The Effects of Static Fitness Function Noise Upon the Performance of Genetic Algorithms

(Appears in the *Proceedings of the 7th Joint Conference on Information Sciences*, pp. 275-278).

Alexander Grushin
agrushin@cs.umd.edu
University of Maryland
Department of Computer Science
A. V. Williams Building
College Park, MD 20742

October 7, 2004

This scholarly paper is submitted in fulfillment of a partial requirement for the completion of the Master's Degree without Thesis in Computer Science.

Abstract: This paper discusses the performance of the genetic algorithm under an imperfect, albeit static fitness function, where the fitness of any solution is modified by the addition of a noise term $N(0, \sigma^2)$. According to a developed definition, which discounts the algorithm-independent effects of noise upon the distribution of fitness values, the decrease in performance is found to be $o(\sigma)$, even though the effect of the noise-related deterioration in the structure of the problem is $\Omega(\sigma)$. It is inferred that while this deterioration creates difficulty for the algorithm in finding solutions with a high observed fitness, the solutions it does find are less likely to have a much lower true fitness. In part, this appears to be a consequence of the genetic algorithm's ability to actively avoid solutions that have a high observed fitness as a result of a large noise term.

Introduction

Evolutionary algorithms are believed to be effective at tackling problems where the fitness of any given solution cannot be determined exactly. This belief has been shaped by their successful application towards a number of such problems, an example being the generation of C code via genetic programming [4], where each solution is a program, whose correctness can be tested only on a limited number of cases. However, studies where the phenomenon of noise-resistance is rigorously examined are fewer in number.

The principles behind evolutionary algorithms in general, and genetic algorithms specifically, have some basis upon theories of evolution. Given this notion, many researchers [2] and users of evolutionary algorithms point out that Nature does not deal with perfect fitness functions: quite often, chance occurrences can kill strong organisms, while allowing weaker ones to live on, and successfully reproduce. A computational view suggests that by virtue of having access to an entire population of solutions, an evolutionary algorithm holds a high amount of knowledge of the search space, and this knowledge endows it with noise-resistant properties. In this vein, [2] suggests that increasing the population size can improve the performance of an evolutionary algorithm when it is faced with a noisy fitness function. Along with [1], it explores the effects of noise upon the solution of the sphere model by genetic and evolution strategy algorithms, and hypothesizes that noise has similar effects on all evolutionary algorithms. Experimental studies have been complemented by theoretical excursions, such as [7], which presents a version of Holland's Schema Theorem, adapted to situations where stochastic effects are present in the fitness function.

The goal of this paper lies in examining the effect of fitness function imperfections upon the performance of genetic algorithms. It is hypothesized that the genetic algorithm will exhibit a "graceful" degradation as the quality of the fitness function deteriorates, and is capable of

acceptable performance for moderate imperfections in the function. The paper will present and discuss the result of an experiment, which was performed in an effort to provide the words "acceptable" and "moderate" with a more concrete meaning. Furthermore, an attempt will be made to infer the causes of the degradation.

Theoretical Considerations

In order to assess the impact that noise has on the performance of genetic algorithms, an appropriate, numerical definition of performance must be specified. It is tempting to define the performance $P_{\Pi}(g)$ of a genetic algorithm (or a group of genetic algorithms) on a problem Π to be the expected value of the fitness $f_{best}(g)$ of a best solution found within g generations:

$$P_{\Pi}(g) = E[f_{best}(g)] \quad (1)$$

However, when the fitness function is imperfect, the definition's utility is questionable. The addition of noise transforms Π into a different problem Θ, which is effectively presented to the algorithm. If (1) is adopted, one cannot, in general, expect any algorithm that has no knowledge of Π to have the same performance when it is presented with Θ as when it is presented with Π. For this reason, a definition of performance should not be based on *algorithm-independent* effects of noise, which are a direct consequence of the convolution of noise with the fitness density distribution of a problem, which is defined as:

$$\phi_{\Pi}(x) = |\{\pi \in \Pi : \pi \text{ has fitness } x\}| \quad (2)$$

A cumulative density function can also be defined:

$$F_{\Pi}(x) = \Sigma_{(\forall t \leq x)} [\phi_{\Pi}(t)] \quad (3)$$

A reasonable assumption is made that for higher values of x, values of $\phi_{\Pi}(x)$ generally decrease as x increases; in other words, better solutions tend to be less common. For the purpose of this discussion, statements about solutions to some problem Π will be conditional on the premise that their fitness values are high enough for this assumption to hold. Two effects of noise can now be described. The first effect, which will be referred to as the *relaxation effect*, causes $\phi_{\Theta}(x)$ to be "flatter" and more elongated than $\phi_{\Pi}(x)$. As a result, if x and y' are the fitness values of solutions in Π and Θ, respectively, where $F_{\Pi}(x) = F_{\Theta}(y')$, then $E[y'] > E[x]$ will hold.

