Strong Recombination, Weak Selection, and Mutation

Alden H. Wright
Computer Science
University of Montana
Missoula, MT 59812
alden.wright@umontana.edu
J. Neal Richter
Computer Science
Montana State University
Bozeman, MT 59714
richter@cs.montana.edu

November 27, 2006

Reference: GECCO 2006 Genetic and Evolutionary Computation Conference, Maarten Keijzer et al., editors. Association for Computing Machinery, 2006. Pages 1369–1376.

Abstract

We show that there are unimodal fitness functions and genetic algorithm (GA) parameter settings where the GA, when initialized with a random population, will not move close to the fitness peak in a practically useful time period. When the GA is initialized with a population close to the fitness peak, the GA will be able to stay close to the fitness peak. Roughly speaking, the parameter settings involve strong recombination, weak selection, and require mutation. This "bistability" phenomenon has been previously investigated with needle-in-the-haystack fitness functions, but this fitness, when used with a GA with random initialization, requires a population size exponential in the string length for the GA to have nontrivial behavior. We introduce sloping-plateau fitness functions which show the bistability phenomenon and should scale to arbitrary string lengths. We introduce and use an unitation infinite population model to investigate the bistability phenomenon. For the fitnesses and GAs considered in the paper, we show that the use of crossover moves the GA to its fixed point faster in comparison to the same GA without crossover.

1 Introduction

A major goal of GA theory is to understand the role of crossover in genetic algorithms. This paper shows that there are situations where crossover can lead to a GA staying far from the optimum of a single-peak fitness function for a long time. This happens when selection pressure is weak, recombination is strong, and mutation is within a range that depends on the selection pressure. It is the disruptive aspects of crossover and mutation that are responsible for this slowdown. We introduce "sloping plateau" fitness functions to illustrate these phenomena. We conjecture that there are sloping-plateau fitness functions where a standard proportional selection GA with strong recombination will require exponentially many fitness evaluations to reach a population with many points on the plateau, whereas a corresponding GA without recombination will reach this kind of population quickly.

This paper considers the maximization of pseudo-boolean functions $f : \Omega \longrightarrow \mathbb{R}_0^+$, where $\Omega = \{0,1\}^\ell$ denotes the space of binary strings of length ℓ. Such a function f is called *function of unitation* if $f(x)$ depends only on the number of ones in the binary string x. We develop a unitation coarse graining of the standard GA infinite population model with uniform crossover, and discus the strengths and weaknesses of this and other infinite population models. The unitation model is used in the analysis of examples where crossover slows down convergence.

2 Review of previous work

There has been much prior work on the role of crossover in GAs, and we are only able to survey work that we feel is most relevant to our work.

The *building block hypothesis* states that crossover in a GA works to combine short low-order schemata (building blocks) into high-fitness strings [5]. There are fitness functions (such as concatenated trap functions [6]) where the building block hypothesis provides a good explanation of how a GA works. There are other situations where the building block hypothesis as stated above does not provide much useful insight.

Based on the building block hypothesis, Mitchell et al [10, 4] proposed the Royal Road family of fitness functions. Their original intention was to give examples of fitness functions where the GA with crossover would work very well. However, they discovered that a GA with crossover did not work nearly as well as a hill climbing algorithm that flips one bit on each move, and accepts new individuals with fitness equal to the current individual. (In other words, this hill climber does a random walk when the fitness is flat.) Hitchhiking was proposed as one explanation of why the GA did not do as well as expected.

Suzuki and Iwasa [18] investigated the role of crossover in a GA on a needle-in-the-haystack fitness (which they called a Babel-like fitness). They developed approximate models for the time T_d that it takes a GA that starts with a population consisting of a multiple copies of a single random string to reach a population dominated by the needle string. (The needle string dominates the population when it is over half of the population.) Their models assume linkage equilibrium[1] and includes finite population effects. They modeled both uniform crossover and a version of multi-point crossover. They defined the acceleration due to crossover as $A^d_{cross} = \frac{T_d|_{\text{without crossover}}}{T_d|_{\text{with crossover}}}$. They found that for appropriate mutation and crossover rates, the acceleration due to crossover could be large (up to about 11 for string length 12 and 70 for string length 20). However, when the crossover rate was too high, the domination time T_d became very large. There was an interaction between population size and mutation rate to achieve the highest acceleration: as the population size increased, the mutation rate for highest acceleration decreased. Their qualitative conclusions for this type of fitness were:

- "The crossover rate should not be to high nor too low for fast evolution."
- "The mutation rate must be adjusted to a moderate value to enhance evolutionary acceleration due to crossover."
- "To achieve a large acceleration effect by crossover, the order of the advantageous schemata to be created needs to be sufficiently large."

Ingo Wegener and his students have initiated a research program of analyzing evolutionary computation algorithms as randomized algorithms. Their results give rigorous bounds on the complexity of some evolutionary algorithms on some fitness functions. Here we discuss only those papers that relate to crossover.

Jansen and Wegener [8, 7, 9] have given carefully constructed fitness functions where a GA with crossover can find the optimum point in polynomial time (with high probability), whereas a mutation based GA without crossover will require exponential time to find the optimum. In these examples, crossover is able to jump a large gap in the fitness by recombining strings on the edge of the gap. This work suggests that crossover is most helpful at the end of a run, and that building blocks are not necessarily short or low order.

2.1 Bistability and related phenomena

Boerlijst et al [2] introduced bistability in the context of a model of viral recombination. Bistability refers to a dynamical system with two stable fixed points. In the context of evolutionary computation, bistability is the situation where a dynamical systems model of a evolutionary computation algorithm applied to a single-peak fitness landscape has two stable fixed points.

