The Gambler's Ruin Problem, Genetic Algorithms, and the Sizing of Populations

George Harik
Illinois Genetic Algorithms Laboratory
University of Illinois
Urbana, IL 61801 USA
gharik@illigal.ge.uiuc.edu

Erick Cantú-Paz
Illinois Genetic Algorithms Laboratory
University of Illinois
Urbana, IL 61801 USA
cantupaz@illigal.ge.uiuc.edu

David E. Goldberg
Illinois Genetic Algorithms Laboratory
University of Illinois
Urbana, IL 61801 USA
deg@illigal.ge.uiuc.edu

Brad L. Miller
I2 Technologies
Boston, MA 02139 USA
bmiller@technologist.com

Abstract

This paper presents a model to predict the convergence quality of genetic algorithms based on the size of the population. The model is based on an analogy between selection in GAs and one-dimensional random walks. Using the solution to a classic random walk problem—the gambler's ruin—the model naturally incorporates previous knowledge about the initial supply of building blocks (BBs) and correct selection of the best BB over its competitors. The result is an equation that relates the size of the population with the desired quality of the solution, as well as the problem size and difficulty. The accuracy of the model is verified with experiments using additively decomposable functions of varying difficulty. The paper demonstrates how to adjust the model to account for noise present in the fitness evaluation and for different tournament sizes.

Keywords

Population size, noise, decision making, building block supply.

1 Introduction

The question of how to choose an adequate population size for a particular domain is difficult and has puzzled practitioners for a long time. If the population is too small, it is not likely that the genetic algorithm (GA) will find a good solution for the problem at hand. Therefore, it may appear reasonable that to find solutions of high quality, the size of the populations must be increased as much as possible. However, if the population is too large, the GA will waste time processing unnecessary individuals, and this may result in unacceptably slow performance. The problem consists of finding a population size that is large enough to permit a correct exploration of the search space without wasting computational resources. The goal of this study is to provide a practical answer to the problem of finding suitable population sizes for particular domains.

Hard questions are better approached using a divide-and-conquer strategy, and the population sizing issue is no exception. This paper identifies two factors which depend on the population size and that influence the quality of the solutions that the GA may reach: the

©1999 by the Massachusetts Institute of Technology

initial supply of building blocks (BBs), and the selection of the best BB over its competitors. One way of approaching the population-sizing problem would be to study these two factors in isolation, and indeed useful models may be found. However, considering only one factor may result in predictions that are too inaccurate to be practical.

The approach used in this study incorporates previous knowledge about the initial supply and the correct selection of BBs in a natural way. The critical point in the modeling is to create an analogy between selection in GAs and simple one-dimensional random walks. Once the analogy is established, the analysis may use well-known results from the literature on stochastic processes to make progress on the population sizing issue. The result is a model that accurately predicts the quality of the solution reached when the GA converges, and from which a population sizing equation can be derived.

The paper begins with a review of the facetwise decomposition that guides this work and a discussion of some previous studies on population sizing. Section 3 revisits in detail a model describing the probability of choosing correctly between two competing individuals. This pairwise decision probability is then used as an integral part of the population sizing model presented in section 4. Section 5 verifies the accuracy of the model with results of experiments using additively decomposable functions of varying difficulty. Next, section 6 discusses how the model is extended to account for explicit noise in the fitness of the individuals. Section 7 further extends the model to consider different tournament sizes. The paper concludes with a summary of results and suggestions for extending this work.

2 Background

Over the years, researchers and practitioners have noticed that population size is strongly related to the convergence quality of GAs and the duration of their run. Large populations usually result in better solutions, but also in increasing computational costs. The challenge is to find adequate population sizes, so that the GA can be designed to reach the desired solution as fast as possible.

Unfortunately, the question of how large the population has to be to reach a solution of certain quality has been largely ignored; there are only a handful of studies that guide users to choose adequate population sizes (Goldberg, 1989b; Goldberg et al., 1992). This section reviews some of these studies, but first describes the decomposition that guides our study of GAs.

2.1 Decomposing the Problem

Despite their operational simplicity, GAs are complex non-linear algorithms. To have any hope of understanding and designing GAs we need to approach them similarly to other difficult engineering tasks: decompose the problem into tractable sub-problems, solve the sub-problems, and integrate the partial solutions. For some time, we have used the following decomposition as a guide in our study of GAs (Goldberg and Liepins, 1991; Goldberg et al., 1992):

1. Know what the GA is processing: building blocks (BBs).
2. Solve problems tractable by BBs.
3. Supply enough BBs in the initial population.

4. Ensure the growth of necessary BBs.
5. Mix the BBs properly.
6. Decide well among competing BBs.

We restrict the notion of building blocks to the minimal-order schemata that contribute to the global optimum (Thierens and Goldberg, 1993). In this view, when crossover juxtaposes two BBs of order k at a particular string it does not lead to a single BB of order $2k$ but instead to two separate BBs. In addition, we assume that the only source of BBs is the random initialization of the population: mutation and crossover do not create or destroy too many BBs.

This study addresses two of the six points listed above: the initial supply and the decision process between competing BBs. The model described later in this paper incorporates these two issues in what elsewhere (Goldberg, 1996) has been called a *little model*; it does not attempt to describe the effect of all possible parameters on the search. Instead, the model focuses only on the supply and decision issues and describes many practical relations between them. We found that, although the model excludes the effects of mixing and growth of BBs, the result is extremely accurate and can be used as a guideline to design faster and more reliable GAs.

Previous estimates of adequate population sizes fall into two categories: models concerned with the initial supply of BBs and models that involve decision-making between competing BBs. The remainder of this section reviews some of these previous studies.

2.2 Supply Models

A basic premise in this study is that GAs work by propagating and combining BBs. However, before selection and recombination can act on the BBs, the GA must have an adequate supply of them. When BBs are abundant, it is likely that the GA will choose and combine them correctly; conversely, when BBs are scarce the chances of the GA converging to a good solution are small.

The first supply model simply considers the number of BBs present in the initial random population. The probability that a single building block of size k is generated randomly is $1/2^k$ for binary domains, and therefore the initial supply of BBs can be estimated as

$$x_0 = \frac{n}{2^k} \tag{1}$$

This simple supply equation suggests that domains with short BBs, and thus with more BBs in the initial population, need smaller population sizes than domains with longer BBs. A later section presents empirical results using functions with BBs of different lengths to corroborate this notion.

Another way of relating the size of the population with the expected performance of the GA is to count the number of schemata processed by the GA. Holland (1975) estimated that a randomly initialized population of size n contains $O(n^3)$ schemata. Holland used the term *implicit parallelism* to denote this fact, and it has become one of the common arguments on why GAs work well. Goldberg (1989a) rederived this estimate in two steps: (1) compute the number of schemata in one string, and then (2) multiply it by the population size.

