Looking Under the EA Hood with Price's Equation

Jeffrey K. Bassett[1], Mitchell A. Potter[2], and Kenneth A. De Jong[1]

[1] George Mason University, Fairfax, VA 22030
{jbassett, kdejong}@cs.gmu.edu

[2] Naval Research Laboratory, Washington, DC 20375
mpotter@aic.nrl.navy.mil

Abstract. In this paper we show how tools based on extensions of Price's equation allow us to look inside production-level EAs to see how selection, representation, and reproductive operators interact with each other, and how these interactions affect EA performance. With such tools it is possible to understand at a deeper level how existing EAs work as well as provide support for making better design decisions involving new EC applications.

1 Introduction

Evolutionary algorithm design is difficult for a number of reasons, not the least of which is that the choices of selection, representation, and reproductive operators interact in non-linear ways to affect EA performance. As a consequence, EA designers are often faced with their own "black box" optimization problem in terms of finding combinations of design choices that improve EA performance on their particular application domain.

Results from the EC theory community continue to provide new insights into this difficult design process, but often are obtained by make simplifying assumptions in order to make the mathematics tractable. This often leaves open the question as to whether particular theoretical results apply in practice to actual EAs being used and/or to newly designed EAs.

The results presented here are part of an ongoing effort that tries to bridge this gap by using theoretical results to help us build useful tools that can be used to look inside actual EAs in order to better understand what's happening "under the hood". In particular, we have been exploring the use of some theoretical work done by Price (1970) to obtain deeper insights into the interactions of selection, representation, and the reproductive operators of crossover and mutation. In this respect we are indebted to Lee Altenberg who for some time now has been encouraging the EC community to pay more attention to Price's Theorem (Altenberg 1994; Altenberg 1995).

In section 2 we provide a brief overview of Price's Theorem and how it can be extended in a way to provide a useful analysis tool. Section 3 describes how Price's Theorem can be further extended to provide additional insights. We illustrate these ideas in section 4 by using the developed tools to instrument actual production-level EAs and showing how two EAs with similar "black box" behavior look quite different "under the hood". Finally, in section 5 we conclude with some observations and future work.

2 Background

In 1970, George Price published the article *Covariance and Selection* (Price 1970) in which he presented an equation that has proved to be a major contribution to the field of evolutionary genetics (Frank 1995). The equation describes the existence of a covariance relationship between the number of successful offspring that an individual produces and the frequency of any given gene in that individual. If this covariance value is high, then the existence of that gene is a good predictor of selection.

2.1 Price's Equation

Although Price focused on gene frequency, his equation is more general and can be used to estimate the change in any measurable attribute from the parent population to the child population, and separate the change attributable to selection from the change attributable to the genetic operators. Specifically,

$$\Delta Q = \frac{Cov(z,q)}{\overline{z}} + \frac{\sum z_i \Delta q_i}{N\overline{z}}, \tag{1}$$

where q_i is the measurement of some attribute of parent i such as the number of occurrences of a particular gene or combination thereof, z_i is the number of children to which parent i contributed genetic material, $\overline{z}$ is the average number of children produced by each parent, N is the number of parents, and Δq_i is the difference between the average q value of the children of i and the q value of parent i.

Price's Equation combines the effect of all the genetic operators into a single term. To further separate the effects of the individual reproductive operators, Potter et al. (2003) extended the equation as follows:

$$\Delta Q = \frac{Cov(z,q)}{\overline{z}} + \sum_{j=1}^{P} \frac{\sum z_i \Delta q_{ij}}{N\overline{z}}, \tag{2}$$

where P is the number of genetic operators and Δq_{ij} is the difference between the average q value of the children of i measured before and after the application of operator j.

2.2 Previous Applications in Evolutionary Computation

Altenberg (1995) was one of the first in the EC community to call attention to Price's Equation. He demonstrated Price's assertion that gene frequency is not the only attribute of the individuals which can be predicted by the equation, and identified several different measurement functions which could be useful, including mean fitness from both the biological and evolutionary computation perspectives, frequency of schemata, and evolvability.