A tractable dynamical systems model of bistability was given in [25, 24] for a genetic algorithm with proportional selection and gene pool crossover. (Gene pool crossover is equivalent to an assumption of linkage equilibrium, and is used in some estimation of distribution algorithms such as UMDA [11] and PBIL [1].) In the case of a needle-in-the-haystack fitness, the fixed points can be found by solving a single-variable equation, and the stability of fixed points

[1] A population is at linkage equilibrium if the population distribution is determined by the order-1 schemata frequencies. In other words, the frequency of a string is equal to the product of the corresponding bit frequencies. For example, suppose that for 2 bits, the frequency of $0*$ is $1/4$ and the frequency of $*0$ is $1/3$. Then the frequencies of $00, 01, 10, 11$ will be $1/12, 1/6, 1/4, 1/2$ respectively.

also determined by a single variable equation. Thus, these infinite population model results apply for all string lengths. Gene pool crossover can be used as an approximation to uniform crossover. This work was extended to tournament selection in [22].

In the cases investigated in [23, 24, 25, 22] one of the fixed points will be close to a uniform population consisting of copies of the best individual, and the other fixed point will be close to the population with equal representation of all strings in the search space (the center of the simplex). In practical terms, this can mean that when the GA is started with a random initial population, the algorithm can get "stuck" near the center of the simplex fixed point and take a very long time to move near to the fitness peak fixed point. However, when started with a population near to the fitness peak or the corresponding fixed point, the population can stay near this fixed point for a long time. (Note that a GA with standard non-zero mutation is exactly modeled by an ergodic Markov chain, and thus the GA will eventually visit every possible population.)

As described above, Suzuki and Iwasa [18] found that the time T_d to domination went to "infinity" as the crossover rate increased. The crossover rate at which this happens for uniform crossover, for string length 12, and mutation rate 0.002 was correctly predicted in [23].

This paper extends previous work on bistability. Previous bistability work emphasized the needle-in-the-haystack fitness function. A finite population GA with random initialization requires a population size which is exponential in the string length to be influenced by the needle string. Otherwise, the GA sees only a flat fitness landscape. Thus, it is of interest to understand how bistability scales with string length in a situation where the size of a finite population does not increase exponentially with the string length.

2.2 An intuitive explanation of bistability

In a genetic algorithm, selection acts to increase the frequency of more fit individuals. However, a source of variation is needed since selection by itself does not introduce any new kinds of individuals. Mutation and crossover both introduce new kinds of individuals, but they do this at the expense of disrupting some of the more fit individuals. When the bistability phenomenon prevents or slows progress towards the optimum, it is because the disruptive properties of mutation and crossover are overwhelming selection.

In the following, we will consider the infinite population model since this removes the necessity for considering genetic drift.

In the case of the needle-in-the-haystack fitness we can be more specific. In this fitness, all strings have equal fitness except for the all-zeros string which has a higher fitness. Selection will increase the frequency of the all-zeros string which will increase the frequency of the zero alleles.

Crossover does not change the expected allele frequencies. However, crossover does decrease the correlation between the alleles. (In other words, crossover moves the population towards linkage equilibrium.) Since the all-zeros string represents correlation between alleles, the frequency of the all-zeros string will be reduced by crossover. However, since the frequency of

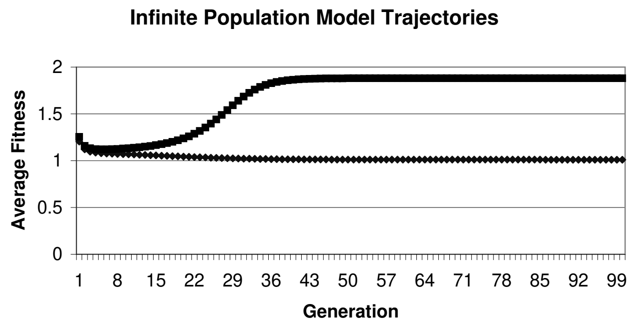

Figure 1: Unitation Model Trajectories Needle Fitness, $\ell = 10$

the zero alleles was increased by selection, even the most extreme crossover (namely gene pool crossover) will not decrease the frequency of the all-zeros string to less than what is was before selection. Thus, if there is no mutation, there will be steady (but possibly slow) progress towards a population consisting entirely of the all-zeros string. (The results of [20] show that a no-mutation GA with proportional selection on a single-peak fitness landscape can have only one stable fixed point at the uniform population consisting of copies of the optimal string.)

Mutation will drive the allele frequencies towards $1/2$. If this mutation pressure overcomes the combined effect of selection and crossover then there will be no progress towards the optimal population.

For a specific example, let's consider a needle-in-the-haystack fitness function where the needle (the all-zeros string) has fitness 6 and all other strings have fitness 1. Let the string length $\ell = 10$. Let C be the center of the simplex population with a weight of $2^{-\ell}$ on every string, and let N be the needle string population with weight 1 on the all-zeros string and 0 weight on all other strings. Let $P = \frac{1}{25}N + \frac{24}{25}C$, and let $Q = \frac{1}{20}N + \frac{19}{20}C$. Figure 1 shows the infinite population model average fitness trajectories starting at P and Q. Starting at P the GA the fitness decreases to the fitness of the center-of-the-simplex fixed point. In other words, the GA is going downhill on the fitness landscape. This is very counterintuitive. On the other hand, starting at Q, which is a little closer to the needle, fitness first decreases and then increases to the fitness of the needle fixed point.

Prior work suggest several ways to avoid bistability. There is always a range of mutation rates for bistability. Thus, either sufficiently raising or lowering the mutation rate may move the GA out of the bistable range. However, the lowest mutation rate for bistability may be extremely small, and the highest may be impractically large. Increasing the strength of selection may eliminate bistability. For example, if the needle height is raised sufficiently in a GA that uses proportional selection, bistability will be avoided. However, for the needle-in-the-haystack fitness, GAs with binary tournament selection and truncation selection have been shown to have bistability. Sufficiently reducing the crossover rate or changing to a weaker recombination (such as reducing the number of crossover points for multi-point crossover) will avoid bistability.

3 A Unitiation-based Infinite Population Model for a GA

In this section we describe a "coarse-graining" of the standard GA infinite population model over unitation classes. To do this, we assume that the fitness function f is compatible with unitation, i. e., we assume that if $|x| = |y|$, then $f(x) = f(y)$. We also assume uniform crossover. In addition, we need to assume that the initial population satisfies the condition that all representatives of each equivalence class are equally likely. The following is a discussion of the need for this assumption.

