Non-Linear Bit Arrangements in Genetic Algorithms

William A. Greene
Computer Science Department
University of New Orleans
New Orleans, LA 70148
bill@cs.uno.edu
504-280-6755

Abstract

In earlier research we laid out a theoretical basis for the supposition that genetic algorithms can succeed even if bits are arranged in ways other than as a linear sequence. In the present paper we report on certain experiments that show such success can occur in practice. Our experiments consider cases in which bits are arranged in two-dimensional grids, in three-dimensional cubes, and as the nodes of a complete binary tree. Moreover, our experiments consider several ways of cutting parental genetic material when performing mating with crossover, and also consider several notions of fitness. Our problems are not particularly difficult, but clearly show the convergence we seek, under these much liberalized ways of arranging bits.

1 INTRODUCTION

In classical Genetic Algorithms, bits are arranged in a linear sequence. In the Schema Theorem of (Holland, 1975), this arrangement is exploited to interrelate the distance between bits and the probability that a schema will be disrupted under crossover. Specifically, the lower is the schema's length (that is, the maximum distance between fixed positions in it), then the lower is the probability that the schema will be disrupted under a random cut made along the sequence of bits.

Alternative bit arrangements have been investigated by earlier researchers. Tree-like arrangements of bits have been studied by Koza and also Poli and Langdon, for instance in (Koza, 1992) and (Poli and Langdon, 1997).

In our earlier research (Greene, 2000), it was shown that how one arranges bits may potentially be vastly liberalized. There it is supposed that bits are arranged in a connected graph, that there is some notion of distance between bits, and that there is some natural way to cut the graph in two, for purposes of performing crossover of parental genetic material. Assuming that distance and cutting have the relationship that a schema's diameter (the maximum distance between fixed positions in it) is as big as the schema's disruption probability (that is, the probability that a random cut will disrupt the schema), then we were able to prove an extension to the classical Schema Theorem. The upshot of the extended theorem is that once again we may expect the burgeoning of building blocks.

Finally in (Greene, 2000), experiments about the theory were reported; they explored the relationship between distance and cutting, in two scenarios. In the first, bits were arranged in a tree, which was cut by clipping out a subtree. In this first scenario, the overwhelming statistical evidence was that schema diameter always exceeded schema disruption probability. For the second scenario, bits were arranged in a three-dimensional cube, which was sliced by a random plane. In this second scenario, schema diameter exceeded schema disruption probability for most schema of low diameter.

In this paper, we put the theory aside and explore what happens empirically. We run experiments with actual problems wherein bits are arranged in ways other than a linear sequence. We are on the trail of a proof-of-concept, namely, that alternative bit arrangements, when coupled with appropriate crossover operators plus the other machinery of geneticism, can have good results. As a proof-of-concept, we have kept our problems simple, but non-trivial. As will be seen, we do get good results. Namely, for our problems, the populations do converge towards solutions to the problems at hand.

2 A NON-LINEAR SCHEMA THEOREM

In this section we briefly review the theorem of (Greene, 2000), whose proof parallels the development of the classical Schema Theorem as found in (Goldberg, 1989). There is a universe of interest. Each individual in the universe is represented by a collection of bit values. Bits are arranged as the nodes in some connected (and finite) graph G. It is assumed there is some notion of distance between graph nodes (perhaps path length in G). Also it is assumed there is some notion of natural ways to cut G into two non-empty subsets of its nodes (possibly by clipping one or several edges in G). A *schema* is exactly as in classic Genetic Algorithms, namely, just like an individual, except that some “bit values” are allowed to be a wildcard

character, say the asterisk, *. Nodes in the schema whose value is one of the actual bit values of 0 or 1 are termed *fixed*; nodes whose value is the wildcard character are termed *unfixed*. We think of the wildcard character as matching either or 0 or 1, and say that a particular individual is a *representative* of a schema if it agrees with the schema at every one of the schema's fixed nodes. A schema with k unfixed (wildcard) nodes thus has altogether 2^k representatives in the universe. The *order* of a schema is the number of fixed nodes in it.