The number of schemata of length l_s or less in one random binary string of length l is $2^{l_s-1}(l - l_s + 1)$. The schema length l_s is chosen such that the schemata survive crossover and mutation with a given constant probability. It is likely that low-order schemata will be duplicated in large populations, so to avoid overestimating the number of schemata, pick a population size $n = 2^{l_s/2}$ so that on average half of the schemata are of higher order than $l_s/2$ and half are of smaller order. Counting only the higher order ones gives a lower bound on the number of schemata in the population as $n_s \geq n(l - l_s + 1)2^{l_s-2}$. Since $n = 2^{l_s/2}$ this becomes $n_s = \frac{(l-l_s+1)n^3}{4}$, which is $O(n^3)$.

In a different study, Goldberg (1989b) computed the expected number of unique schemata in a random population and used this quantity together with an estimate of the convergence time to find the optimal population size that maximizes the rate of schema processing. Goldberg considered serial and parallel fitness evaluations, and his results suggest that high schema turnover is promoted with small populations in serial GAs and with large populations in parallel GAs.

More recently, Mühlenbein and Schlierkamp-Voosen (1994) derived an expression for the minimum population size needed to converge to the optimum with high probability. Their analysis is based on additive fitness functions, and their study focuses on the simplest function of this type: the onemax, which we also use in our investigation. They conjectured that the optimal population size depends on the initial supply of the desired alleles, the size of the problem, and the selection intensity. Our study also considers the effect of these variables on the population size. Later, Cvetković and Mühlenbein (1994) empirically determined that for the onemax the population size is directly proportional to the square root of the size of the problem, and it is inversely proportional to the square root of the proportion of correct alleles in the initial population. Our results are also based on additive fitness functions and are consistent with their experimental fit on the problem size but indicate that, in general, the population size is inversely proportional to the proportion of correct BBs present initially (not to the square root).

2.3 Decision Models

The second aspect of population sizing involves selecting better partial solutions. Holland (1973, 1975) recognized that the issue of choosing between BBs (and not between complete strings) can be recast as a two-armed bandit problem, a well-known problem in statistical decision theory. The problem consists of choosing the arm with the highest payoff of a two-armed slot machine at the same time the information needed to make this decision is collected. This classic problem is a concrete example of the tradeoff between exploring the sample space and exploiting the information already gathered. Holland's work assumes an idealization of the GA as a cluster of interconnected 2-armed bandits, so his result relating the expected loss and the number of trials can be directly applied to schema processing. Although Holland's calculations are based on an idealization, his results give an optimistic bound on the allocation of trials on a real GA.

De Jong (1975) recognized the importance of noise in the decision process and proposed an estimate of the population size based on the signal and noise characteristics of the problem. Unfortunately, he did not use his estimate in the remainder of his groundbreaking empirical study and the result was unverified and ignored by many.

Goldberg and Rudnick (1991) gave the first population sizing estimate based on the variance of fitness. Later, Goldberg et al. (1992) developed a conservative bound on the

Figure 1: Two competing building blocks of order four.

convergence quality of GAs. Their model is based on deciding correctly between the best BB in a partition and its closest competitor, while the decision process is clouded by collateral noise coming from the other partitions.

The result of that investigation is a population sizing equation which for binary alphabets and ignoring external sources of noise is

$$n = 2c(\alpha)2^k m' \frac{\sigma_{bb}^2}{d^2}, \tag{2}$$

where $c(\alpha)$ is the square of the ordinate of a unit normal distribution where the probability equals α; α is the probability of failure; k is the order of the BB; m' is one less than the number of BBs in a string (m); σ_{bb}^2 is the root mean square (RMS) fitness variance of the partition that is being considered; and d is the fitness difference between the best and second best BBs.

Their model conservatively approximates the behavior of the GA by considering that if the wrong BBs were selected in the first generation, the GA would be unable to recover from the error. Likewise, if the decisions were correct in the first generation, the model assumes that the GA would converge to the right solution. Our study is a direct extension of the work by Goldberg, Deb, and Clark. The main difference between our model and theirs is that we do not approximate the behavior of the GA by the outcome of the first generation.

3 Deciding Well Between Two BBs

The role of selection in GAs is to decide which individuals survive to form the next generation. The selection mechanism is supposed to choose those individuals having the correct BBs and to eliminate the others, but sometimes the wrong individuals are chosen. To understand why this may occur this section reviews the calculations by Goldberg et al. (1992) of the probability of deciding well between an individual with the best BB and another individual with the second best BB. Their idea is to focus on one partition and to consider the fitness contributions from the other partitions as noise that interferes in the decision process.

Consider a competition between an individual i_1 that containing the optimal BB in a partition H_1, and an individual i_2 with the second best BB H_2. This is illustrated in Figure 1. Ideally, the selection mechanism should choose i_1, but there is a chance of erroneously choosing i_2. This may occur because the contribution of the other partitions

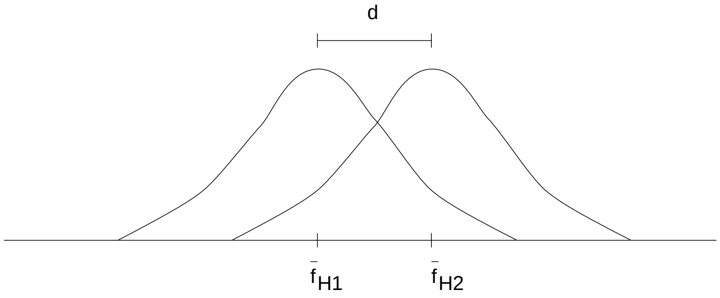


Figure 2: Fitness distributions for two competing individuals.

to the fitness of i_2 may be large enough to exceed the fitness advantage that i_1 has for possessing the best BB. The probability of deciding correctly between these two individuals is the probability that the fitness of i_1 (f_1) is greater than the fitness of i_2 (f_2) or equivalently the probability that $f_1 - f_2 > 0$.

Figure 2 illustrates the distributions of the fitness of individuals containing H_1 and H_2. The distance between the mean fitness of individuals with H_1 $(\overline{f_{H_1}})$ and the mean fitness of individuals with H_2 $(\overline{f_{H_2}})$ is denoted by d. Assuming that the fitness is an additive function of the fitness contributions of all the partitions in the problem, we may consider the distributions of f_1 and f_2 to be normal by the central limit theorem.

Since the fitness distributions of f_1 and f_2 are normal, the distribution of $f_1 - f_2$ is itself normal and has known properties: the mean is the difference of the individual means, and the variance is the sum of the individual variances. Therefore,

$$f_1 - f_2 \sim N(\overline{f_{H_1}} - \overline{f_{H_2}}, \sigma^2_{H_1} + \sigma^2_{H_2})$$

Substituting $d = \overline{f_{H_1}} - \overline{f_{H_2}}$ in the expression above and normalizing, the probability of making the correct decision on a single trial is

$$p = \Phi\left(\frac{d}{\sqrt{\sigma^2_{H_1} + \sigma^2_{H_2}}}\right), \tag{3}$$

where Φ is the cumulative distribution function for a normal distribution with zero mean and unit standard distribution.

