Using Problem Generators to Explore the Effects of Epistasis*

Kenneth A. De Jong
Computer Science Department
George Mason University
Fairfax, VA 22030
kdejong@gmu.edu

Mitchell A. Potter
Computer Science Department
George Mason University
Fairfax, VA 22030
mpotter@gmu.edu

William M. Spears
Code 5510 - AI Center
Naval Research Laboratory
Washington, DC 20375-5337
spears@aic.nrl.navy.mil

Abstract

In this paper we develop an empirical methodology for studying the behavior of evolutionary algorithms based on problem generators. We then describe three generators that can be used to study the effects of epistasis on the performance of EAs. Finally, we illustrate the use of these ideas in a preliminary exploration of the effects of epistasis on simple GAs.

1 Introduction

Although we have made significant progress in recent years with respect to our ability to analyze evolutionary algorithms (EAs) formally, there are still considerable gaps between the algorithms we use, the classes of problems we wish to solve, and our formal models. As a consequence, much of our understanding of the behavior and the capabilities of EAs is the result of a large body of empirical studies in which comparative studies are performed on a set of test problems, and conclusions are drawn concerning the relative merits of one EA over another.

However, a continual concern of such empirical studies is that their results may not generalize beyond the test problems used. A classical example of this is a study in which a new algorithm is carefully tuned to the point that it outperforms some existing algorithms on a few problems, for example, the De Jong test suite (De Jong 1975). The results of such studies typically have only weak predictive value regarding relative performance on new problems.

*To appear in Proceedings of The Seventh International Conference on Genetic Algorithms, July 1997, Michigan State University, East Lansing, MI

This frequently leaves us "between a rock and a hard place" in that neither our theoretical models nor our empirical studies are sufficiently powerful to make accurate predictions about the performance of EAs on many classes of problems. And, of course, we are also periodically reminded that there is no free lunch (Wolpert and Macready 1995)!

There is considerable effort being put into improved theoretical models (cf. Belew and Vose 1996). There has also been some discussion about improving our empirical tools, mostly along the lines of improved test suites. One of the recurring criticisms of current test suites is that they do not contain many instances of non-separable problems (Salomon 1996). Hence, EAs with strong biases to search along coordinate axes perform better than other EAs on such "impoverished" test suites.

While we share such concerns, we feel there is more that needs to be done than just adding a few more problems to existing test suites. Rather, we feel that to improve our empirical studies we need to develop and use problem generators that, when used appropriately can strengthen considerably the results obtained.

In this paper we develop an empirical methodology based on problem generators. We then describe three generators that can be used to study the effects of epistasis on EAs. Finally, we illustrate the use of these ideas in a preliminary exploration of the effects of epistasis on simple GAs. In this paper we concentrate on epistatic problems that are non-separable, although epistasis need not imply non-separability.

2 Using Problem Generators

There are several ways in which one can strengthen the results obtained from empirical studies, the most important of which is to remove the opportunity to hand-tune algorithms to a particular problem or sets

of problems. An abstract model that accomplishes this is one in which EAs are submitted for testing by an independent agent who treats each EA as a "black box" and makes no tuning adjustments during evaluation. There must, of course, be some agreement as to the classes of problems to be used for testing, but there should be no *a priori* knowledge concerning the particular problem instances to be used. It is the responsibility of the evaluator to provide a set of problem instances that are both rich in variety and unbiased for testing purposes.

While this model seems to suggest the need for an independent testing agency, our experience has been that these goals can be achieved quite simply (and perhaps more systematically) through the use of test problem generators that are capable of producing randomly generated test problems on demand. Since our EAs are stochastic algorithms, we already require empirical results to be reported in terms of average behavior along with the variance and tests of statistical significance. Having problem generators allows us to report such results over a randomly generated set of problems rather than a few hand-chosen examples. Thus, by increasing the number of randomly generated problems, we increase the predictive power of the results for the problem class as a whole.

A second advantage of problem generators is that in most cases they are quite easy to parameterize, allowing one to design controlled experiments in which one or more properties of a class of problems can be varied systematically to study the effects on particular EAs.

In the remainder of this paper we describe three problem generators that allow for a more systematic study of epistasis, and we illustrate their use in understanding better the effects of epistasis on GAs.