It was assumed there is some notion of distance between *G*-nodes. Given a set *S* of nodes, define its *diameter*, $\Delta(S)$, to be the maximum distance between any two nodes of *S*. Define the *relative diameter*, $rel\Delta(H)$, of a schema *H* to be $\Delta(fixed(H))/\Delta(G)$ where *fixed*(*H*) is the set of fixed nodes of *H*. It follows that $rel\Delta(H)$ is a number in the real unit interval [0, 1]. It was assumed there is some natural notion of ways to cut *G* into two non-empty subsets of its nodes. Finally, define the *disruption probability*, *dp*(*H*), of a schema *H* to be the probability that a random such cut will separate a pair of fixed nodes of *H* from one another.

Non-Linear Schema Theorem: Let *H* be a schema. Assume (i) the mutation rate p_m is low, (ii) *H* has low order, (iii) *H* has low relative diameter $rel\Delta(H)$, and (iv) $rel\Delta(H) \geq dp(H)$. If representatives of *H* have somewhat above average fitness within the population *P*(*t*) at time *t*, then the number of representatives of *H* can be expected to increase in the next generation *P*(*t+1*).

A lesson from (Greene, 2000) is that the distances between nodes are more important than the connections between them. It ought to be the case that nodes which are closer together are less likely of being separated under a random cut. This is to be viewed as a continuity property.

3 THE GENETIC OPERATORS IN USE

In this section we describe the genetic operators used in the experiments conducted for this paper. Since we were pursuing a proof-of-concept, we were content to demonstrate the behavior of convergence, and so in general we did not trouble to do much fine-tuning. Essentially the same course was taken in all the experiments.

There is a population, of a fixed size, denote it *N*. (In general, our population size has stayed fairly low; such as *N* = 100.) Evolutionary change is generational: there is a population *P*(*t*) at time *t*, which in a generational change is replaced by the population *P*(*t+1*) at time *t+1*. The new population is formed as follows. A certain number (we chose 4) of the fittest individuals automatically survive into the next generation; these are termed the *elite survivors*. Then the rest of the next generation is filled up to the same size as the current generation, by mating with crossover.

For mating, parents are chosen so as to favor fitter individuals for parenting. First, the fitnesses of all the individuals in population *P*(*t*) are linearly transformed to the real range [1, 4]; such scaling is known to avoid some problems of premature convergence. Then a weighted roulette wheel is used to stochastically select individuals for mating: the individual a_j, with fitness f_j, is selected for parenting with a probability equal to its relative fitness within the population, that is, with probability $f_j/(\Sigma_i f_i)$.

Once two parents have been chosen, each is cut by a randomly chosen natural cut (whose existence was assumed earlier), then one child is re-formed from complementary parental “wedges” and added to the next generation. (In fact, two such children could be so formed, but we formed just one.) Once the next generation has been filled up to the same size as the current generation, each individual in it, except for the elite survivors, is subjected to some degree of mutation.

The mutation undergone by population members (except for the elite survivors, who are spared any mutation) is stochastic and graduated. Each individual undergoes some number of mutation attempts. The mutation is graduated, in that the number of attempts is (negatively) proportional to the individual's fitness, with the least fit individual undergoing some maximum number of attempts (which we usually chose to be 40). The mutation is stochastic, in that a mutation attempt succeeds with only a 50% chance. Thus, the least fit individual may undergo 40 mutations, but we expect it to undergo only about 20. Not surprisingly, a mutation consists of flipping the bit value at a randomly chosen node.

This completes construction of the next generation *P*(*t+1*) of the population.

In our experiments, an initial population was formed, which consisted of random individuals. Then, generation after generation, *P*(*1*), *P*(*2*), *P*(*3*), etc., of the population was evolved, until either there surfaced an individual that was an exact solution to the problem at hand (and so had maximum fitness), or until some maximum number of generations had been evolved.

4 EXPERIMENT: FIGURE RECOGNITION

For this problem, the bits of an individual are arranged in a 24×24 grid. There is a target bit pattern, which we chose to be a familiar figure, namely, a representation of the letter capital 'A'. Figure 1.a shows the target. First we define the *error* of an individual to be its Hamming distance from the target. (Recall, to calculate the Hamming distance between two individuals, summing up over all pairs of corresponding bits between the two individuals, we add either 0 or 1 according as the pair of corresponding bits either do or do not equal.) The maximum error is then the number of bits, which here is 24 * 24 = 576. Then we define the fitness of an individual to be the maximum error (576) minus the individual’s error. It follows that fitnesses are integers in the range from 0 to 576. For crossover, parents were cut by boxes. Specifically, a random rectangular subset of the grid is chosen (its left boundary could potentially extend as far as the left boundary of the full 24×24 grid, and similarly for

```
000000000000000000000000
000000000000000000000000
000000000011110000000000
000000000011110000000000
000000000011111000000000
000000000011001100000000
000000001100001100000000
000000001100001100000000
000000011100001110000000
000000011000000110000000
000000110000000011000000
000000111111111111000000
000001111111111111100000
000001100000000001100000
000011100000000001110000
000011000000000000110000
000111000000000000111000
000110000000000000011000
001110000000000000011100
011111000000000000111110
011111000000000000111110
000000000000000000000000
000000000000000000000000
000000000000000000000000
```