To calculate $\sigma^2_{H_1}$ and $\sigma^2_{H_2}$, we follow Goldberg et al. (1992) and assume that the fitness function F is the sum of m independent subfunctions F_i, each of the same size k of the most deceptive partition. Now, the overall fitness variance may be calculated as

$$\sigma^2_F = \sum_{i=1}^{m} \sigma^2_{F_i}$$

For domains where the m partitions are uniformly scaled (equally weighted), the average root mean square (RMS) BB variance (denoted by σ^2_{bb}) is simply $\sigma^2_{F_i}$. In this case,

Figure 3: The bounded one-dimensional space of the gambler's ruin problem.

the total noise coming from the $m' = m - 1$ partitions that are not competing directly is $\sigma^2 = m'\sigma_{bb}^2$. Therefore, the probability of making the right choice in a single trial in a problem with m independent and equally-scaled partitions becomes

$$p = \Phi\left(\frac{d}{\sqrt{2m'}\sigma_{bb}}\right) \tag{4}$$

Goldberg et al. (1992) used this probability to create the first model relating the size of the population with the quality of decisions. Their model showed how to incorporate the effect of the collateral noise into the population-sizing question, and their paper describes how to estimate the parameters necessary to calculate p. In the next section, the knowledge about noise and decision-making at the BB level is unified with the estimate of the initial supply of BBs.

4 The Gambler's Ruin Model

The gambler's ruin problem is a classical example of random walks, which are mathematical tools used to predict the outcome of certain stochastic processes. The most basic random walk deals with a particle that moves randomly on a one-dimensional space. The probability that the particle moves to the left or to the right is known, and it remains constant for the entire experiment. The size of the step is also constant, and sometimes the movement of the particle is restricted by placing barriers at some points in the space. For our purposes, we consider a one-dimensional space bounded by two absorbing barriers that capture the particle once it reaches them.

In the gambler's ruin problem, the capital of a gambler is represented by the position, x, of a particle on a one-dimensional space, as depicted in Figure 3. Initially, the particle is positioned at $x_0 = a$ where a represents the gambler's starting capital. The gambler plays against an opponent that has an initial capital of $n - a$, and there are absorbing boundaries at $x = 0$ (representing bankruptcy) and at $x = n$ (representing winning all the opponent's money). At each step in the game, the gambler has a chance p of increasing his capital by one unit and a probability $q = 1 - p$ of loosing one unit. The object of the game is to reach the boundary at $x = n$, and the probability of success depends on the initial capital and on the probability of winning a particular trial.

The analogy between selection in GAs and the gambler's ruin problem appears naturally if we assume that partitions are independent and we concentrate on only one of them. The particle's position on the one-dimensional space, x, represents the number of copies of the correct BBs in the population. The absorbing barriers at $x = 0$ and $x = n$ represent convergence to the wrong and right solutions, respectively. The initial position of the particle, x_0, is the expected number of copies of the BB in a randomly initialized population, which in a binary domain and considering BBs of order k is $x_0 = \frac{n}{2^k}$.

There are a number of assumptions that we need to make to use the gambler's ruin problem to predict the quality of the solutions of the GA. First, the gambler's ruin model considers that decisions in a GA occur one at a time until all the n individuals in its population converge to the same value. In other words, in the model there is no explicit notion of generations, and the outcome of each decision is to win or lose one copy of the optimal BB. Assuming conservatively that all competitions occur between strings that represent the best and the second best BBs in a partition, the probability of gaining a copy of the global BB is given by the correct decision-making probability p, which was calculated in the previous section for additively-decomposable fitness functions (see Equation 4).

The calculation of p implicitly assumes that the GA uses pairwise tournament selection (two strings compete), but adjustments for other selection schemes are possible, as we see in a later section. The analogy between GAs and the gambler's ruin problem also assumes that the only source of BBs is the random initialization of the population. This assumption implies that mutation and crossover do not create or destroy significant numbers of BBs. The boundaries of the random walk are absorbing; this means that once a partition contains n copies of the correct BB it cannot lose one, and likewise, when the correct BB disappears from a partition there is no way of recovering it. We recognize that this is a simplification, but experimental results suggest that is it a reasonable one.

As we discussed above, the GA succeeds when there are n copies of the correct BB in the partition of interest. A well-known result in the random walk literature is that the probability that the particle will eventually be captured by the absorbing barrier at $x = n$ is:

$$P_{bb} = \frac{1 - \left(\frac{q}{p}\right)^{x_0}}{1 - \left(\frac{q}{p}\right)^{n}}, \tag{5}$$

where $q = 1 - p$ is the probability of losing a copy of the BB in a particular competition (Feller, 1966). From this equation, it is relatively easy to find an expression for the population size. First, note $p > 1 - p$ (because the mean fitness of the best BB is greater than the mean fitness of the second best), and x_0 is usually small compared to the population size. Therefore, for increasing values of n, the denominator in Equation 5 approaches 1 very quickly and can be ignored in the calculations. Substituting the initial supply of BBs ($x_0 = n/2^k$), P_{bb} may be approximated as

$$P_{bb} \approx 1 - \left(\frac{1-p}{p}\right)^{n/2^k} \tag{6}$$

Since we assume that the BBs are independent of each other, the expected number of partitions with the correct BB at the end of a run is $E(BBs) = mP_{bb}$. Assuming that we are interested in finding a solution with an average of $\hat{Q}$ BBs correct, we can solve $P_{bb} = \frac{\hat{Q}}{m}$ for n to obtain the following population sizing equation:

$$n = \frac{2^k \ln(\alpha)}{\ln\left(\frac{1-p}{p}\right)}, \tag{7}$$

where $\alpha = 1 - \frac{\hat{Q}}{m}$ is the probability of GA failure. To observe more clearly the relations between the population size and the domain-dependent variables involved, we may expand p and write the last equation in terms of the signal (d), the noise (σ_{bb}), and the number of partitions in the problem (m). First, approximate p using the first two terms of the power series expansion for the normal distribution as:

$$p = \frac{1}{2} + \frac{1}{\sqrt{2\pi}} z,$$

where $z = d/(\sigma_{bb}\sqrt{2m'})$ (Abramowitz and Stegun, 1972). Substituting this approximation for p into Equation 7 results in

$$n = 2^k \ln(\alpha)/\ln\left(\frac{1 - \frac{z\sqrt{2}}{\sqrt{\pi}}}{1 + \frac{z\sqrt{2}}{\sqrt{\pi}}}\right) \tag{8}$$

Since z tends to be a small number, $\ln(1 \pm \frac{z\sqrt{2}}{\sqrt{\pi}})$ may be approximated as $\pm\frac{z\sqrt{2}}{\sqrt{\pi}}$. Using these approximations and substituting the value of z into the equation above gives

$$n = -2^{k-1} \ln(\alpha) \frac{\sigma_{bb}\sqrt{\pi m'}}{d} \tag{9}$$

This rough approximation makes more clear the relations between some of the variables that determine when a problem is harder than others. It quantifies many intuitive notions that practitioners have about problem difficulty for GAs. For example, problems with long BBs (large k) are more difficult to solve than problems with short BBs, because long BBs are scarcer in a randomly initialized population. The equation shows that the required population size is inversely proportional to the signal-to-noise ratio. Problems with a high variability are hard because it is difficult to detect the signal coming from the good solutions when the interference from not-so-good solutions is high. Longer problems (larger m) are more difficult than problems with a few partitions because there are more sources of noise. However, the GA scales very well to the problem size; the equation shows that the required population grows with the square root of the size of the problem.