3.1 Compatibility with Unitation

The infinite population model for the simple genetic algorithm is described in detail in [19]. In this model, populations are represented as real vectors indexed by the search space $\Omega = \{0,1\}^\ell$. If p is such a population, p_x denotes the fraction of individuals in the population which are the bit string x. The space of possible infinite populations is the simplex $\Lambda = \{p \in \mathbb{R}^{2^\ell} : p_x \geq 0 \text{ and } \sum p_x = 1\}$. (The finite populations of a particular size form a lattice of points within the simplex.) The model is defined by a function $\mathcal{G} : \Lambda \longrightarrow \Lambda$ where $\mathcal{G}(p)$ gives the expected next generation population resulting from applying the GA to population p.

The unitation of a bit string is the number of ones in the string. If x is a bit string, we will denote the number of ones in x by $|x|$. Let the unitation equivalence relation $\equiv$ on the search space be defined by $\{0,1\}^\ell$ by $x \equiv y$ iff $|x| = |y|$. (Note that there are $\ell+1$ equivalence classes corresponding to the unitations 0 to ℓ.) Let $\widetilde{\Omega} = \Omega/\equiv$ denote the set of unitation equivalence classes, and let $\widetilde{\Lambda}$ denote the space of populations from $\widetilde{\Omega}$. (In other words, $\widetilde{\Lambda} = \{p \in \mathbb{R}^{\ell+1} : p_u \geq 0 \text{ for all } u \in \widetilde{\Omega} \text{ and } \sum p_u = 1\}$.) There is a natural linear "projection" from Λ to $\widetilde{\Lambda}$ whose matrix Ξ is defined by

$$\Xi_{[y],x} = \begin{cases} 1 & \text{if } y \equiv x \\ 0 & \text{otherwise} \end{cases}$$

(Note that this definition does not depend on the equivalence class representative y of the equivalence class $[y]$.) For example, if $\ell = 2$,

$$\Xi = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Assuming a fitness function of unitation, our objective is to define an infinite population model $\widetilde{\mathcal{G}} : \widetilde{\Lambda} \longrightarrow \widetilde{\Lambda}$ that takes populations over unitation classes to populations over unitation classes. It would be nice if the map $\widetilde{\mathcal{G}}$ was compatible with $\equiv$ in the sense that the following diagram commuted:

$$\begin{array}{ccc} \Lambda & \xrightarrow{\mathcal{G}} & \Lambda \\ \Xi\big\downarrow & & \big\downarrow\Xi \\ \widetilde{\Lambda} & \xrightarrow{\widetilde{\mathcal{G}}} & \widetilde{\Lambda} \end{array}$$

(This definition of compatibility is given in Chapter 17 of [19] and is further elaborated in [15, 14].) The mutation map is compatible with $\equiv$ as is shown in [13]. However, crossover (even uniform crossover) is not compatible with $\equiv$. This can be seen by a simple example with string length $\ell = 2$. Consider the two population vectors $p = [0 \;\; 1 \;\; 0 \;\; 0]^T$ and $q = [0 \;\; 1/2 \;\; 1/2 \;\; 0]^T$. The first represents a population consisting entirely of the 01 string and the second represents a population with equal representation of the 01 and 10 strings. The map represented by Ξ maps these populations to the same population in $\widetilde{\Lambda}$. If crossover is applied to p, the result is p since crossing a string with itself always gives that string back. If uniform crossover with rate 1 is applied to q, the result is the population vector $[1/4 \;\; 1/4 \;\; 1/4 \;\; 1/4]$, and Ξ does not map this population vector to $\Xi(p)$. Thus, the diagram does not commute.

Let $\Psi = \{p \in \Lambda : |x| = |y| \text{ implies } p_x = p_y\}$. In other words, Ψ is the set of populations where all representatives of each unitation class are equally likely. The infinite population model equations for uniform crossover are invariant under a permutation of the string positions. (This can be seen by referring to the tensor-product formulation of infinite population model given in [3].) Mutation and proportional selection (assuming a fitness function of unitation) are compatible with unitation. If the initial population p is in Ψ, then $\mathcal{G}(p)$ will be in Ψ, and the trajectory starting at p will be in Ψ. We do obtain commutativity of the following diagram:

$$\begin{array}{ccc} \Psi & \xrightarrow{\mathcal{G}} & \Psi \\ \Xi\big\downarrow & & \big\downarrow\Xi \\ \widetilde{\Psi} & \xrightarrow{\widetilde{\mathcal{G}}} & \widetilde{\Psi} \end{array}$$

Thus, our model will coarse-grain the full infinite population model on Ψ. In other words, if both the initial population is in Ψ, then the trajectory of $\mathcal{G}$ will be in Ψ, and our model will correctly track this trajectory.

Even though the unitation infinite population model may accurately mirror the behavior of the full model on Ψ, this may be misleading since the trajectory of the full model in Ψ may be unstable in Λ. In other words, given a slight perturbation away from Ψ, the trajectory of the full model may diverge from Ψ, perhaps to converge to a fixed point which is far from Ψ. This can be illustrated by a 2-bit GA example. Let the fitness function f be given by $f(00) = f(11) = 1$ and $f(01) = f(10) = 2$. Suppose that we start the full infinite population model from the population vector $[1 \;\; 0 \;\; 0 \;\; 0]$ which corresponds to a population consisting solely of copies of the 00 string. Then the dynamical system trajectory will converge to the center of the simplex, namely $[1/4 \;\; 1/4 \;\; 1/4 \;\; 1/4]$. But the center of the simplex is an unstable fixed point, and given a small perturbation that breaks the symmetry between the frequencies of 01 and 10, the trajectory will either converge to the fixed point $[0 \;\; 1 \;\; 0 \;\; 0]$ or the fixed point $[0 \;\; 0 \;\; 1 \;\; 0]$, both of which are stable. A finite population GA is likely to move close to one or the other of these fixed points (and infrequently jump from a neighborhood of one fixed point to a neighborhood of the other).

3.2 Limitations of the infinite population model

It is conjectured (see [19]) that the infinite population model of a standard GA that uses mutation by a rate always converges to a fixed point. (Numerical examples of cyclic behavior were given in [21], but these were for a very nonstandard mutation.)