Fig. 1.a: The target.

```
000000000000000000000100
000000000000000000000100
000000000011110000000000
000000000011110000000000
100000000111111000000000
000000000110011000000000
000000001100001100000000
000000001100001100000000
000000011100001110000000
000000001000000010000000
000000110000000011000000
000100111111111111000000
000000111111111111100000
000001100000000001100000
000011100000000001110000
000011000000000000110000
000111000000000000111000
000110000000000100011000
001110000000000000011100
011111000000000000101110
011111000000000000111110
000000000000000000000000
000000000000000000000000
000100000000000000010000
```

Fig. 1.b: A typical trial's best individual, with fitness = 563 (= 97.7% of optimal fitness 576).

Figure 1: Figure Recognition

the other three boundaries). Then a child is formed by copying bits inside the rectangular subset of one parent and copying bits outside the rectangular subset of the other parent.

(We seem to be ignoring the distances and connections between nodes in our graph (grid). We may suppose that a grid point is connected to its immediate neighbors in the four compass directions, and we may suppose that distance in the graph then is minimum path length between nodes.)

Table 1 shows the results of 20 trials of this experiment. (Atypically, for this problem the maximum number of mutations is 20, not 40.) The target was never encountered, but it was closely approximated. The best individual found on each trial had average fitness 562.85, or 97.7% of the optimal fitness of 576. Figure 1.b shows a typical trial's best individual.

This problem, though not particularly difficult, does illustrate that we can arrange bits in a two-dimensional grid, wed that decision with a compatible method of crossover which is conducive to solving the problem at hand, and get the convergence we desire.

We did not expect this problem to be terribly difficult for our genetic problem solver. Consider the fitness landscape. There is one point of maximum fitness. An individual has 576 immediate neighbors in Hamming space (namely, flip any one of the 576 bits), and each such neighbor has a fitness one less or one greater than that of the individual. The landscape everywhere gently slants toward the point of maximum fitness. Simple hill-climbing will always eventually arrive at the point of maximum fitness. For this problem, the best way to find the individual of maximum fitness would be a hybrid approach: first run a genetic algorithm to approximately reach the absolute maximum, then use hill-climbing to directly traverse the remaining short distance to it.

If we had followed a more traditional approach, by putting the bits of the grid in some linear sequence, such as in row-major order, and used traditional one- or two-point crossover, then probably we would have achieved an eventual "best individual" which decoded to a pattern not so dissimilar from that of Figure 1.b. But intuition suggests that approach would have taken quite a bit longer than the two-dimensional track we pursued.

Table 1: Figure Recognition

Optimal fitness	576
Population size	50
Max number of generations	2000
Max number of mutations	20
Number of trials	20
Avg final gener'n number	2000.0
Avg best fitness	562.85
... as % of optimal fitness	97.7%

5 EXPERIMENT: EIGHT (ETC.) QUEENS

This is the classic Eight Queens Problem, and a variant. In chess, the queen can move horizontally, vertically, and diagonally. Two queens *conflict* if they lie on the same row, column, or diagonal. A solution to the Eight Queens Problem consists of a placement of eight queens on the board so there are no conflicts between the queens; thus, there is one queen in each row and each column and no two queens lie on the same diagonal.

For this problem, we arrange bits in an 8×8 grid, the same dimensions as a standard chessboard. A bit value of 0 in the grid signifies that the chessboard square is empty, and a bit value of 1 signifies that the square is occupied by a queen.

This is an interesting problem for attack by a genetic algorithm. There are several distinct solutions to the problem, so in the fitness landscape there are several absolute maxima. Also we note that in our 8×8 grid, bits that are far apart can be strongly related with regard to fitness. For instance, two opposing corners should not both contain queens. Of course, a solution must satisfy a large number of constraints.