5 Experimental Verification

This section verifies that the gambler's ruin model accurately predicts the quality of the solutions reached by simple GAs. The experiments reported in this section use test functions of varying difficulty. First, a simple one-max function is used, and later the experiments use fully deceptive trap functions. The population sizes required to solve the test problems vary from a few tens to a few thousands, demonstrating that the predictions of the model scale well to problem difficulty.

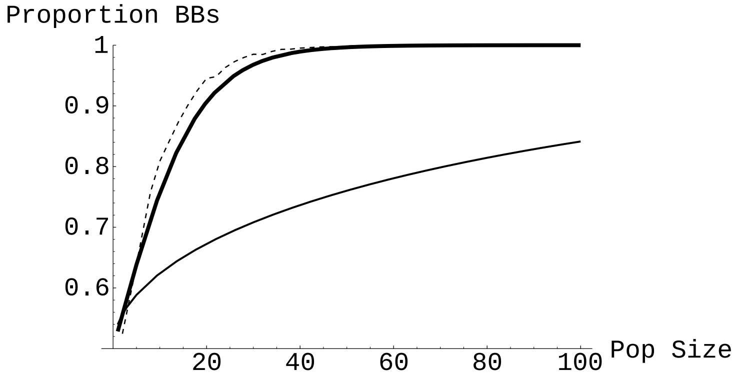


Figure 4: Experimental and theoretical results of the proportion of correct BBs on a 100-bit one max function. The prediction of the gambler's ruin model (Equation 5) is in bold, the experimental results are the dotted line, and the previous decision-based model is the thin line.

All the results in this paper are the average over 100 independent runs of a simple generational GA. The GA uses pairwise tournament selection without replacement. The crossover operator was chosen according to the order of the BBs of each problem and is specified in each subsection below. In all the experiments, the mutation probability is set to zero, because the model considers that the only source of diversity is the initial random population. Also, mutation would make it possible to escape the absorbing boundaries of the random walk. Each run was terminated when the population had converged completely (which is possible because the mutation rate is zero), and we report the percentage of partitions that converged to the correct value. The theoretical predictions of the gambler's ruin model were calculated using Equation 5.

5.1 One-max

The one-max problem is one of the most frequently used fitness function in genetic algorithms research because of its simplicity. The fitness of an individual is equal to the number of bits set to one in its chromosome. This is an easy problem for GAs since there is no isolation or deception and the BBs are short. The supply of BBs is no problem either, because a randomly initialized population has on average 50% of the correct BBs.

In this function the order of the BBs is $k = 1$. The fitness difference is $d = 1$ because a correct partition contributes one to the fitness and an incorrect partition contributes nothing. The variance may be calculated as $\sigma_{bb}^2 = (1 - 0)/4 = 0.25$ (see Goldberg et al. (1992) for a discussion on how to approximate or bound σ_{bb}^2). The first set of experiments considers strings with 100 bits, so $m = 100$. Substituting these values into Equation 4 gives the probability of choosing correctly between two competing BBs as $p = 0.5565$.

Since the length of the BBs is one, crossover cannot disrupt them, and uniform crossover was chosen for this function as it mixes BBs quite effectively. The probability of exchanging each bit was set to 0.5.

In Figure 4 the bold line is the theoretical prediction of the gambler's ruin model (Equation 5) and the dotted line is the experimental results for a 100-bit function. The thin

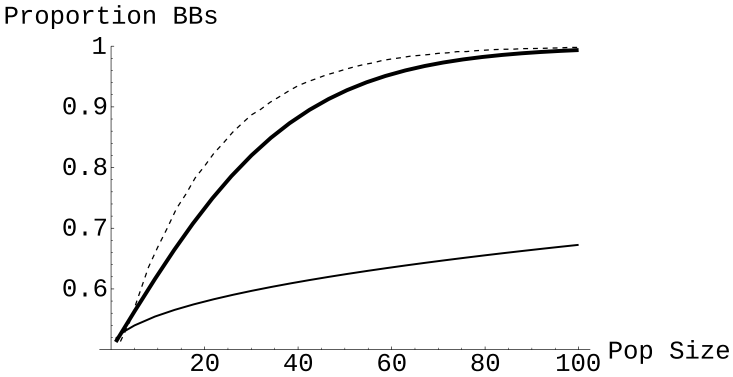


Figure 5: Experimental and theoretical results of the proportion of correct BBs on a 500-bit one-max function. The bold line is the prediction of the gambler's ruin model (Equation 5), the dotted line is the experimental results, and the thin line is the previous decision-only model.

line in the figure is the theoretical prediction of the population sizing model of Goldberg et al. (1992). The results of a second set of experiments with a 500-bit onemax problem ($p = 0.5252$) are shown in Figure 5.

The gambler's ruin model predicts the outcome of the experiments for the 100 and 500-bit functions quite accurately. However, in the 500-bit function the match is not as close as in the 100-bit case. The reason for this small discrepancy may be that the theory only considers one partition at a time and that decisions for one partition are independent of all the others. In order to achieve this independence, crossover must distribute BBs completely at random across all the individuals in the population. The goal of crossover in this case is to smooth the distribution of BBs in the different alleles to avoid hitchhiking. However, it would not be practical to reach this perfect distribution, because many rounds of crossover would be necessary in every generation.

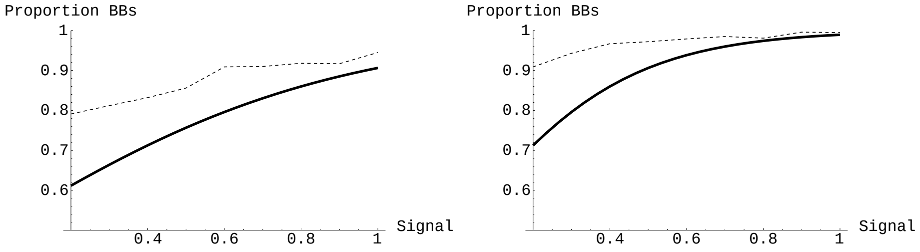
5.2 Scaled Problems

The next set of experiments tests the predictions of the model on scaled subfunctions. The fitness function is similar to a one-max except that the contribution of every tenth bit to the fitness is reduced. In particular, the first scaled function is

$$F_{sc} = \sum_{i=0}^{m} c_i x_i,$$

where $x_i \in \{0, 1\}$, and $c_i = 0.8$ when $i \in \{0, 10, 20, ..., 90\}$ and $c_i = 1$ otherwise.