A standard way to apply a dynamical system model is to find the fixed points and their stability, and then to hope that the corresponding finite population GA will spend a lot of time near stable fixed points. There are some reasons why this may not happen.

The behavior of the infinite population model will be influenced by all points in the search space. Thus, if there are points or regions of the search space which a finite population GA is unlikely to sample, then these points or regions may have a substantial influence on the behavior of the infinite population model that is not reflected in the behavior of a typical run of the finite population GA.

An example is the needle-in-the-haystack fitness. Suppose that the string length is at least 20 and the population size is 1000 or less. In the infinite population model, there will be a selective pressure towards the needle string. If the mutation rate is not too high, there will be a stable fixed point "close" to the uniform population consisting of copies of the needle string. However, a finite population GA will be unlikely to ever sample the needle string in a reasonable number of generations, so the finite population GA will be doing random search in almost all runs.

Genetic drift is another reason why the infinite population model may not predict the behavior of a finite population GA. The infinite population model is a deterministic dynamical system, whereas the finite population GA is a stochastic system. We will expect that the finite population GA will periodically jump from the domain of attraction of one stable fixed point to the domain of attraction of another. For a sufficiently large population size and a sufficiently "attractive" fixed point, this will happen relatively infrequently, but for a small population size, this may happen more frequently. (For a small population size, it is more likely that the next population will be further from the expected population. In fact, theorem 3.5 of [19] shows that the variance of the distance of the next population from the expected next population is proportional to the inverse population size.)

Further, there may be unstable fixed points that strongly influence the behavior of both the infinite population model and the finite population GA. The infinite population model will take small steps (i. e. not move very far in each generation) when it is sufficiently close to an unstable fixed point. The same can be true for the finite population GA. For example, a fixed point may be unstable due to the attraction of a high-fitness point or region in the search space that the finite population GA is unlikely to sample, and thus the finite population GA may "stagnate" near the unstable fixed point until it does sample the high fitness point or region.

Despite the above limitations of the infinite population model, it still "usually" (in some imprecise sense) does a good job of predicting the behavior of a GA. Chapter 8 of [19] provides examples of how populations tend to stay close to stable fixed points. Rowe [13] provides a number of examples where the infinite population model gives very good explanations of the behavior of a finite population GA. However, it helps to keep the above limitations in mind when applying these models.

3.3 Determining the crossover probabilities for the unitation model

Assume that unitations u, v, and w are given, and that we want to find the probability that a string of unitation w results when strings of unitation u and v are crossed using uniform crossover with rate 1. (We will consider other rates later.) Let $r_1(u, v, w)$ denote this probability (where the subscript denotes the crossover rate).

First, we need to consider how the bits in the parent strings can correspond. Let x be a given string of unitation u. Let y be a randomly chosen string of unitation v, and let k denote the number of positions where a 1 in y corresponds to a 1 in x. The choice of y can be viewed as the following hyper-geometric experiment. We choose the 1 positions of y from the ℓ possible string positions, and we consider a "success" to be when the chosen position corresponds to a 1 in x. Then the probability of k successes is given by the hyper-geometric formula:

$$Pr[|x \otimes y| = k] = \frac{\binom{u}{k}\binom{\ell-u}{v-k}}{\binom{\ell}{v}}$$

Once k has been determined, the relationship between x and y with respect to uniform crossover is determined. (The order of bits is not significant for uniform crossover.) Without loss of generality, we can assume that the string positions have been ordered so that the 1-positions of x come first. Then we have the following picture of the relationship between the bits of x and the bits of y.

$$\begin{array}{rcccc} x = & \overbrace{1\ldots1}^{k} & \overbrace{1\ldots1}^{u-k} & \overbrace{0\ldots0}^{v-k} & \overbrace{0\ldots0}^{\ell-u-v+k} \\ y = & 1\ldots1 & 0\ldots0 & 1\ldots1 & 0\ldots0 \end{array} \tag{1}$$

By looking at these groups, we can determine the lower and upper bounds for possible values of k. Since $\ell - u - v + k \geq 0$, we have $k \geq u + v - \ell$. And k must be nonnegative. Clearly k cannot be greater than u, v, or w.

The next step is to consider the choice of a crossover mask that will give child string z of unitation w. Notice that the bits of the mask corresponding the to first and last groups of positions in (1) have no effect on the result of the crossover since x and y agree in these positions. Furthermore, the first group of positions will contribute exactly k one bits to the child, and the last group of positions will contribute no one bits to the child. Thus, the mask bits corresponding to the second and third groups of positions in (1) must be set to contribute $w - k$ one bits to the child z. There are $u + v - 2k$ positions in these these two groups, and there will be $\binom{u+v-2k}{w-k}$ settings of these mask bits that will produce $w - k$ one bits in the child. Thus, given k, the number of masks that will give a child with exactly w one bits is $2^{\ell-u-v+2k}\binom{u+v-2k}{w-k}$, and the probability of such a mask is $2^{-\ell}$ times this quantity.

Thus, the formula for the uniform crossover probability $r_1(u,v,w)$ from unitation classes u and v to unitation class w is given by:

$$r_1(u,v,w) = \sum_{k=\max(0,u+v-\ell)}^{\min(u,v,w)} \frac{\binom{u}{k}\binom{\ell-u}{v-k}\binom{u+v-2k}{w-k}}{\binom{\ell}{v}} 2^{-u-v+2k}$$

Next we consider a crossover rate of χ that may be less than 1. This means that in the operation of crossing parents u and v, crossover as described above is done with probability χ, or the child is u with probability $(1-\chi)/2$, or the child is v with probability $(1-\chi)/2$. Thus, the crossover rate χ probability of crossing u and v to get w is given by

$$r_\chi(u,v,w) = \begin{cases} \chi\, r_1(u,v,w) + \chi & \text{if } u = v = w \\ \chi\, r_1(u,v,w) + (1-\chi)/2 & \text{if } u = w \neq v \\ \chi\, r_1(u,v,w) + (1-\chi)/2 & \text{if } v = w \neq u \\ \chi\, r_1(u,v,w) & \text{if } u \neq w \text{ and } v \neq w \end{cases}$$

3.4 Determining mutation probabilities for the unitation model

To simplify, we will assume standard bitwise mutation using a rate μ. In other words, each bit of a string is mutated independently, and a bit is flipped with probability μ.

