	38	**801**	35
f2	10375	745	9486	561
f3	1310	85	1453	95
f4	2539	159	2588	148
f5	2254	137	**1798**	103
Perf.	1.12	0.24	0.83	0.21

These results suggest that shuffle crossover is, in practice, generally superior to traditional crossover. Shuffle crossover statistically outperformed traditional crossover on two of the five test functions and is not significantly worse on the other three. (Finding two of the five independent tests favoring shuffle crossover is itself significant at the 0.05 level.) Shuffle crossover's total performance is also better (though not statistically significantly so) than traditional lpt crossover's. (Other experiments have consistently given shuffle crossover the edge over traditional lpt-crossover in total-performance.)

4. Discussion

Inductive learning algorithms use bias in order to make learning more effective [Utgoff 1983], For example, biasing search towards certain regions of the search space is an important means of enabling efficient learning from a manageable set of examples. In the case of the GA, trials are allocated to clusters of genes based on their observed fitness. The GA's bias, as expressed in the *schema sampling theorem,* implies that an exponentially increasing number of trials will be allocated to sets of bits (schemata) occurring in the better performing individuals. However, the GA using traditional crossover also exhibits a sampling bias that is sensitive to the position of genes on the chromosome.

It is recognized that the GA's positional bias, like most biases, can be detrimental if not properly exploited, but it has always been assumed that there are ways of exploiting it. In particular, genes that are thought to interact should be placed near each other on the chromosome. The assumption has been that for some classes of problems there are *natural* representations that are optimal or nearly optimal. For example, it seems natural to place all of the bits coding for a single numeric parameter next to each other as was done in the tests on fl-f5. Unfortunately, it is not clear that this kind of representation will usually, or even often, be good. If the function exhibits behavior similar to a parity function, then traditional crossover's positional bias suggests that it would be better to group the least significant bits together. In fact, for parity problems, the natural representation of keeping the bits defining each parameter together is quite similar to the distributed representations used on the test problems in section 2. Thus for parity-like problems the natural representation is likely to be one of the worst representations.

The results presented suggest that the natural representation probably is not often the best — or perhaps even a good — representation. Any attempt to exploit traditional crossover's positional bias is in danger of being counterproductive because the obvious effects of the bias are counteracted by less visible, but still important, effects. Caruana and Schaffer noted a similarly opaque bias resulting from an interaction between representations (binary coding and Gray coding) and the mutation operator [Caruana and Schaffer 1988]. They used the term *hidden bias* to refer to the general class of biases resulting from unforeseen interaction between representations and search operators. The positional bias of traditional one-point crossover leads, we believe, to another instance of hidden bias.

The precise mechanisms by which traditional crossover's positional bias hinder search using natural representations is not intuitively obvious. We hypothesize that there are two undesirable aspects to the sampling bias caused by traditional crossover's favoring of short schemata. One of these is well known, the other is not. First, interacting clusters of bits that are far apart on the chromosome are less likely to propagate together. Second, short clusters of non-interacting genes (i.e. not causally related to the good performance, but occurring by chance in better individuals) are less likely to be disrupted, contributing to premature convergence of the gene pool to suboptimal individuals. This later phenomenon we call *spurious correlation.*

We believe that it is the combination of these two effects that accounts for the behavior observed on the test functions. The new shuffle crossover operator is designed to eliminate these biases by having a schema disruption probability that is independent of schema defining length. More generally, shuffle crossover provides for better sampling of interacting bits which are far apart (i.e., long defining length) and more disruption of neighboring bits that are only spuriously correlated. It is true that shuffle crossover will also be more disruptive of neighboring bits that truly interact, but our data shows this to be a price worth paying. In the absence of information prescribing how to work in concert with crossover's positional bias, we recommend eliminating this bias altogether. Our results indicate that eliminating this bias is relatively straightforward, incurs little computational cost, and is indeed beneficial.

5. Summary

Although the positional bias of the traditional one-point crossover operator is well known in the GA community, little has been done to suggest how to work in concert with it. Our experiments demonstrate that it may be harder to exploit this bias than had been thought, and that it may be more useful to eliminate it via shuffle crossover. Our experiments indicate that shuffle crossover is often superior to, and probably not often worse than, traditional crossover for GA function optimization. We believe that our results apply to most domains where it is not obvious how to select a representation to take advantage of traditional crossover's positional bias. We also believe that the hidden bias resulting from the interaction between traditional crossover's positional bias and the representation is not alone, and that other hidden biases wait to be made less hidden and then exploited, if possible, or eradicated.

References

David H. Ackley, *A Conncctionist Machine for Genetic Hillclimbing,* Kluwer Academic Publishers, Boston, MA, 1987.

James E. Baker, Reducing Bias and Efficiency in the Selection Algorithm, *Gen. Alg. and Their Appi: Proc. of the Second Intern. Conf. on Gen. Alg.$_y$* Lawrence Erlbaum Associates, Hillsdale, NJ, 1987, 14-21.

Richard A. Caruana and J. David Schaffer, Representation and Hidden Bias: Gray vs. Binary Coding for Genetic Algorithms, *Proc. Intern. Conf. Mach. Learn.,* Morgan Kaufmann, Los Altos, CA, June 12-14 1988, 153-161.

Richard A. Caruana, J. David Schaffer, and Larry J. Eshelman, Using Multiple Representations to Improve Inductive Bias: Gray and Binary Coding for Genetic Algorithms, Proc. Sixth Intern. Conf. on Machine Learning, Morgan Kaufmann, San Mateo, CA, 1989.

Kenneth A. De Jong, Analysis of the Behavior of a Class of Genetic Adaptive Systems, Ph.D. Dissert., Dept. of Computer and Communication Sciences, Univ. of Michigan, Ann Arbor, MI, 1975.

Larry J. Eshelman, Richard A. Caruana, and J. David Schaffer, Biases in the Crossover Landscape, Proc. of the Third Intern. Conf. on Gen. Alg., Morgan Kaufmann, San Mateo, CA, 1989.

John J. Grefenstette, ed., Proceedings of an International Conference on Genetic Algorithms and Their Applications, Lawrence Erlbaum Associates, Hillsdale, NJ , 1985.

John J. Grefenstette, Optimization of Control Parameters for Genetic Algorithms, IEEE Trans, on Sys., Man & Cybern. SMC-16,1 (Jan.-Feb. 1986), 122-128.

John J. Grefenstette, ed., Proceedings of the Second International Conference on Genetic Algorithms and Their Applications, Lawrence Erlbaum Associates, Hillsdale, NJ, 1987.

John H. Holland, Adaptation in Natural and Artificial Systems, Univ. of Michigan Press, Ann Arbor, MI, 1975.

J. David Schaffer and Amy Morishima, An Adaptive Crossover Distribution Mechanism for Genetic Algorithms, Gen. Alg. and Their Appl.: Proc. of the Second Intern. Conf on Gen. Alg., Lawrence Erlbaum Associates, Hillsdale, NJ, 1987, 36-40.

J. David Schaffer , ed., Proceedings of the Third International Conference on Genetic Algorithms, Morgan Kaufmann, Los Altos, CA , 1989.

J. David Schaffer, Richard A. Caruana, Larry J. Eshelman, and Rajarshi Das, A Study of Control Parameters Affecting Online Performance of Genetic Algorithms for Function Optimization, Proc. of the Third Intern. Conf. on Gen. Alg., Morgan Kaufmann, San Mateo, CA, 1989.

Paul E. Utgoff, Adjusting Bias in Concept Learning, 8th Intern. Joint Conf. on Art. Int., Karlsruhe, Germany, Aug. 1983.