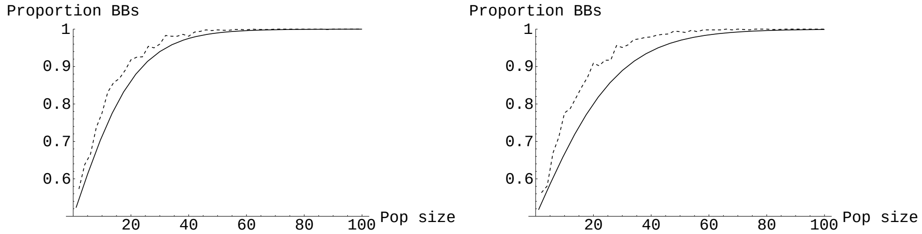
The signal of the scaled partitions is smaller than before ($d = 0.8$), and the noise is reduced slightly ($\sigma_{bb}^2 = 0.245$). The reduction of the signal makes it harder to decide correctly between the right and wrong BBs, and therefore larger populations are required to reach the correct solution (in this case $p = 0.5457$). The parameters for the experiments are the same as for the one-max functions, but the results reported are the proportion of badly scaled bits that converged correctly. The stronger bits converge more easily, and including them in the results would paint an overly optimistic picture.



(a) $n = 20$

(b) $n = 40$

Figure 6: Experimental and theoretical results on scaled 100-bit one-max functions varying the signal from 0.2 to 1. The predictions (Equation 5) are plotted in bold and the experiments with a dotted line. The population size remained constant in each experiment at the values indicated.



(a) $d = 0.8$

(b) $d = 0.6$

Figure 7: Experimental and theoretical results of the proportion of correct BBs on two scaled 100-bit one-max problems. The predictions of the random walk model (Equation 5) are plotted in bold and the experimental results with a dotted line.

The second scaled function is similar to the first, but the signal of every tenth bit is reduced further to $d = 0.6$ (and p becomes 0.5346). The predictions and the empirical results for the two scaled functions are displayed in Figure 6. In both cases the model correctly predicts larger population sizes, but it is more conservative than with the uniformly-scaled one-max.

A probable reason is that the estimate of variance might be too high in the scaled cases. In the GA, the more salient BBs (those that are not scaled down) converge first. Therefore, when the competitions between the scaled BBs begin, the noise coming from the other partitions is much lower because most of them must have converged already.

The effect of reducing the signal even further may be observed in Figure 7. In this case, the population size is maintained constant at $n = 20$ and $n = 40$ while the signal of the scaled bits ranges from 0.2 to 1. The model correctly predicts that the proportion of partitions that converge correctly decreases as the signal becomes weaker. The model is more accurate as the signal increases, but it is always conservative and may be used

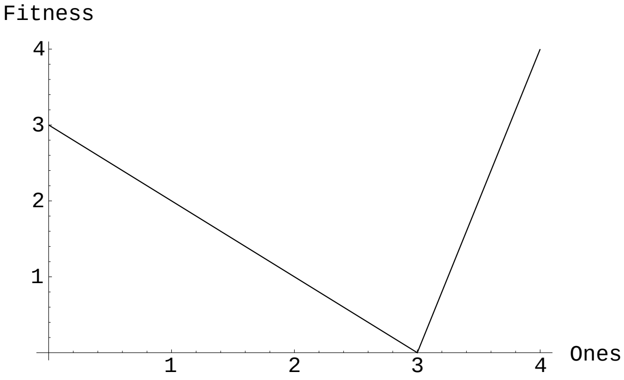


Figure 8: A 4-bit deceptive function of unity.

confidently to determine the population size required to find the desired solution.

5.3 Deceptive Functions

The next three sets of experiments use deceptive trap functions. Fully deceptive trap functions are used in many studies of GAs because their difficulty is well understood and they can be regulated easily (Deb and Goldberg, 1993).

Trap functions are hard for traditional optimizers because they will tend to climb to the deceptive peak, but GAs with properly sized populations can solve them satisfactorily. We expect to use larger population sizes than before to solve these functions for two reasons: (1) the BBs are much scarcer in the initial population because they are longer; and (2) the signal to noise ratio is lower, making the decision between the best and the second best BBs more difficult.

To solve the trap functions, tight linkage was used (i.e., the bits that define the trap function are next to each other in the chromosome), although there are algorithms such as the messy GA (Goldberg and Bridges, 1990) and its relatives (Goldberg et al., 1993; Kargupta, 1996; Harik and Goldberg, 1996) that are able to find tight linkages autonomously.

The first deceptive test function is based on the 4-bit trap function depicted in Figure 8, which was also used by Goldberg et al. (1992) in their study of population sizing. As in the one-max, the value of this function depends on the number of bits set to one, but in this case the fitness increases with more bits set to zero until it reaches a local (deceptive) optimum. The global maximum of the function occurs precisely at the opposite extreme where all four bits are set to one, so an algorithm cannot use any partial information to find it. The signal difference d (the difference between the global and the deceptive maxima) is 1 and the fitness variance (σ_{bb}^2) is 1.215. The test function is formed by concatenating $m = 20$ copies of the trap function for a total string length of 80 bits. The probability of making the right decision between two individuals with the best and the second best BBs is $p = 0.5585$. The experiments with this function use two-point crossover to avoid the excessive disruption that uniform crossover would cause on the longer BBs. The crossover probability is set to 1.0 and, as before, there is no mutation.

Figure 9 presents the prediction of the percentage of BBs correct at the end of the run

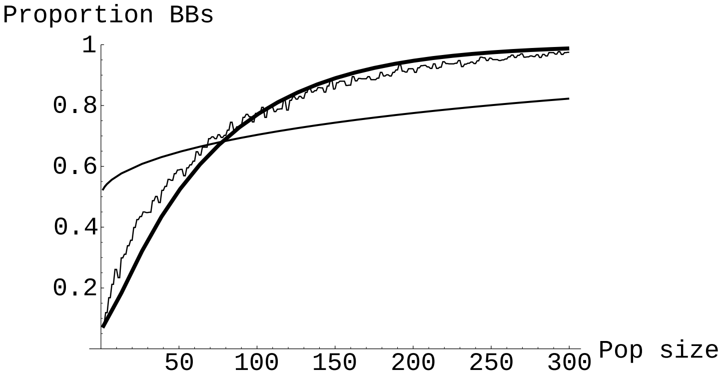


Figure 9: Theoretical predictions and experimental results for a 4-bit deceptive function with 20 BBs. The model (Equation 5, solid bold line) approximates well the experimental results (dotted). The solid thin line is the previous decision-making model.

along with the results from the experiments. The bold line is the prediction of the gambler's ruin model and the thin line is the prediction of the previous decision-only model. Note that the convergence quality for small population sizes is dominated by the initial supply of BBs and, as expected, the previous model is not accurate in this region.

The second deceptive test function is formed by concatenating $m = 10$ copies of an 8-bit fully deceptive trap. The 8-bit trap is similar to the 4-bit used above with a signal difference $d = 1$, but the fitness variance is $\sigma^2_{bb} = 2.1804$. The higher variance and longer BBs make this function more difficult than the 4-bit problem, so larger populations are expected in this case. One-point crossover was used for this function because the longer BBs are more likely to be disrupted by crossover. The crossover probability was again set to 1.0 and there is no mutation.

Figure 10 presents the results for this problem. As in previous cases, the random walk model (in bold) approximates the experimental results well. As expected, the previous population-sizing model (thin line) gives a very conservative estimate of convergence quality, especially for small population sizes.