Given a string x of unitation u, we want to determine the probability that it will be mutated to a string y of unitation v. Clearly, this can happen if w zero bits are mutated to one bits, and $u-v+w$ one bits are mutated to zero bits where $0 \leq w \leq \min(\ell-u, v)$. The probability that this happens is

$$\sum_{w=0}^{\min(\ell-u,v)} \binom{\ell-u}{w}\binom{u}{u-v+w} \mu^{2w+u-v}(1-\mu)^{\ell-2w-u+v}$$

(An equivalent formula was derived in [13]).

4 Experimental Results

In this section we present results that illustrate bistability for string lengths of 50 and 100. We define the "sloping plateau" plateau fitness functions, and give results using both the unitation model and finite population GA runs.

The sloping plateau functions are defined by

$$P_{a,b,k}(x) = \begin{cases} a+b+1 & \text{if } |x| < k \\ b + (\ell - |x|)/\ell & \text{if } |x| \geq k \end{cases}$$

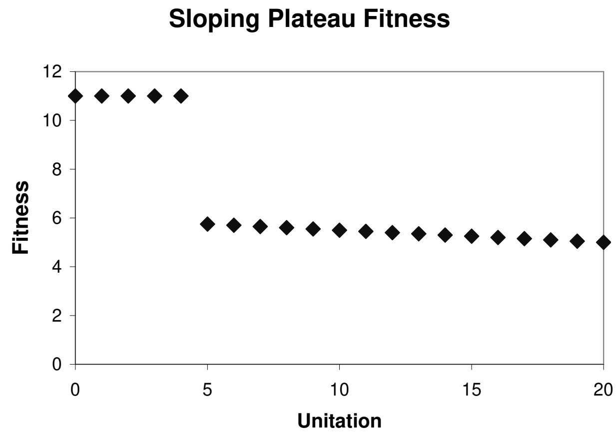

Figure 2: Sloping Plateau Fitness, $\ell = 20$, $k = 5$, $a = 5$, $b = 5$

The function has a "plateau" of fitness height $a + b + 1$ for strings whose unitation is less than k. The plateau is the global optimum. For unitation classes greater than or equal to k, there is a gradual linear slope up to the plateau. The value of b determines the slope: a larger value of b means that the slope up to the plateau is more gradual. The value of a determines the height of the plateau: a larger value of a means that the fitness plateau is higher above the non-plateau points. The sloping plateau fitness for $\ell = 20$, $k = 5$, $a = 5$, and $b = 5$ is plotted in Figure 2.

We are interested in showing situations where crossover is harmful to the performance of a GA. When there is bistability, there is a stable fixed point near the center of the simplex. The sloping plateau functions are designed to give weak but nonzero selection pressure near the center of the simplex. Once the plateau is reached, then there is reasonably strong selection pressure to stay on the plateau.

We could have added more fitness structure on the plateau, such as an isolated optimum on the plateau. However, this would not contribute to our discussion of bistability, and it might have led to problems with the correspondence between the infinite population model and finite population behavior.

4.1 Infinite population models for the sloping plateau

This section will explore the results of using the unitation infinite population model on plateau functions. The model was coded in Maple™. The correctness was checked by comparing to the previously coded full infinite population model.

Bistability for a particular setting of the parameters was checked by iterating the with-crossover model with two different starting populations: The model was started from the center of the simplex and from a population with all weight on the all-zeros string. The model was iterated for up to 2000 generations or until it converged (two successive populations had sum of absolute value difference less than 10^{-9}). If the ending populations from the two starting points were substantially different, we deduced bistability.

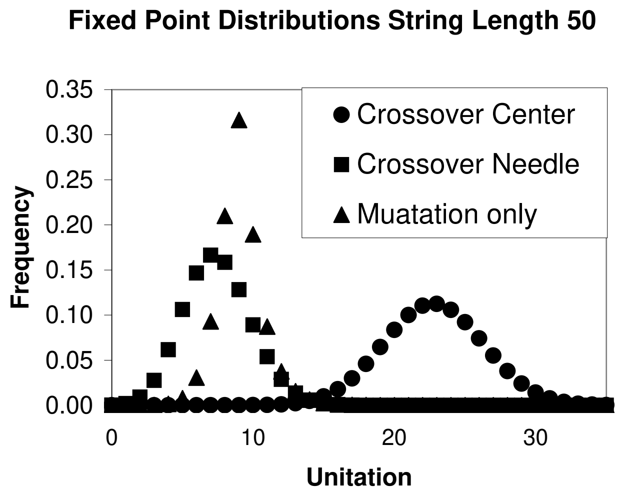

Figure 3: Fixed Point Distributions, $\ell = 50$, $k = 10$, $a = 5$, $b = 5$, $\mu = 0.01$

For $\ell = 50$, $a = b = 5$, there was bistability for all values of k from 1 to 14. For small values of k, mutation rates for bistability were low, and for large values of k, mutation rates for bistability were high. For example, for $k = 5$, there was bistability between $1/(8\ell)$ and $1/(2\ell)$, and for $k = 15$ there was bistability between $1/\ell$ and $3/(2\ell)$. The critical mutation rates for bistability decreased as k decreased.

We also ran the model without crossover on the same two starting populations for comparison. The Perron-Frobenius theorem implies that there can be only one stable fixed point for the model without crossover, and this is what we observed. The three fixed point distributions for the sloping plateau fitness with $\ell = 50$, $a = b = 5$, $k = 10$, and $\mu = 1/(2 * \ell) = 0.01$ are shown in Figure 3.