3 An NK Landscape Generator

The most obvious candidate for a problem generator to study the effects of epistasis is Kauffman's tunable NK model of fitness landscapes (Kauffman 1989). In the NK model, N represents the number of genes in a haploid chromosome and K represents the number of linkages each gene has to other genes in the same chromosome. To compute the fitness of the entire chromosome, the fitness contribution from each locus is averaged as follows:

$$f(chromosome) = \frac{1}{N}\sum_{i=1}^{N} f(locus_i),$$

where the fitness contribution of each locus, $f(locus_i)$, is determined by using the (binary) value of gene i together with values of the K interacting genes as an index into a table T_i of size 2^{K+1} of randomly generated numbers uniformly distributed over the interval $[0.0, 1.0]$. For a given gene i, the set of K linked genes may be randomly selected or consist of the immediately adjacent genes.

From a practical perspective, there are some problems in developing a generator of NK landscape problems, the most important of which is the space required to store the tables used to compute fitness. The amount of storage is $N2^{K+1}$. Even with clever programming tricks involving sparse tables and lazy evaluation we are restricted to relatively small models.

Another concern with the NK model is that all genes have the identical degree (K) of epistasis, while most "real world" problems vary considerably in the amount of epistasis among parameter subsets. This concern is not easily remedied within the NK model, but can be addressed by looking at other problem classes such as boolean satisfiability that are not quite so abstract.

4 A Boolean Satisfiability Problem Generator

Boolean satisfiability (SAT) problems have served as a useful testbed for many areas of algorithm development. They are known to be NP-complete and are frequently used as the canonical representative of this class of difficult problems (De Jong and Spears 1989). The most general statement of SAT problems is: given an arbitrary boolean expression involving V boolean variables, find an assignment of truth values to the V boolean variables that makes the entire boolean expression true. There is no guarantee that such an assignment exists, and clearly the difficulty of the problem generally increases as a function of both the number of boolean variables and the complexity of the boolean expression.

Since any boolean expression can be converted to an equivalent expression in a canonical form such as disjunctive normal form (DNF) or conjunctive normal form (CNF), it is frequently assumed that SAT problems are presented in a canonical form. Such forms are usually simplified further by assuming (at no loss of generality) that all clauses are of the same length L. This makes it easier to quantify the complexity of a boolean expression in terms of the number C of disjunctive (or conjunctive) clauses.

It should be clear that for such normal forms it is quite straight forward to implement a boolean expression generator with parameters specifying V, C, and L.

In our case we are looking for a particular form that allows us to study the effects of epistasis in a controlled fashion. One such generator that has this property is referred to as Random L-SAT (Mitchell, Selman, and Levesque 1992), which we now describe in more detail.

The Random L-SAT problem generator creates random problems in conjunctive normal form subject to the three parameters V, C, and L. Each clause is generated by selecting L of the V variables uniformly randomly and negating each variable with probability 0.5.

We can make direct contact here with the biological notions of pleiotropy (a gene may influence multiple traits) and polygeny (a trait may be influenced by multiple genes). For these L-SAT problems, each clause can be considered to be a trait. Hence, the polygeny is of order L. The pleiotropy is estimated by noting that each variable occurs (on average) in CL/V clauses.

By systematically controlling and varying these parameters, one can vary both the type and the amount of epistasis. In addition, note that the Random L-SAT generator produces problems with much smaller storage requirements (of order LC), which allows one to study much larger problems than the NK model. Also, as noted above the amount of epistasis varies among subsets of genes, which is more representative of practical problems than the uniform level of epistasis built into the NK model.

5 GAs and Epistasis

There has been a considerable amount of discussion and (often contradictory) studies regarding how the performance of EAs in general and GAs in particular is affected by epistasis. These discussions frequently center on the usefulness, or lack thereof, of operators such as crossover and mutation in solving epistatic problems, with evidence (pro and con) in the form of theoretical arguments or empirical studies on a few carefully chosen examples (Booker 1992; Davidor 1990; De Jong and Spears 1992; Fogel 1995; Goldberg 1989; Holland 1975; Schaffer and Eshelman 1991; Spears 1992).

The lack of emergence of a clear picture of these effects suggests to us that there is an opportunity here to improve our understanding by way of systematic studies using problem generators. We have begun to do so, and report our initial findings here.

To keep things simple initially, we have focused on the effects of epistasis on a standard textbook GA. That is, we used a generational GA with scaled fitness proportional selection. Recombination is done in the standard way using two-point crossover, and mutation is the standard bit-flipping operator.