An opposite tendency, termed the *inflation effect*, makes it likely that a solution's observed fitness is greater than its true fitness. The sloping density assumption suggests that in the neighborhood of the observed fitness value y', the majority of solutions to Π have a fitness that is less than y'. Consequently, if y is the true fitness that corresponds to y' (i.e., the fitness of the corresponding solution in Π), then $E[y'] > E[y]$ will hold.

Maintaining the definitions for x, y, and y', and selecting some small, non-negative constant ε, the notions of relaxation $R_{(\Pi,\Theta)}(f)$ and inflation $I_{(\Pi,\Theta)}(f)$ can now be provided with a numerical definition:

$$R_{(\Pi,\Theta)}(f) = E[(y'-x) \mid (f-\varepsilon \leq x \leq f+\varepsilon)] \quad (4)$$

$$I_{(\Pi,\Theta)}(f) = E[(y'-y) \mid (f-\varepsilon \leq y' \leq f+\varepsilon)] \quad (5)$$

Thus, the following is expected:

$$y \approx x + R_{(\Pi,\Theta)}(x) - I_{(\Pi,\Theta)}(x + R_{(\Pi,\Theta)}(x)) \quad (6)$$

As the results will show, for the solution spaces of the problems that are discussed in this paper, $R_{(\Pi,\Theta)}(x) < I_{(\Pi,\Theta)}(x + R_{(\Pi,\Theta)}(x))$ is true, implying that $y < x$ is expected. Therefore, the combined algorithm-independent effects of noise are viewed as being harmful.

Given the aforementioned definitions for relaxation and inflation, it is relatively straightforward to express a notion of performance that discounts these algorithm-independent effects. Let $E[f] = E[f_{best}(g)]$ be the expected fitness value of a best solution found within g generations for problem Π. Likewise, define $E[h] = E[h_{best}(g)]$ to be the expected true fitness value of a best (according to the distorted fitness function) solution found within g generations for problem Θ. Then, the definition of performance can be expressed thus:

$$P_{(\Pi,\Theta)}(g) = E[h] + I_{(\Pi,\Theta)}(E[f] + R_{(\Pi,\Theta)}(E[f])) - R_{(\Pi,\Theta)}(E[f]) \quad (7)$$

It is possible to show that for an unbiased random search algorithm, which generates a number of solutions at random and selects the best one, $P_{(\Pi,\Theta)}(g) \approx P_{(\Pi,\Pi)}(g)$ for all time values g. This agrees with the intuitive notion that an unbiased random search is a noise-insensitive, albeit not typically very effective algorithm. Moreover, this property is a corollary of the fact that in the average case, an unbiased random search algorithm's performance in the face of noise is dictated strictly by the relaxation and the inflation effects. Therefore, for difficult problems, where the magnitude of these effects cannot be calculated exactly, a random search can be employed in approximating them. The details of this approximation will be covered in the following section.

Methods

The well-known *0-1 knapsack* problem [3,6] was selected to benchmark the algorithm's performance. The problem can be formally defined as follows:

There exist n items, where an item i has a value v[i] and a size s[i], where v[i] and s[i] are non-negative real numbers. For a knapsack of capacity C, find a combination (subset) K of items, subject to the constraint $\sum_{i \in K} s[i] \leq C$, *such that the value* $\sum_{i \in K} v[i]$ *is maximized.*

For the purpose of the experiment, two problems – one lightly constrained, and one heavily constrained – were randomly generated. Both problems consist of 50 items, where values and sizes are uniformly distributed between 0 and 10. The first problem's items have an approximate total size $s_{tot} \approx 270.61$; it was assigned a knapsack capacity of 200. The second problem's total size and capacity values are 229.36 and 100, respectively.

The experiment employed a rather standard version of the genetic algorithm [5], which was implemented by the author in the Java™ programming language. A binary encoding was used to represent arbitrary solutions to the knapsack problem, since it offers a very natural representation – the presence of an item in the knapsack is indicated by a "1" in the corresponding locus of the chromosome, while the absence is indicated by a "0".

In the algorithm, the initial population of individuals is generated randomly, and for each individual, an item is placed into the knapsack with a probability 0.5, thus giving each possible solution the same likelihood of being present initially. Given an existing generation of S solutions, the genetic algorithm creates a new generation (of the same size S) in the following manner. After the fitness of every solution is computed by a method, which shall be subsequently outlined, two elite (best) solutions are carried over to the next generation.

Tournament selection with replacement is then repeatedly applied, with a tournament size s_t, each time choosing two solutions to be potential parents. After each application, the parents are subjected to one-point crossover, with a probability p_c. Each parent (or resulting child) can be considered for mutation, with a probability p_m. During mutation, each bit is inverted with probability p_i.