Boerlijst et al [2] gave results for an approximate plateau fitness function with string length 15. Their results were replicated in [12]. Their approximate plateau function had a fitness of 5 for unitation class 0, 4.8 for unitation class 1, 4.6 for unitation class 2, and 3.5 for the remaining unitation classes. They compared the needle fixed point distribution with mutation rate 0.011 and recombination rate 0.5 with the no recombination fixed point for the same mutation rate. They found that the needle fixed point had a higher frequency for unitation classes 0 and 1 and lower frequencies for unitation classes 6 and higher. Thus, the with-recombination distribution had less variance (was more compact) than the without-recombination distribtuion. In our figure 3, the with-recombination fixed-point distibution has a higher frequency for the low unitation classes (agreeing with the Boerlijst results), but the variance of the with-recomination distribution has larger variance and is less compact than the without-recombination distribution (which disagrees with the Boerlijst results).

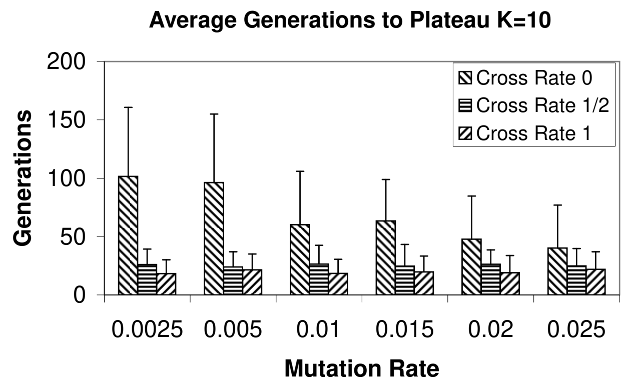

Figure 4: Generations to Optimum, $\ell = 50$, $k = 10$

4.2 Finite population results for sloping plateau

The results in this section are for a generational GA that uses proportional (roulette-wheel) selection, standard mutation, and uniform crossover either with crossover rates 0, $1/2$, and 1. Runs were made with mutation rates $\mu = 1/(8\ell)$, $1/(4\ell)$, $1/(2\ell)$, $3/(4\ell)$, $1/\ell$, and $5/(4\ell)$. The population size for all runs except those for Figure 9 was 10,000. Figures 4, 5, 7, and 8 all show 1-standard deviation error bars. The GA was implemented in Java by the first author and his students.

Experiments were done for $\ell = 50, 100$, $a = b = 5$. One set of runs was done to determine the number of generations necessary to reach the plateau, and another set of runs was done to count the number of plateau points and determine the average fitness after 200 or 300 generations.

4.2.1 Phase One: Finding the Plateau

The first set of experiments were designed to find the waiting time for the GA to hit the first plateau (or optimal) point with and without crossover for $\ell = 50, k = 10$, and various mutation rates. (The value $k = 10$ was chosen so that an initial random population of size 10,000 would not be likely to contain plateau points, but the GA would not take very long to get to the plateau.) Figure 4 shows that the with-crossover GA is reaching the plateau much more rapidly than the without-crossover GA. Each result is the average of 250 runs.

Second similar experiment with $\ell = 100, k = 28$ was run as well with very similar results. Figure 5 shows again that the uniform crossover operator is beneficial for first hitting time of the plateau. Each result is the average of 576 runs.

4.2.2 Phase Two: Converging on the Plateau

The next set of experiments explored the average fitness over a longer number of generations for the previous two plateau functions. Bistability induced by the crossover operator has a strong effect here. The results are very counter-intuitive. The previous section showed a drastically

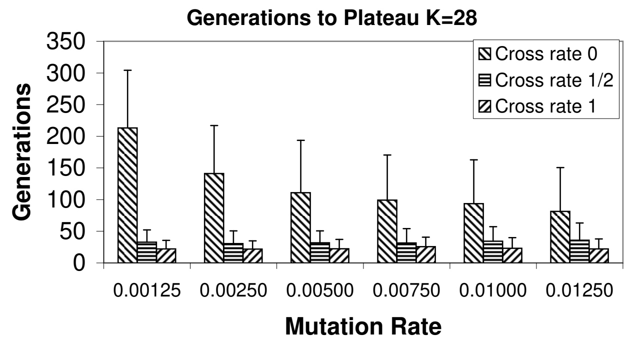

Figure 5: Generations to Optimum, $\ell = 100$, $k = 28$

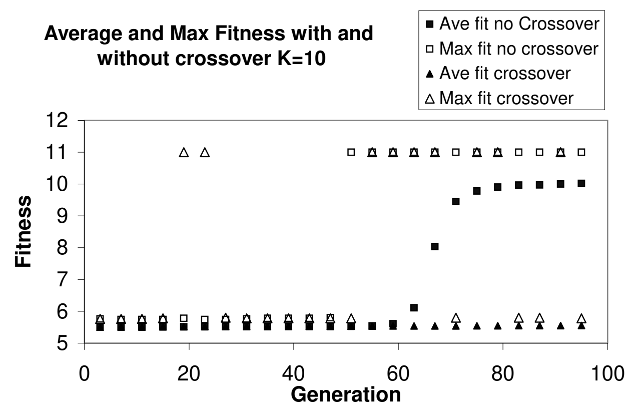

Figure 6: GA runs with and without crossover, $\ell = 50$, $k = 10$, population size 10000

shortened discovery time for optimal strings with crossover. Yet when the GA is allowed to run past that point, crossover inhibits the GA from accumulating many individuals on the plateau in the population. For all runs except those with very small mutation rates relative to $1/\ell$, the mutation-only GA outperformed two GAs with crossover by a significant margin. (The GA for the very small mutation rates was not bistable.) When the GA was bistable, the mutation-only GAs were better able to accumulate highly fit strings on the plateau, producing an increase in the average fitness of the population.

Figure 6 demonstrates that this is the typical situation. At generation 19, the with-crossover run first hits the plateau. From then on, there are intermittent copies of plateau strings, but they do not accumulate, so the average fitness stays below 6. The without-crossover GA does not hit the plateau until about generation 50, and then starting at about generation 60, it quickly accumulates plateau strings to bring the average fitness up to around 10. Many more generations could be shown that would look like the last generations on the graph. The with-crossover GA will continue to have intermittent plateau strings and low fitness, while the without-crossover GA will maintain high fitness and many plateau strings.