5.4 Overlapping Subfunctions

The next test problem is an additive function of $m = 20$ 3-bit fully deceptive subfunctions defined over overlapping substrings,

$$F = \sum_{i=1}^{m} f(u_{2i}),$$

where u_{2i} is the number of ones in the three-bit long substring that starts in position $2i$ and $i = 0, 2, .., 19$. By defining the function in this manner, the last bit of each group of three overlaps with the first bit of the next group. There are $l = 41$ total bits in the strings. Each

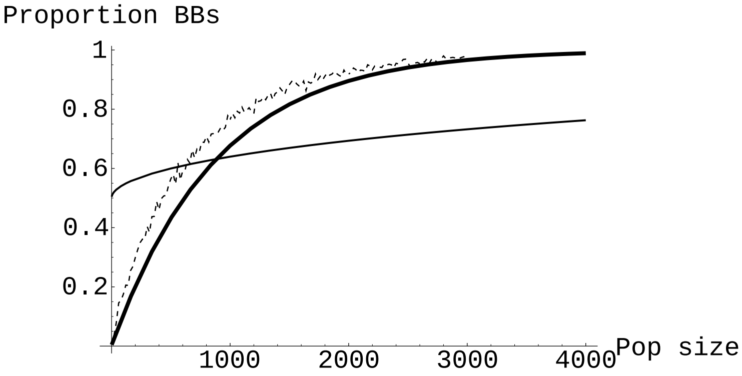


Figure 10: Theoretical predictions and experimental results (dotted line) for a 8-bit deceptive function with 10 BBs. The predictions of the gambler's ruin model (Equation 5) are in bold, and the previous decision model is plotted with a thin line.

three-bit trap function is defined as:

$$f(u_{2i}) = \begin{cases} 2 & \text{if } u_{2i}\text{=0} \\ 1 & \text{if } u_{2i}\text{=1} \\ 0 & \text{if } u_{2i}\text{=2} \\ 3 & \text{otherwise.} \end{cases}$$

The distance between the best and second best BBs is $d = 3 - 2 = 1$, and the BB variance is $\sigma_{bb}^2 = 1$. Substituting these values in Equation 4 results in $p = 0.5644$.

Figure 11 presents the experimental results along with the predictions of Equation 5. The experimental results are the average of 100 independent runs using pairwise tournament selection, two-point crossover with probability 1 and no mutation. Despite the overlap of BBs, the model accurately predicts the proportion of BBs at the end of the run.

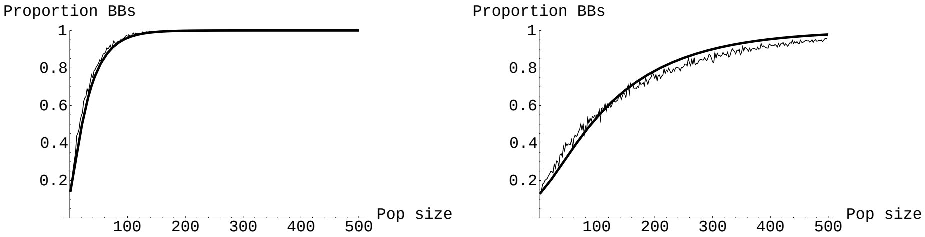
To make the function more deceptive we changed the subfunction f to

$$f(u_{2i}) = \begin{cases} 0.9 & \text{if } u_{2i}\text{=0} \\ 0.8 & \text{if } u_{2i}\text{=1} \\ 0 & \text{if } u_{2i}\text{=2} \\ 1 & \text{otherwise.} \end{cases}$$

In this case, the distance between the best and second best BB is $d = 1 - 0.9 = 0.1$, and the BB variance is $\sigma_{bb}^2 = 0.1773$. Substituting these values in Equation 4 results in $p = 0.5153$. The experimental results (using the same setup as above) are plotted in Figure 11, and show good agreement with the theory.

6 Noise and Population Sizing

Genetic algorithms are increasingly used to solve problems where the fitness of an individual cannot be determined exactly due to noise in the evaluation function. The noise may come



(a) $d = 1$

(b) $d = 0.1$

Figure 11: Experimental and theoretical results of the proportion of correct BBs on a function with overlapping 3-bit traps. The solid line plots the predictions of the random walk model (Equation 5) and the experimental results are plotted with the dotted line.

from an inherently noisy domain or from a noisy approximation of an excessively expensive fitness function. This section examines how this explicit fitness noise affects the size of the population.

Following Goldberg et al. (1992) the noisy fitness F' of an individual may be modeled as

$$F' = F + N,$$

where F is the true fitness of an individual and N is the noise present in the evaluation. The effect of the added noise is to increase the fitness variance of the population, making it more difficult to choose correctly between two competing individuals. Therefore, the one-on-one decision-making probability (Equation 4) has to be modified to include the effect of explicit noise; then, it may be used to find the required population size as was done in Section 4.

Assuming that the noise is normally distributed as $N(0, \sigma_N^2)$, the fitness variance becomes $\sigma_F^2 + \sigma_N^2$. Therefore, the probability of choosing correctly between an individual with the optimal building block and an individual with the second best BB in a noisy environment is

$$p = \Phi\left(\frac{d}{\sqrt{\sigma_{H_1}^2 + \sigma_{H_2}^2 + 2\sigma_N^2}}\right)$$

In the case of uniformly scaled problems, this becomes (Miller, 1997):

$$p = \Phi\left(\frac{d}{\sqrt{2(m'\sigma_{bb}^2 + \sigma_N^2)}}\right) \tag{10}$$

Using this form of p and the same procedure used to obtain Equation 9 results in the following population sizing equation for noisy domains:

$$n = -2^{k-1} \ln(\alpha) \frac{\sqrt{\pi m' \sigma_{bb}^2 + \sigma_N^2}}{d} \tag{11}$$

The one-max domain was used to test the predictions of the gambler's ruin model (Equation 5) using the decision probability given by Equation 10. The experiments considered a 100-bit problem and used the same GA as the previous one-max experiments (pairwise tournament selection, uniform crossover and no mutation).

For this function, the fitness variance is $\sigma_F^2 = m\sigma_{bb}^2 = 25$, and the experiments were run for σ_N^2 values of $\sigma_F^2 \times (0, 1, 2)$. Figure 12 displays the average proportion of partitions that converge correctly (taken over 100 independent runs at each population size) for each noise level. Note that the experiments with $\sigma_F = 0$ are equivalent to those presented in Figure 4, and are included here to facilitate comparisons.

The effect of increasing the noise in the quality of convergence may be observed very clearly in Figure 13. In this set of experiments, the population size was held constant at $n = 20$ and $n = 40$ while the noise increased. The gambler's ruin model correctly predicts that additional noise causes the GA to reach solutions of lower quality, and although the predictions are not exact, the model is conservative and may be used as a guideline to calculate the required population size.

7 Effect of the Selection Type

Besides the population size, an important factor in the convergence quality of GAs is the selection scheme. After all, the selection mechanism is the part of the GA making the decisions we have discussed in the previous sections. The goal of any selection method is to favor the proliferation of good individuals in the population. However, selection methods differ in how much pressure they exert on the population.