More recently, Langdon and Poli (2002) showed how measuring gene frequencies is equivalent to determining the frequency of use of the available primitives in the evolving solution trees, and used Price's Equation to diagnose the probable causes of poorer performing runs.

Finally, Potter, Bassett, and De Jong (2003) concentrated on using fitness as a measurement function and applied Price's Equation to the visualization of the dynamics of evolvability, that is, the ability of an EA to continue to make improvements in fitness over time.

3 Variance of Operator Effects

Each term in Price's equation calculates the contribution of a different operator as a mean of the attribute being measured. Although visualizations based on the decomposition of delta mean fitness into operator-based components certainly provides more information than simple best-so-far curves (Potter et al. 2003), focusing on the mean can sometimes be misleading. In particular, the mean may be close to zero, leading one to believe that an operator is making a minimal contribution, when in fact it is a critical component of the evolutionary process.

In fact, the average individuals are not the ones driving evolution forward. It is the occasional exceptional individuals created by crossover and mutation that enable the population to continue to improve over time. The best and worst individuals are at the upper and lower tails of the population fitness distributions. Therefore, if we want to get some sense of how often an operator creates above or below average individuals, we need to look at the variance of the Δq values for mutation and crossover, not just the mean.

We should emphasize that we are interested in the variance of the effect of an operator on the individuals of a population, not the variance of the mean effect given multiple runs. Specifically, let $E[X]$ be the expected change in the measurement of q due to a particular operator such as crossover. For simplicity we will assume for the moment that only a single operator is used. Expanding the second term of equation 1 we have

$$\begin{aligned} E[X] &= \frac{\sum_{i=1}^{N} z_i \Delta q_i}{N\overline{z}} \\ &= \frac{\sum_{i=1}^{N} z_i \frac{\sum_{k=1}^{z_i}(q_{ik}-q_i)}{z_i}}{N\overline{z}} \\ &= \frac{\sum_{i=1}^{N}\sum_{k=1}^{z_i}(q_{ik}-q_i)}{N\overline{z}}, \end{aligned}$$

where q_{ik} is the measured q of the kth child of parent i, N is the number of parents, z_i is the number of children produced by parent i, and $\overline{z}$ is the average number of children produced by each parent. From this expansion we see that the random variable of interest is $X = q_{ik} - q_i$, and

$$Var[X] = \frac{\sum_{i=1}^{N}\sum_{k=1}^{z_i}(q_{ik}-q_i)^2}{N\overline{z}} - \left(\frac{\sum_{i=1}^{N}\sum_{k=1}^{z_i}(q_{ik}-q_i)}{N\overline{z}}\right)^2. \quad (3)$$

This can be extended to multiple operators by expanding equation 2, resulting in $X = q_{ijk} - q_{i(j-1)k}$, where q_{ijk} is the measured q of the kth child of parent i after the application of operator j.

4 Looking Under the Hood

To illustrate how to use these ideas to look under the hood, we will compare the performance of our evolutionary algorithm using two different mutation operators. In particular we will focus on the importance of the fitness variance in relation to Price's Equation.

4.1 Example Problem

To help in our illustration, we have chosen a standard problem from the function optimization literature introduced by Schwefel (1981). The objective function

$$f(x) = 418.9829n + \sum_{i=1}^{n} x_i \sin\left(\sqrt{|x_i|}\right)$$

defines a landscape covered with a lattice of large peaks and basins. The predominant characteristic of the Schwefel function is the presence of a second-best maximum far away from the global maximum, intended to trap optimization algorithms on a suboptimal peak. The best maximums are near the corners of the space. In this formulation of the problem, the global minimum is zero and the global maximum is $837.9658n$. In our experiments the problem has thirty independent variables constrained to the range $(-500.0, 500.0)$.

4.2 Algorithm Design

Typically as one makes design decisions for an evolutionary algorithm they go through a process of trial and error. Most of us simply exchange components, perform some runs, and then compare the best-so-far curves. We find a component that seems to work well for our problem, and then move on to other design decisions or parameter tuning, hoping that all of our choices will complement each other.