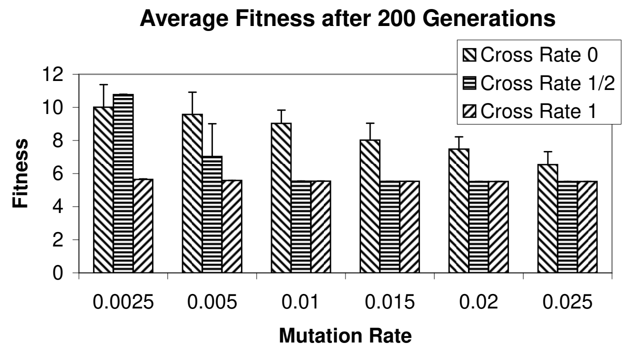

Figure 7: Average Fitness, $\ell = 50$, $k = 10$, 200 generations

4.2.3 Bistability and Finite Populations

The large finite population effects of bistability are well illustrated by considering the case where $\ell = 50$ and $k = 10$. The average fitness after 200 generations and various mutation rates are shown in Figure 7. Each result is the average of 100 runs, and error bars are shown. In looking at this figure, the reader should keep in mind that the fitness of plateau strings is $a + b + 1 = 11$ and the maximum fitness of non-plateau strings is $b + (\ell - k)/\ell = 5 + (50 - 10)/50 = 5.8$. For crossover rate 1, the four larger mutation rates, namely $1/\ell/2 = 0.010$, $3/\ell/4 = 0.015$, $1/\ell = 0.02$, and $5/\ell/4 = 0.025$ are all infinite-population bistable, and the center-of-simplex fixed point for $1/\ell/4 = 0.005$ is just barely unstable, while only the needle fixed point for $1/\ell/8 = 0.0025$ is stable.

For mutation rate $\mu/2 = 0.005$ the reader might compare to Figure 3. The center-of-simplex fixed point distribution in Figure 3 has a very small weight on plateau points, and that is what the finite population experiment shows with only 25 of 100 with-crossover runs ending with populations containing plateau points, and for those population with plateau points, there were at most 2 of these points. Figure 4 shows that all populations are likely to have hit the plateau by 200 generations, but they cannot maintain any substantial number of points on the plateau. On the other hand, mutation-only fixed point distribution of Figure 3 has a heavy weight on the plateau, and over 85% of these runs ended with populations with over 1000 plateau strings.

For the other bistable and nearly bistable mutation rates, the results are similar. For $\mu = 1/\ell/8 = 0.0025$, the GA with crossover rate $1/2$ was able to achieve as high average fitness as the no-crossover GA.

Results for $\ell = 50$, $k = 28$, and 300 generations are shown in Figure 8. For mutation rates $\mu = 1/(8\ell)$ and $1/(4\ell)$, the infinite population model was not bistable. For mutation rates $1/(2\ell)$, $3/(4\ell)$, $1/\ell$, and $5/(4\ell)$ the model was bistable. Each result is the average of 66 runs.

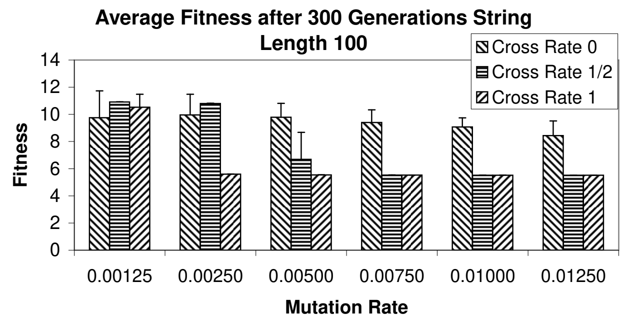

Figure 8: Average Fitness, $\ell = 100$, $k = 28$, 300 generations

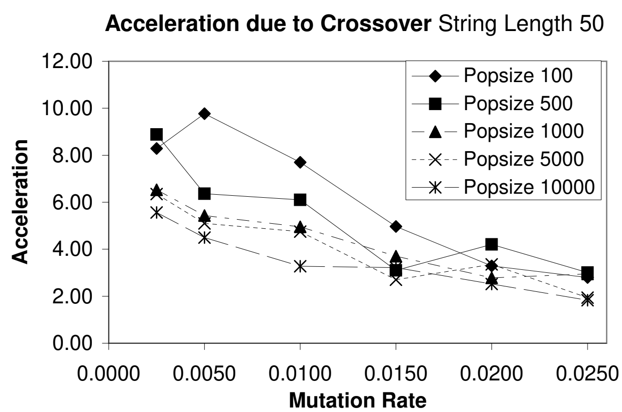

Figure 9: Acceleration of Crossover to Plateau, $\ell = 50$, $k = 10$

4.3 Acceleration due to Crossover

The speed up of the first hitting time of the plateau is similar to the acceleration due to crossover seen by Suzuki and Iwasa [18], except that here we are looking at the first hitting time of the plateau whereas Suzuki and Iwasa were looking at the time to domination of the needle string. As we have seen above, when there is bistability, it will take a very long time for the with-crossover GA to reach a a population with a substantial number of plateau points.

Figure 9 shows the acceleration of crossover for the first hitting time to the plateau as a function of both mutation rate and population size. Each point is based on the average of 50 GA runs. Define T_h as the number of generations until the GA population first includes a plateau point. This hitting time acceleration is defined by

$$A^h_{cross} = \frac{T_h|_{\text{without crossover}}}{T_h|_{\text{with crossover}}}$$

where "with crossover" means uniform crossover with rate 1.

5 Conclusion

Crossover in a GA can both very beneficial and very harmful. This paper gives fitness functions and GA parameters where the GA will get stuck and not move far from a random population in any practical time frame. In this situation, which we call bistability, the GA obviously has failed. The sloping plateau fitness functions that we have used to demonstrate this are unimodal, simply defined, and in our opinion not contrived. It seems that bistability happens in situations with relatively strong recombination, weak selection, and an appropriate level of mutation. Crossover can also accelerate the progress of the GA to wherever it is going. We have observed speedups of up to 10 by using crossover over the same GA without crossover, and Suzuki and Iwasa [18] have observed and predicted speedups of up to about 70 in a somewhat different situation. So if bistability can be avoided, crossover can be of great benefit.