The fitness $f = \sum_{i \in K} v[i]$ of any infeasible solution (i.e., any solution where $\sum_{i \in K} s[i] > C$ holds) is modified by a penalty function [6] as follows:

$$f = \sum_{i \in K} v[i] - (\sum_{i \in K} v[i])(\sum_{i \in K} s[i] - C) \div (\min(C, s_{tot} - C)) \quad (8)$$

A random number generator (RNG) is then used to modify the *true value* f to an *observed value* f' by the addition of a normal random variable with a mean of 0 and a standard deviation of σ:

$$f' = f + N(0, \sigma^2) \quad (9)$$

To simulate a problem that has a static, but imperfect fitness function, the observed fitness f' of any given solution is computed only once, as necessary. After the initial computation, this fitness value is hashed by the binary string of the solution, so that it can be retrieved if the fitness of this solution is requested again.

In order to produce results that not only carry greater statistical significance, but also enjoy an independence from one specific algorithm setting, the genetic algorithm was applied to the 0-1 knapsack problems under various parameter sets, as well as different RNG seeds. All possible combinations of parameters were employed, subject to the following constraints: $(p_c, p_m) \in$

$\{(0.9, 0.1), (0.5, 0.5)\}$, $s_t \in \{2, 5\}$, $S \in \{50, 100\}$, $p_i \in \{0.02\}$. In addition, two choices of seed values were available both for the RNG that is used by the algorithm to generate the initial population, and for the RNG that is used in producing new generations. In effect, the system simulated a collection of $2^5 = 32$ genetic algorithms, each with different parameter settings.

This set of genetic algorithms was invoked on each problem, for thirteen different noise levels. Each algorithm was allowed to run for 15 generations, and the best observed fitness $f'_{best}(15)$, as well as the true fitness $f_{best}(15)$ of the corresponding solution were recorded. For each noise level, ten invocations were performed, with the noise-producing RNG initialized with a distinct seed each time. Thus, for each of the two problems, and for every noise level, any given measurement is an average of $32 \times 10 = 320$ genetic algorithm runs.

To assess the algorithm's performance according to the definition developed earlier in the paper, the relaxation and inflation effects were approximated by measuring the performance of a simple unbiased random search algorithm on the two problems. First, this algorithm was applied to each problem without noise, until it generated a result that is as close as possible to the corresponding noise-free result produced by the genetic algorithm. The average time (measured in the number fitness function evaluations) necessary to achieve these results was 80,425 and 105,100, respectively. Subsequently, the algorithm was invoked on the two problems with positive levels of noise, but for the same number of evaluations. As for the genetic algorithm, ten invocations were performed at each noise level, with ten distinct noise-producing RNG seed values. Furthermore, for each such seed, thirty-two different seed values were used for the random algorithm's RNG. Thus, for the random search, each given measurement is also an average of 320 trials.

Results

For the random search algorithm (RS), the observed fitness curves of Figure 1 increase relative to the noise level σ, indicating the presence of the relaxation effect. True fitness, on the other hand, decreases as σ increases, as a result of a more powerful inflation effect. Similar trends are noted for the genetic algorithm (GA), but with a significant difference – for higher levels of noise, the observed fitness values of solutions that are returned by the genetic algorithm become progressively lower, relative to the corresponding fitness values that are discovered by the random search. In other words, as the noise level increases, the genetic algorithm's progress towards the upper tail of the fitness density distribution $\phi_\Theta(x)$ is impeded. In fact, the $\Delta f'_{best}(15)$ curves of Figure 2 indicate that this progress decreases at a rate that is somewhat greater than linear in σ. Thus, it appears that the genetic algorithm is affected by the noise-related

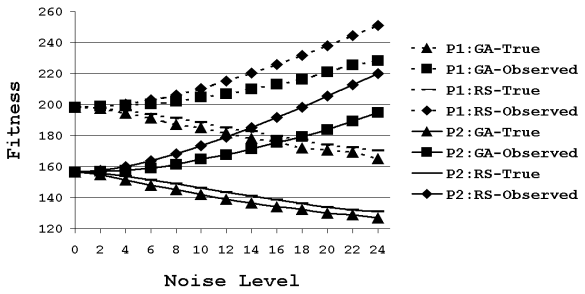

Figure 1: True and observed fitness values for best solutions found to Problem 1 (P1) and Problem 2 (P2).