How shall we measure the fitness of an individual? First we will think in terms of error. As well as by having conflicting queens, an individual may also err by having too few or too many queens. We will add up all errors we spot. For each row, count the number of queens on that row, call it n, and to the count of errors, add $| n - 1 |$. Do likewise for each column. Each diagonal (both those of positive slope and those of negative slope) should have at most one queen on it. For each diagonal, count the number of queens on it, call that n, and if $n > 1$, then to the count of errors add $n - 1$. Thinking in this way, the most erroneous individual has a queen at every grid cell (which of course is far too many queens); it turns out that such an individual has error equal to 2 * (edgesize-1) * (2*edgesize - 1). We let an individual's fitness be the maximum error minus the individual's error. Thus, the maximum value a fitness can be is the same as the maximum value of error, which we just revealed.

For crossover, parents should be split and re-combined in ways that are conducive to solving the problem at hand. For this problem we do the following. At random we choose a row or column or diagonal. Such a line splits a parent in two, of course, namely the bits to one side versus the other side of such a line. A child is then formed from parental "wedges" in the way one would expect.

We let our initial population consist of individuals each of whose bits were randomly set to 0 or 1 (and so each such individual has quite an excess of queens). Table 2 shows the results of a running 20 trials, each of at most 500 generations. Only once was a perfect solution found (on generation number 336), but on the other trials our genetic problem solver came very close to finding an individual of maximum fitness. Over the 20 trials, the best individual found in a trial had an average fitness of 208.3, or 99.2% of the maximal fitness of 210. Figure 2 shows a typical best individual found in a trial.

Table 2: Eight Queens Problem.

Optimal fitness	210
Population size	50
Max number of generations	500
Max number of mutations	40
Number of trials	20
Avg final gener'n number	491.8
Avg best fitness	208.3
... as % of optimal fitness	99.2%

```
— — — — — — Q —
— — Q — — — — —
— — — — — Q — —
— Q — — — — — —
— — — — Q — — —
— — — — — — — Q
— — — — — Q — —
— — — Q — — — —
```

Figure 2: Eight Queens Problem.

A typical trial's best individual, with fitness of 208, versus optimal fitness 210. (The two errors: column 6 has two queens, and column 1 has none.)

We vary this problem, by swelling edgesize to 20; thus we attack the Twenty Queens Problem. This should be a very challenging problem! The results are quite satisfying. Table 3 shows the results of 20 trials, and Figure 3 shows a typical best individual found on a trial. In no trial did a perfect solution arise, but on average the best individual found had a fitness equal to 99.3% of the maximum fitness.

6 EXPERIMENT: TREES

We next investigated bits arranged as the nodes in a complete binary tree. The tree has $N = 2^h - 1$ nodes in it, where h is the height of the tree. There are 2^{h-1} leaves, all on level h. Except for cutting parents at crossover time, the machinery of geneticism is as described earlier.

Table 3: Twenty Queens Problem	
Optimal fitness	1482
Population size	100
Max number of generations	2000
Max number of mutations	40
Number of trials	20
Avg final gener'n number	2000.0
Avg best fitness	1472.1
... as % of optimal fitness	99.3%

```
— — — — — — — — — — — — — Q — — — — — — —
— — — — — — Q — — — — — — — — — — — — — —
— — — — — — — — — — — — — — — — — — — — Q
— — — — — — — — — — — — Q — — — — — — — —
— — — — — — — — — — — — — — — — — — — Q —
— — — — — Q — — — — — — — — — — — — — — —
— — — — — — — — — — — — — — — — — — — — —
— — — — — — Q — — — — — — — — — — — — — —
— Q — — — — — — — — — — — — — — — — — — —
— — — — — — — — — — — — — — — — — — — — —
— — — — Q — — — — — — — — — Q — Q — — — —
— — — — — — — — — — — Q — — — — — — — — —
— — — — — — — — — — — — — — — Q — Q — — —
— — — Q — — — — — — — — — — — — — — — — —
— — — — — — — — — — — — — — — — — — — — —
Q — — — — — — — — — — — — — — — — — — — —
— — — — — — — — — — — — — — — — — — Q — —
— — — — — — — — — — Q — — — — — — — — — —
— — Q — — — — — — — — — — — — — — — — — —
— — — — — — — — — Q — — — — — — — — — — —
```

Figure 3: Twenty Queens Problem.