What we mean by weak selection? If we had used a rank-based selection method on the sloping plateau fitness, the GA would move quickly up the slope and this would probably have eliminated bistability. However, previous results on the needle-in-the-haystack fitness show bistability with binary tournament and truncation selection. Thus, we would expect that these selection methods would give bistability with the plateau fitness. In this case, strength of selection for the infinite population model might be determined by the fraction of the search space that is on the plateau.

Our results and the Suzuki and Iwasa results support the concluding of Stephens and Waelbroeck [17, 16] that the recombination of long high-order schemata is very important when schema reconstruction dominates, i. e., when bistability is avoided.

References

[1] S. Baluja and R. Caruana. Removing the genetics from the standard genetic algorithm. In A. Prieditis and S. Russel, editors, *The Int. Conf. on Machine Learning 1995*, pages 38–46, San Mateo, CA, 1995. Morgan Kaufmann Publishers.

[2] M. C. Boerlijst, S. Bonhoeffer, and M. A. Nowak. Viral quasi-species and recombination. *Proc. Royal Society London B*, 263:1577–1584, 1996.

[3] C. Chryssomalakos and C. R. Stephens. What basis for genetic dynamics? In K. D. et al., editor, *GECCO 2004: Proceedings of the Genetic and Evolutionary Computation Conference*, pages I–1018–1029. Springer Verlag LNCS 3102, 2004.

[4] S. Forrest and M. Mitchell. What makes a problem hard for a genetic algorithm? some anamalous results and their explanation. *Machine Learning*, 13:285–319, 1993.

[5] D. E. Goldberg. *Genetic Algorithms in Search, Optimization & Machine Learning*. Addison Wesley, Reading, MA, 1989.

[6] D. E. Goldberg, B. Korb, and K. Deb. Messy genetic algorithms: Motivation, analysis, and first results. *Complex Systems*, 3:493–530, 1989.

[7] T. Jansen and I. Wegener. Real royal road functions - where crossover provably is essential. In L. Spector, E. D. Goodman, A. Wu, W. Langdon, H.-M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. H. Garzon, and E. Burke, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, pages 375–382, San Francisco, California, USA, 7-11 July 2001. Morgan Kaufmann.

[8] T. Jansen and I. Wegener. The analysis of evolutionary algorithms - a proof that crossover really can help. *Algorithmica*, 34(1):47–66, 2002.

[9] T. Jansen and I. Wegener. Real royal road functions — where crossover provably is essential. *Discrete applied mathematics*, 149:111–125, 2005.

[10] M. Mitchell, S. Forrest, and J. H. Holland. The royal road for genetic algorithms: Fitness landscapes and GA performance. In F. J. Varela and P. Bourgine, editors, *Proceedings of the First European Conference on Artificial Life. Toward a Practice of Autonomous Systems*, pages 243–254, Cambridge, MA, 1992. MIT Press.

[11] H. Mühlenbein. The equation for the response to selection and its use for prediction. *Evolutionary Computation*, 5(3):303–346, 1997.

[12] G. Ochoa and I. Harvey. Recombination and error thresholds in finite populations. In *Foundations of Genetic Algorithms 5*, pages 245–264, San Mateo, 1997. Morgan Kaufmann.

[13] J. E. Rowe. Population fixed-points for functions of unitation. In W. Banzhaf and C. Reeves, editors, *Foundations of genetic algorithms (FOGA-5)*, pages 60–84, San Mateo, 1999. Morgan Kaufmann.

[14] J. E. Rowe, M. D. Vose, and A. H. Wright. Coarse graining selection and mutation. In K. deJong, L. Schmitt, M. D. Vose, and A. H. Wright, editors, *Foundations of Genetic Algorithms 8*, Lecture Notes in Computer Science. Springer Verlag, 2005. In press.

[15] J. E. Rowe, M. D. Vose, and A. H. Wright. State aggregation and population dynamics in linear systems. *Artificial Life*, 11(4), 2005.

[16] C. R. Stephens and H. Waelbroeck. Schemata evolution and building blocks. *Evolutionary Computation*, 7(2):109–124, 1999.

[17] C. R. Stephens, H. Waelbroeck, and R. Aguirre. Schemata as building blocks: does size matter. In *Foundations of Genetic Algorithms 5*, pages 117–133, San Mateo, 1997. Morgan Kaufmann.

[18] H. Suzuki and Y. Iwasa. Crossover accelerates evolution in gas with a babel-like fitness landscape: Mathematical analyses. *Evolutionary Computation*, 7(3):275–310, 1999.

[19] M. D. Vose. *The Simple Genetic Algorithm: Foundations and Theory*. MIT Press, Cambridge, MA, 1999.

[20] M. D. Vose and A. H. Wright. Stability of vertex fixed points and applications. In L. D. Whitley and M. D. Vose, editors, *Foundations of genetic algorithms 3*, pages 103–113, San Mateo, 1995. Morgan Kaufmann.

[21] A. H. Wright and G. L. Bidwell. A search for counterexamples to two conjectures on the simple genetic algorithm. In *Foundations of genetic algorithms 4*, pages 73–84, San Mateo, 1997. Morgan Kaufmann.

[22] A. H. Wright and G. Cripe. Bistability of the needle function in the presence of truncation selection. In *GECCO 2004: Proceedings of the Genetic and Evolutionary Computation Conference*. Springer Verlag, 2004.

[23] A. H. Wright, J. E. Rowe, and J. R. Neil. Analysis of the simple genetic algorithm on the single-peak and double-peak landscapes. In *Proceedings of the Congress on Evolutionary Computation (CEC) 2002*, pages 214–219. IEEE Press, 2002.

[24] A. H. Wright, J. E. Rowe, R. Poli, and C. R. Stephens. A fixed point analysis of A gene pool GA with mutation. In W. B. Langdon and others, editors, *GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference*, pages 642–649. Morgan Kaufmann Publishers, 2002.

[25] A. H. Wright, J. E. Rowe, R. Poli, and C. R. Stephens. Bistability in a gene pool GA with mutation. In *Foundations of genetic algorithms (FOGA-7)*, San Mateo, 2003. Morgan Kaufmann. http://www.cs.umt.edu/u/wright/pubs.htm.