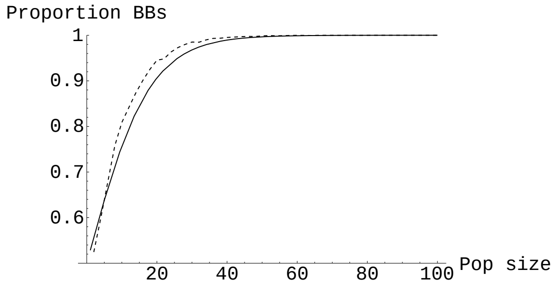
Mühlenbein and Schlierkamp-Voosen (1993) introduced the concept of *selection intensity* to measure the pressure of selection schemes. The selection intensity is the expected average fitness of the population after selection. The selection pressure of tournament selection increases as the tournament size, s, becomes larger. Only tournament selection is considered here, but the results can be extrapolated to other selection methods with known constant selection intensities (see Bäck, 1994; Miller and Goldberg, 1996). A different analysis would be necessary for schemes such as proportional selection that do not have a constant selection pressure.

Assume conservatively that the correct BB competes against $s - 1$ copies of the second best BB. As larger tournaments are considered, the probability of making the wrong decision increases proportionately to s; thus, we approximate the probability of making the right decision as $1/s$. In reality, this probability is higher than $1/s$ since a tournament might involve more than one copy of the best BB—especially as the run progresses and the proportion of correct BBs increases. However, $1/s$ is a good initial approximation as the experiments suggest.

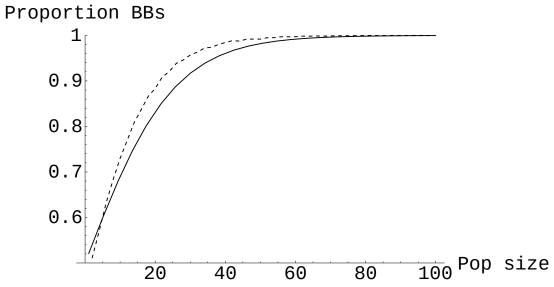
The increasing difficulty of decision-making as the tournament size increases can be accounted for as a contraction in the signal that the GA is trying to detect. Setting the new signal to

$$d' = d + \Phi^{-1}\left(\frac{1}{s}\right)\sigma_{bb}, \tag{12}$$

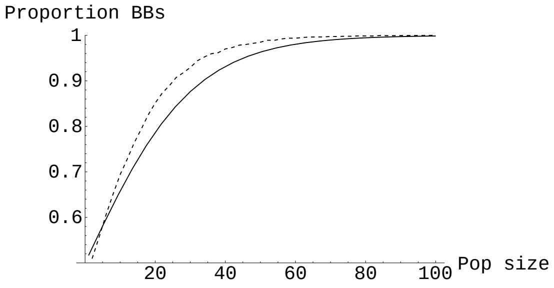
where $\Phi^{-1}(1/s)$ is the ordinate of a unit normal distribution where the CDF equals $1/s$,



(a) No noise

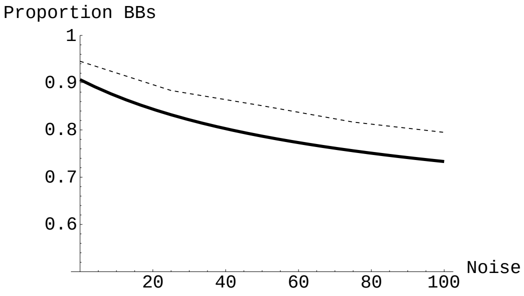


(b) $\sigma_N^2 = 25$

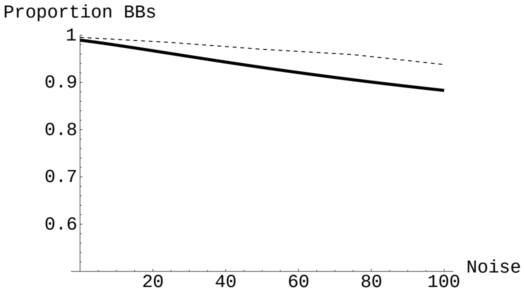


(c) $\sigma_N^2 = 50$

Figure 12: Theoretical (solid lines) and experimental (dotted lines) results for 100-bit one-max functions with different noise levels. The noise is constant for each set of experiments, and the population size ranges from 2 to 100.



(a) $n = 20$



(b) $n = 40$

Figure 13: Theoretical (solid lines) and experimental (dotted lines) results for 100-bit one-max functions varying the noise levels from $\sigma_N^2 = 0$ to 100. The population size remained constant for each set of experiments at the values indicated.

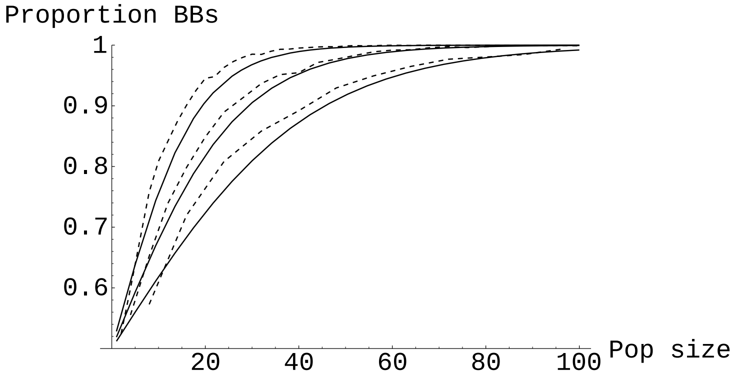


Figure 14: Predictions and experimental results for a 100-bit one-max function varying the selection intensity. From left to right: $s = 2, 4, 8$.

we can compute a new probability of deciding well using d' instead of d in Equation 4.

Experiments were performed using a 100-bit one-max function and tournament sizes of 2, 4, and 8. The experimental results are plotted in Figure 14 with dotted lines along with the theoretical predictions obtained with Equation 5 using the appropriate value of p for each case. The leftmost plot corresponds to a tournament size $s = 2$, the next to $s = 4$, and the rightmost to $s = 8$. Once again, the model is a good predictor of the proportion of BBs correct at the end of the run.

8 Extensions

The gambler's ruin model integrates two facets that influence the convergence of GAs: the initial supply of BBs (x_0) and correct decision making between competing BBs (p). The success of the integration of two facetwise models is evidence that the divide-and-conquer strategy can have good results. Following this notion of integrating small models, the results of this paper may be extended to include other facets of GA efficiency. Previous sections hinted at some possible extensions; here they are treated in more detail and other possibilities are presented.

Probably the most important extension to the model is to incorporate the effects of crossover and mutation. In its current state, the model assumes that all the BBs in a string are independent of each other and that the correct BBs are evenly distributed along the whole population. This assumption implies that BBs are mixed properly and that strings have on average the same number of correct BBs. In reality, some strings may contain more copies of the correct BB and they may reproduce faster, impeding the crossover operator to mix the BBs evenly.

Furthermore, crossover presents a tradeoff: we want to distribute BBs throughout the population, but we do not want to disrupt those BBs that have been found already. More aggressive crossover operators (e.g., 5-point or uniform) would increase the chances that BBs mix, but more BBs would be disrupted. We need to model both of these effects to find a balance between mixing and disruption.