Let us assume that we have already made a series of design decisions, including using a real-valued representation and (μ, λ) selection of (500, 1000). We've chosen these unusually large population sizes in order to increase the sample sizes and reduce the amount of noise when calculating the terms in Price's equation. We've also decided to use two-point crossover at a rate of 0.6 and we've implemented it all with the ECKit Java class library developed by Potter (1998).

Now we are left with one final decision, the mutation operator. We are going to use a Gaussian mutation which is applied to all genes in the genome, but we want to decide between using a fixed standard deviation or one where the standard deviations are adapted, as described in (Bäck and Schwefel 1993).

4.3 Comparing Mutation Operators

In order to compare operators, we perform 100 runs using the fixed Gaussian mutation (with a standard deviation of 1.0), and 100 runs with the adaptive gaussian mutation. The best-so-far curves for each are plotted Figure 1. Based on what we see here, both

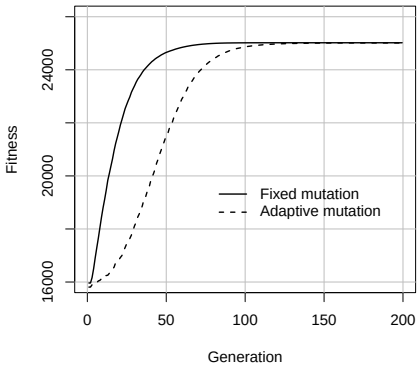

Fig. 1. Comparison of best fitness curves from Schwefel function optimization using fixed and adaptive Gaussian mutations. The global optimum is approximately 25,140.

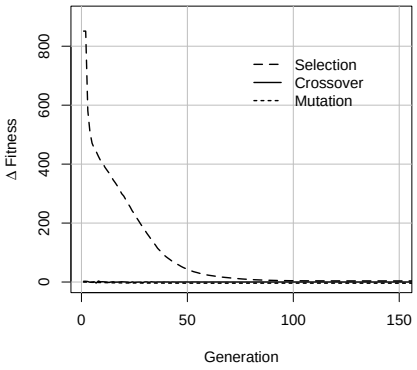

Fig. 2. The mean contributions from selection, crossover and mutation during optimization of the Schwefel function using fixed Gaussian mutation. These were calculated using the extended Price's equation.

appear to be able to reach the optimum consistently, but the fixed Gaussian mutation operator seems to be able to do so more quickly. Without any other information, this is the operator we would choose.

But do we have any idea why this operator is better than the other? The EA is essentially a black box, and we have very little idea of what is happenning inside. If we could get more information about how the operators are interacting and what effects they are having, we could make more intelligent decisions about how to improve the performance of our EA.

By applying the extended version of Price's Equation (equation 2) to the EA while it is running, we can calculate the average effect that each operator has on the population. Figure 2 shows the mean effects of selection, crossover and mutation when using the fixed Gaussian mutation operator. The plots shows that the crossover and mutation operators have an average effect near zero (which may be difficult to read in the plot), while selection seems to be doing all of the work. What exactly does this mean? These average values are clearly not telling us the whole story. Recall that in section 3 we claimed that the average individuals created by the genetic operators are not as important as the exceptional ones because it is the exceptional individuals which are chosen by selection.

We want to look at the standard deviations of the Δq values, but instead of plotting them in a separate plot, we have combined it with the existing plots of the delta mean fitnesses. Figure 3 gives an example of this type of plot for the crossover operator. For each generation a gray band is drawn starting from the mean effect of crossover (which is very close to zero) up to plus one standard deviation and down to minus one standard

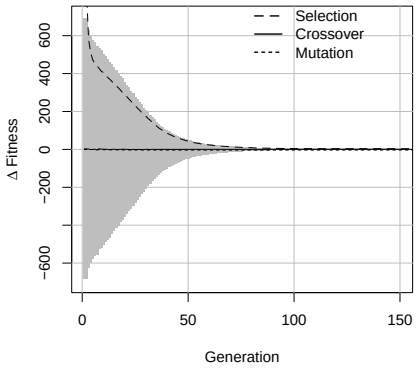

Fig. 3. Standard deviation of delta fitness effects caused by the crossover operator. The experiment was run using a fixed Gaussian mutation with standard deviation of 1.0 on the Schwefel function. Results are averaged over 100 runs.