A typical trial's best individual, with fitness of 1472, versus optimal fitness 1482.

We formed a child from two parents by selecting one of the 2^h - 2 proper subtrees and replacing one parent's subtree by that of the other. Two ways of selecting subtrees occurred to us. One way is to choose each with uniform likelihood. The other way is to favor larger subtrees. We achieved the latter as follows. Each proper subtree was assigned a mock fitness equal to its size. Then we proceeded as with stochastically choosing parents for mating: mock fitnesses were linearly scaled to the range [1, 4] and these values were used to create a weighted roulette wheel for choosing subtrees. Here is our rationale for favoring larger subtrees. In our tree, one-half the nodes are on the lowest level and another one-fourth are on the level above that. Thus, it is a somewhat remote chance that a larger subtree gets chosen if we choose subtrees uniformly, whereas it may be advantageous for crossover to exchange a larger parental chunk.

For our first tree experiments, we chose a target individual whose properties were tree related, and let the fitness of an individual be the maximum Hamming distance *N*, minus the individual's Hamming distance to the target. We considered three target individuals. Target # 1 had 1's at all the nodes of the root's rightmost grandchild, and 0's everywhere else (thus, about one-fourth of the bits were 1's). Target # 2 had 1's on the even levels (root, root's grandchildren, great-great-grandchildren, etc.) and 0's elsewhere. Target # 3 had 1's at the leaves and 0's elsewhere. Table 4 shows the results of these experiments. For these experiments the number of tree nodes is 511, which is quite a few nodes. The results are excellent. The genetic searcher always came very close to finding the targeted individual (it finds an individual of fitness around 510, versus 511) and 40% of the time it finds exactly the targeted individual.

Table 4: Trees, with fitness determined by Hamming distance to a targeted individual.		
Optimal fitness	511	
Population size	50	
Max number of generations	1000	
Max number of mutations	40	
Number of trials	20	
	Subtrees	
	Chosen Uniformly	Favoring Larger
Target 1		
Avg final gen #	969.3	983.2
Avg best fitness	510.25	510.05
Target 2		
Avg final gen #	970.7	956.15
Avg best fitness	509.85	510.45
Target 3		
Avg final gen #	954.45	977.25
Avg best fitness	510.45	509.95

We ran a further two sets of experiments with trees. For these, the fitness landscapes are much different, and there are many absolute maxima. In one set, termed Twins, the fitness of an individual equals how many pairs of sibling nodes have the identical value. In the other set, termed Palindrome, the fitness of an individual equals how many pairs of diametrically opposed nodes have the identical value. (Here, by the diametrically opposed node, we mean

the one located at the reflection across the line which passes through the root and the middle of each tree level.) For both these experiments, again our tree has 511 nodes, so there are 255 pairs in question and therefore the maximum fitness is 255. The results of the experiments are in Table 5. To our surprise, the evidence is that these two problems are markedly easier than our earlier ones of searching for a targeted individual. Here the genetic searcher always finds an individual of maximum fitness. Moreover, it does so at the cost of examining about half as many individuals as in the earlier experiments (the average generation number on which an individual of maximal fitness is found is circa 475 and 445, respectively, for Twins and Palindrome, versus an average final generation number circa 970 in the earlier experiments). We note that for the Palindrome experiments, our method of cutting parents is of debatable help in preserving individuals of high fitness, so it must be survival of the fittest that mostly is driving the successful searches.

As Tables 4 and 5 reveal, neither of the two approaches to choosing subtrees at crossover time (choosing all subtrees with uniform likelihood, versus favoring larger subtrees) is convincingly better than the other.

Table 5: Trees, with fitness determined by relationships between pairs of nodes.

Optimal fitness	255	
Population size	50	
Max number of generations	1000	
Max number of mutations	40	
Number of trials	20	
	Subtrees	
	Chosen Uniformly	Favoring Larger
Twins		
Avg final gen #	480.45	473.25
Avg best fitness	255	255
Palindrome		
Avg final gen #	449.6	441.15
Avg best fitness	255	255

7 EXPERIMENT: CUBES

In this set of experiments, bits were arranged in a 3-dimensional cube. In Euclidean space, a *lattice point* is a point all of whose coordinates are integers. Our cubes consist of the lattice points (j, k, l) where the integer coordinates j, k, and l are all in the range 0 to E, where $E + 1$ is edge size. For the experiments we report below, edge size $E + 1$ equals 11, which means there are $11^3 = 1331$ bits altogether; this is a very large number of bits.