Another fundamental assumption of the model is that crossover and mutation do not create or destroy many BBs. Extending the model to consider that BBs can be created or destroyed by the operators implies that the boundaries of the gambler's ruin random walk would not be absorbing anymore. However, the idea of modeling selection in the GA as a biased random walk can still be used. Instead of calculating the probability of absorption, we may calculate the probability of hitting the barrier that denotes success ($x = n$) before hitting the barrier that represents failure ($x = 0$).

Another extension is the sizing of populations for parallel GAs. Parallel implementations are important because they open opportunities to solve harder or larger problems than is possible with single processors. Work is underway to quantify the effect of some relevant parameters of parallel GAs on the size of the populations, and there are already some results for cases that bound the interactions between multiple populations (Cantú-Paz and Goldberg, 1997).

9 Summary and Conclusions

This paper presented a model that permits users to determine an adequate population size to reach a solution of a particular quality. The model is based on an analogy between the selection mechanism of GAs and a biased random walk. The position of a particle on the bounded one-dimensional space of the random walk represents the number of copies of the correct BBs present in the population. The probability that the particle will be absorbed by the boundaries of the space is well known, and is used to derive an equation that relates the population size with the required solution quality and several domain-dependent parameters.

The accuracy of the model was verified with experiments using test problems that ranged from the very simple to the moderately hard. The results showed that the predictions of the model are accurate for the functions tested, and that the predictions scale well with the difficulty of the domain.

The investigation was influenced by a facetwise decomposition of GAs. This method of integrating small models opens opportunities to develop models for other facets of GA performance. For instance, the basic model was extended to consider non-uniformly scaled fitness functions, explicit noise in the fitness evaluation and different selection schemes. Other extensions may require different adjustments, but we have shown that the facetwise decomposition of tasks that guided this investigation is appropriate.

Acknowledgments

The work was sponsored by the Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grant number F49620-97-1-0050. Research funding for this project was also provided by a grant from the US Army Research Laboratory Program, Cooperative Agreement DAAL01-96-2-003. The US Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies and endorsements, either expressed or implied, of the Air Force Office of Scientific Research or the US Government.

Erick Cantú-Paz was partially supported by a Fulbright-García Robles Fellowship.

Brad Miller was supported by NASA under grant number NGT-9-4.

References

Abramowitz, M. and Stegun, I. (1972). *Handbook of mathematical functions with formulas, graphs, and mathematical tables*. Dover Publications, Mineola, New York.

Bäck, T. (1994). Selective pressure in evolutionary algorithms: A characterization of selection mechanisms. In *Proceedings of the First IEEE Conference on Evolutionary Computation*, Volume 1, pages 57–62, IEEE Service Center, Piscataway, New Jersey.

Cantú-Paz, E. and Goldberg, D. E. (1997). Modeling idealized bounding cases of parallel genetic algorithms. In Koza, J., Deb, K., Dorigo, M., Fogel, D., Garzon, M., Iba, H. and Riolo, R., editors, *Genetic Programming 1997: Proceedings of the Second Annual Conference*, pages 353–361, Morgan Kaufmann, San Francisco, California.

Cvetković, D. and Mühlenbein, H. (1994). *The optimal population size for uniform crossover and truncation selection*. Technical Report No. GMD AS GA 94-11. German National Reseach Center for Computer Science (GMD), Germany.

De Jong, K. A. (1975). *An analysis of the behavior of a class of genetic adaptive systems*. Doctoral dissertation, (University Microfilms No. 76-9381), Department of Computer and Communication Sciences, University of Michigan, Ann Arbor, Michigan.

Deb, K. and Goldberg, D. E. (1993). Analyzing deception in trap functions. In Whitley, L. D., editor, *Foundations of Genetic Algorithms 2*, pages 93–108, Morgan Kaufmann, San Mateo, California.

Feller, W. (1966). *An introduction to probability theory and its applications(2nd ed.)*, Volume 1, John Wiley and Sons, New York, New York.

Goldberg, D. E. (1989a). *Genetic algorithms in search, optimization, and machine learning*. Addison-Wesley, Reading, Massachusetts.

Goldberg, D. E. (1989b). Sizing populations for serial and parallel genetic algorithms. In Schaffer, J. D., editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 70–79, Morgan Kaufmann, San Mateo, California, (Also TCGA Report 88004).

Goldberg, D. E. (1996). The design of innovating machines: Lessons from genetic algorithms. *Computational Methods in Applied Sciences '96*, pages 100–104.

Goldberg, D. E. and Liepins, G. (1991). *A tutorial on genetic algorithm theory*. A presentation given at the Fourth International Conference on Genetic Algorithms, University of California at San Diego, La Jolla, California.

Goldberg, D. E. and Bridges, C. L. (1990). An analysis of a reordering operator on a GA-hard problem. *Biological Cybernetics*, 62:397–405, (Also TCGA Report No. 88005).

Goldberg, D. E., Deb, K. and Clark, J. H. (1992). Genetic algorithms, noise, and the sizing of populations. *Complex Systems*, 6:333–362.

Goldberg, D. E., Deb, K., Kargupta, H. and Harik, G. (1993). Rapid, accurate optimization of difficult problems using fast messy genetic algorithms. In Forrest, S., editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 56–64, Morgan Kaufmann, San Mateo, California.

Goldberg, D. E. and Rudnick, M. (1991). Genetic algorithms and the variance of fitness. *Complex Systems*, 5(3):265–278. (Also IlliGAL Report No. 91001).

Harik, G. R. and Goldberg, D. E. (1996). Learning linkage. In Belew, R. K. and Vose, M. D., editors, *Foundations of Genetic Algorithms 4*, pages 247–262, Morgan Kaufmann, San Francisco, California.

Holland, J. H. (1973). Genetic algorithms and the optimal allocation of trials. *SIAM Journal on Computing*, 2(2):88–105.

Holland, J. H. (1975). *Adaptation in natural and artificial systems*. University of Michigan Press, Ann Arbor, Michigan.

Kargupta, H. (1996). The gene expression messy genetic algorithm. In Bäck, T., Kitano, H. and Michalewicz, Z., editors, *Proceedings of 1996 IEEE International Conference on Evolutionary Computation*, pages 814–819, IEEE Service Center, Piscataway, New Jersey.

Miller, B. L. (1997). *Noise, sampling, and efficient genetic algorithms*. Doctoral dissertation, Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, Illinois. Also available as IlliGAL tech report No. 97001.

Miller, B. L. and Goldberg, D. E. (1996). Genetic algorithms, selection schemes, and the varying effects of noise. *Evolutionary Computation*, 4(2):113-131.

Mühlenbein, H. and Schlierkamp-Voosen, D. (1993). Predictive models for the breeder genetic algorithm: I. Continuous parameter optimization. *Evolutionary Computation*, 1(1):25–49.

Mühlenbein, H. and Schlierkamp-Voosen, D. (1994). The science of breeding and its application to the breeder genetic algorithm (BGA). *Evolutionary Computation*, 1(4):335–360.

Thierens, D. and Goldberg, D. E. (1993). Mixing in genetic algorithms. In Forrest, S., editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 38–45, Morgan Kaufmann, San Mateo, California.