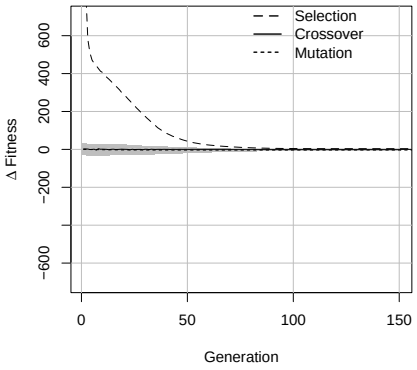

Fig. 4. Standard deviation of delta fitness effects caused by the mutation operator. The experiment was run using a fixed Gaussian mutation with standard deviation of 1.0 on the Schwefel function. Results are averaged over 100 runs.

deviation. This of course assumes that the fitness distribution of the Δq values is normal, which is not always a good assumption.

The advantage of this type of plot is that one can compare the relative effects of the delta mean fitness to the effects of the variance. In other words, an operator may on average be quite disruptive (low mean fitness), but still have a high variance. This would mean that the operator could still be effective. One should keep in mind though that we are plotting only one standard deviation. Genetic operators can often create individuals with fitnesses that are two to three standard deviations from the mean when run on this problem.

Getting back to the experiments, Figures 3 and 4 show the variance effects of crossover and fixed Gaussian mutation. Whereas before the two operators were indistinguishable, now we can see a large difference between them. Notice that the upper edge of the crossover standard deviation curve follows the curve representing mean contribution from selection very closely. It seems clear that crossover is contributing more to the search process, at least in the early generations. The variance of the mutation operator is much lower, as can be seen by the very narrow gray band along the x-axis. It is unlikely mutation does much more than local hill climbing.

Moving on to the adaptive Gaussian mutation operator, Figure 5 shows the mean effects of selection, crossover and mutation when using this operator. Here we see that once again the average effect of crossover is close to zero, but this time the average effects of mutation are much more negative. The adaptive mutation operator appears to be much more disruptive than the fixed Gaussian mutation.

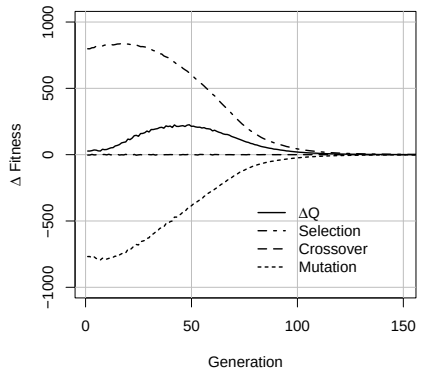

Fig. 5. The mean contributions from selection, crossover and mutation during optimization of the Schwefel function using adaptive Gaussian mutation. These were calculated using the extended Price's equation.

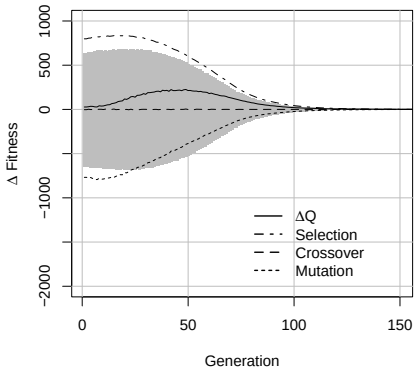

Fig. 6. Standard deviation of delta fitness effects caused by the crossover operator. The experiment was run using an adaptive Gaussian mutation on the Schwefel function. Results are averaged over 100 runs.