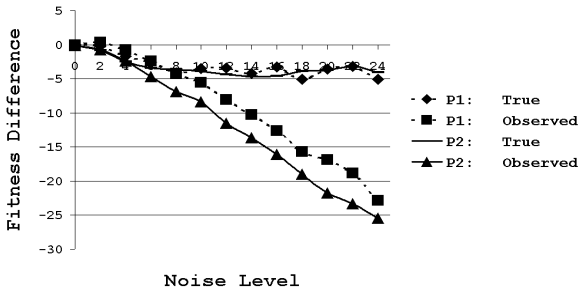

Figure 2: The differences $\Delta f_{best}(15)$ and $\Delta f'_{best}(15)$ between the genetic algorithm and the random search.

deterioration of the structure of the fitness landscape. On the other hand, from the definition that is adopted in this paper, it follows that the $\Delta f_{best}(15)$ curves show the relative performance of the genetic algorithm (with values close to 0 corresponding to noise-free performance), which decreases at a rate that is much slower than linear in σ, and in fact, may be approaching a constant value.

Discussion and Conclusions

One possible explanation for the seeming contradiction between the $\Delta f_{best}(15)$ and $\Delta f'_{best}(15)$ curves is that at the observed fitness values, which the genetic algorithm reaches (such as 228.25 and 194.69 for Problems 1 and 2, respectively, at $\sigma = 24$), the inflation effect is considerably smaller than at corresponding higher fitness values (251.11 and 220.16 at $\sigma = 24$) reached by the random search algorithm. In order to test for this possibility, the inflation effect was experimentally measured for each problem at $\sigma = 24$, by running the random search algorithm until it reached a value that is close to the observed fitness value given by the genetic algorithm for $\sigma = 24$, and subtracting the corresponding true fitness value. For Problem 1, $I_{(\Pi 1, \Theta 1)}(251.11) \approx 80.70$, while $I_{(\Pi 1, \Theta 1)}(228.25) \approx 67.27$, and for Problem 2, $I_{(\Pi 2, \Theta 2)}(220.16) \approx 89.11$, while $I_{(\Pi 2, \Theta 2)}(194.69) \approx 71.77$. This suggests that the primary reason for the genetic algorithm's robust performance in spite of its relative inability to take a full advantage of the relaxation effect is indeed statistical in nature – while for higher levels of noise, the observed fitness values of discovered solutions lie farther away from the upper tail of the fitness density distribution, these solutions are expected to be less misleading.

At the same time, the reduced inflation effect does not account for the full extent of the robustness, since the difference between $f'_{best}(15)$ and $f_{best}(15)$ is 62.92 for Problem 1 and 67.71 for Problem 2. Thus, it is apparent that the genetic algorithm has some ability to differentiate

between solutions that have a high observed fitness due to a large noise term, and solutions whose high observed fitness corresponds to a high true fitness. The author believes that this is an interesting phenomenon, both from a theoretical and a practical point of view, and that it deserves further study, which could be the subject of future work. Presently, a hypothesis is made that if a solution's observed fitness carries a particularly large noise term, then it tends to appear as a "spike" in the space of surrounding solutions, according to some notion of distance that is used by the genetic algorithm, and is less likely to be reached.

A final observation is that for both the heavily constrained and the lightly constrained problem, results are qualitatively similar, although the performance curve for the lightly constrained problem appears to have greater random variation. This fact gives hope to the idea of building a generalized theory for predicting how an evolutionary algorithm will be affected by noise, although experimentation with other problems will be necessary for the endeavor. Future research could also involve building more complex models of fitness function imperfections that can be present in various situations, and assessing the impact of such imperfections upon evolutionary algorithms.

Acknowledgements

The author wishes to extend his thanks to Dr. James A. Reggia and Dr. Eric V. Slud of the University of Maryland, as well as Mr. Apperson H. Johnson of Quantum Leap Innovations, Inc., for their advice in the development of this work.

References

[1] Dirk V. Arnold and Hans-Georg Beyer. Local Performance of the (1+1)-ES in a Noisy Environment. *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, Feb. 2002, pp. 30-41.

[2] Hans-Georg Beyer. Evolutionary Algorithms in Noisy Environments: Theoretical Issues and Guidelines for Practice. *Computer Methods in Applied Mechanics and Engineering*, 186 (2002), pp. 239-267.

[3] Pierluigi Crescenzi and Viggo Kann. A Compendium of NP Optimization Problems, (September 2002). http://www.nada.kth.se/~viggo/problemlist/compendium.html.

[4] Robert E. Keller and Wolfgang Banzhaf. Genetic Programming Using Genotype-Phenotype Mapping from Linear Genomes into Linear Phenotypes. *Genetic Programming 1996: Proceedings of the First Annual Conference*, pp. 116-122. 1996 MIT Press.

[5] Melanie Mitchell. *An Introduction to Genetic Algorithms*. 1996 MIT Press.

[6] Anne L. Olsen. Penalty Functions and the Knapsack Problem, pp. 554-558. 1994 IEEE.

[7] Riccardo Poli. Why the Schema Theorem is Correct also in the Presence of Stochastic Effects, pp. 487-492. 2000 IEEE.