First we let an individual's fitness be determined in the now familiar way by its Hamming distance to some targeted individual. For the targeted individual, we chose that individual having 1's along the three equators of the cube and 0's elsewhere.

We examined two ways of cutting cubes, to perform crossover between parents. In the first way, a random plane in Euclidean 3-space which intersects the cubes is chosen. (A plane in Euclidean 3-space consists of all vectors (x, y, z) which satisfy $\alpha x + \beta y + \gamma z = \delta$, for some choice of real numbers α, β, γ, δ. We choose random α, β, γ, δ in the real interval (-1, 1) and then first confirm that the associated plane intersects our cube.) In the second way, a random rectangular solid subset (or “box”, with sides parallel to those of the cubes) is chosen. This second method is the analogue to how we cut the 24×24 grids in our first experiment.

Table 6 reports the results of experiments for searching for the targeted individual. Note that for this choice of fitness, the maximum fitness is the same as the number of nodes in the cube, or 1331. This was a challenging problem. As in certain earlier experiments reported above, the fitness landscape slopes gently towards the one absolute maximum. The search never located the targeted individual, though on average it came rather close. Other experiments suggested that more generations would lead the searcher ever closer to the target.

Table 6: Cube, with fitness determined by Hamming distance to a targeted individual.

Optimal fitness	1331	
Population size	100	
Max number of generations	2000	
Max number of mutations	40	
Number of trials	20	
	Cutting Cubes	
	by Planes	by Boxes
Avg final gen #	2000.0	2000.0
Avg best fitness	1261.95	1293.45
... as % opt’l fitness	94.8%	97.2%

For the remaining experiments in this section, we defined fitness as the number of antipodal pairs that had the same value. By the point *antipodal* to (j, k, l), we mean the one located diametrically across from the center of the cube; the antipodal point has coordinates ($E - j$, $E - k$, $E - l$). With this notion of fitness, the maximum fitness is half the number of nodes, or 665. Now the fitness landscape has many absolute maxima. The results of these experiments are given in Table 7. Here we have met with greater success. Here the search comes very close indeed to finding an individual of maximum fitness, and sometimes finds one exactly.

Tables 6 and 7 indicate that, for the problems at hand, cutting the cubes by boxes is a somewhat better approach than cutting them by random planes.

8 CONCLUSIONS

(Greene, 2000) presents the theory to support the contention that building blocks will burgeon and geneticism will succeed even if bits are arranged in ways other than as a linear sequence. Here we give experiments that show the latter happens in practice. Our experiments have been kept simple, so as to leave uncluttered the observation of a population converging as desired. We have examined essentially four different problem areas: a 2-dimensional arrangement of bits, a 2-dimensional problem with complex constraints (eight queens), bits arranged in trees, and 3-dimensional arrangements. Also, we have used various fitness measures, some with simple fitness landscapes, and some with complex landscapes having numerous absolute maxima.

Table 7: Cube, with fitness determined by antipodal points having the same value.

Optimal fitness	665	
Population size	100	
Max number of generations	2000	
Max number of mutations	40	
Number of trials	20	
	Cutting Cubes	
	by Planes	by Boxes
Avg final gen #	1999.2	1889.85
Avg best fitness	660.4	664.35
... as % opt'l fitness	99.3%	99.9%

The lesson of these experiments is that by choosing crossover methods which are compatible with the representation of the problem at hand, then in concert with the other evolutionary forces, we can achieve success even when bits are arranged in ways other than as a linear sequence.

References

Goldberg, David (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison-Wesley Publishing.

Greene, William A. (2000). A non-linear schema theorem for genetic algorithms. In D. Whitley *et al*. (eds.), *Proceedings of the 2000 Genetic and Evolutionary Computation Conference*, pp 189-194. Morgan Kaufmann Publ., San Francisco, CA.

Holland, John (1975). *Adaptation in Natural and Artificial Systems.* Ann Arbor, MI: University of Michigan Press.

Koza, John (1992). *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA: The MIT Press.

Poli, Riccardo, and Langdon, W. B. (1997). A new schema theory for genetic programming with one-point crossover and point mutation. In J. R. Koza *et al*. (eds.), *Genetic Programming 1997: Proceedings of the Second Annual Conference*, pp 278-285. Morgan Kaufmann Publ., San Francisco, CA.