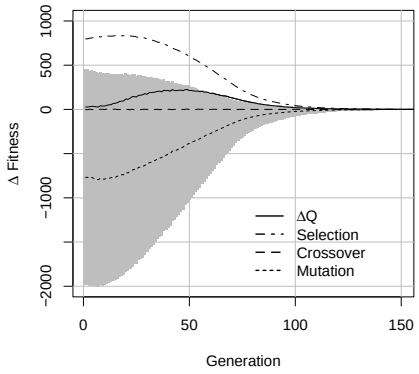

Fig. 7. Standard deviation of delta fitness effects caused by the mutation operator. The experiment was run using an adaptive Gaussian mutation on the Schwefel function. Results are averaged over 100 runs.

But we should not draw too many conclusions before we see the variance effects of the operators, which are plotted in Figures 6 and 7. One of the first things to note in Figure 7 is that although mutation is very disruptive on average, the upper edge of the standard deviation curve still comes above the x-axis by a fair margin, which indicates that it is contributing to the search process more than the fixed gaussian mutation operator did.

There is something more interesting though. Now we can get some insight into why it takes longer to reach the optimum using the adaptive Gaussian mutation than it does using the fixed Gaussian mutation. Compare the crossover standard deviations in Figure 6 with the ones in Figure 3. In conjunction with the adaptive mutation, crossover continues to have high variances out to generation 50, as opposed to the rapidly decreasing variances we see when fixed Gaussian mutation is used. The disruptive effects of mutation are so high that crossover is having to spend more time repairing individuals. It cannnot make headway in the search process until the standard deviations for mutation have been reduced.

With this knowledge we can now make a more informed decision about choosing our mutation operator. It is clear that crossover can take care of exploration on its own (in this domain), so we do not need a mutation operator which performs search also, especially when it comes with the undesirable side effects of disuption.

5 Conclusions and Future Work

In this paper we have shown how tools based on extensions of Price's equation allow us to look inside production-level EAs to see how selection, representation, and reproductive operators interact with each other, and how these interactions affect EA performance. In particular, we have shown how these extensions can provide insight into the way in which reproductive operator variance provides the exploratory power needed for good performance.

The reported results focused on ES-like EAs. We are in the process of completing a similar study for GA-like EAs. The interesting preliminary results suggest that in this case crossover and mutation interact internally in quite different ways. With results such as these we believe that it is possible to understand at a deeper level how existing EAs work as well as provide support for making better design decisions involving new EC applications.

Acknowledgments

We thank Donald Sofge, Magdalena Bugajska, Myriam Abramson, and Paul Wiegand for helpful discussions on the topic of Price's Equation. The work reported in this paper was supported by the Office of Naval Research under work request N0001403WX20212.

References

Altenberg, L. (1994). The evolution of evolvability in genetic programming. In K. E. Kinnear, Jr. (Ed.), *Advances in Genetic Programming*, Chapter 3, pp. 47–74. MIT Press.

Altenberg, L. (1995). The schema theorem and Price's theorem. In L. D. Whitley and M. D. Vose (Eds.), *Foundations of Genetic Algorithms III*, pp. 23–49. Morgan Kaufmann.

Bäck, T. and H.-P. Schwefel (1993). An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation 1*(1), 1–23.

Frank, S. A. (1995). George price's contributions to evolutionary genetics. *Journal of Theoretical Biology 175*, 373–388.

Langdon, W. B. and R. Poli (2002). *Foundations of Genetic Programming*. Berlin Heidelberg: Springer-Verlag.

Potter, M. A. (1998). Overview of the evolutionary computation toolkit. http://cs.gmu.edu/ mpotter/.

Potter, M. A., J. K. Bassett, and K. A. De Jong (2003). Visualizing evolvability with price's equation. In *Proceedings of the 2003 Congress on Evolutionary Computation*, pp. 2785–2790. IEEE.

Price, G. (1970). Selection and covariance. *Nature 227*, 520–521.

Schwefel, H.-P. (1981). *Numerical optimization of Computer models*. Chichester: John Wiley & Sons, Ltd.