To solve NK and L-SAT problems with a GA we need to select an internal representation and a fitness function. The representation used is the most natural one: a bit string of length N for NK problems, and of length V for the L-SAT problems. The fitness function was given for NK problems previously. For L-SAT problems the boolean expressions are in CNF, so we used the simplest and most intuitive fitness function, the fraction of clauses that are satisfied by the assignment. More formally,

$$f(chromosome) = \frac{1}{C}\sum_{i=1}^{C} f(clause_i),$$

where the fitness contribution of each of the clauses, $f(clause_i)$, is 1 if the clause is satisfied and 0 otherwise.

The conventional wisdom is that recombination should have a relative advantage over mutation when epistasis is small, while mutation should have a relative advantage when epistasis is large. These two problem generators allow us to test this hypothesis by varying K (for a fixed N) on NK problems systematically. For L-SAT problems there are a number of options. We have chosen the following strategy in our initial studies: we keep L and V fixed, and we vary the number of clauses C to increase or decrease the amount of pleiotropic epistasis.

6 Initial Experimental Results

A full study of epistasis effects on GAs is beyond the scope and the size limitations of a conference paper. We present here some interesting initial results that illustrate the usefulness of the problem generator approach. The experimental methodology we used consistently throughout was as follows: for each of the selected settings of our problem generator parameters, we randomly generated 50 problems. Hence, all the results presented in this paper represent observed behavior averaged over 50 randomly generated problems.

6.1 NK Experiments

For our initial experiments with NK landscape problems we use a fixed N of 48 and varied K from 0 to 24. As noted earlier, the storage requirements for the NK tables make it difficult to explore models with much larger K. We also used the simpler form of linkage, the neighborhood model, rather than randomly generated linkages.

We started with fairly standard settings of the GA parameters: a crossover rate of 0.6, and a mutation rate equal to the reciprocal of the chromosome length, and population sizes ranging from 50–200. We were interested in how the behavior of this simple GA is affected by increasing epistasis. To separate out the individual effects due to crossover and mutation, we compared three GA variations: one using only crossover (GA-c), one using only mutation (GA-m), and the standard GA using both operators.

Figures 1–3 illustrate typical results for low ($K = 4$), medium ($K = 12$), and high ($K = 24$) epistasis, with a population size of 200. The vertical bars overlaying the best-so-far curves represent 95-percent confidence intervals computed from Student's t-statistic (Miller 1986).

As expected, one can see a clearly diminishing advantage of crossover over mutation as the amount of epistasis increases. However, there were several surprising results. Note that the standard GA is consistently outperformed by GA-c, and isn't much better than GA-m. On the basis of our own personal experiences and other results in the literature, we expected the standard GA to dominate on low epistasis problems, and GA-m to dominate on problems with high epistasis.

These trends were relatively insensitive to changes in population size. What was not clear was whether these results were peculiar to NK landscape problems. To clarify this, we ran a similar set of experiments with the L-SAT generator.

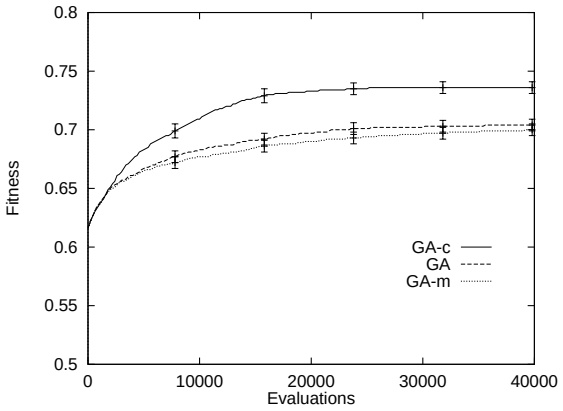

Figure 1: Average best-so-far curves for GA, GA-c, and GA-m on NK problems with low epistasis.

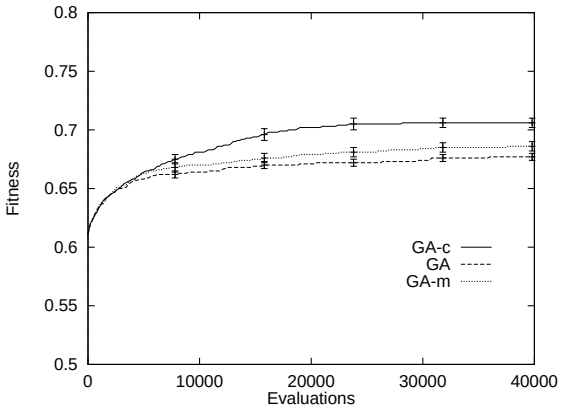

Figure 2: Average best-so-far curve for GA, GA-c, and GA-m on NK problems with medium epistasis.

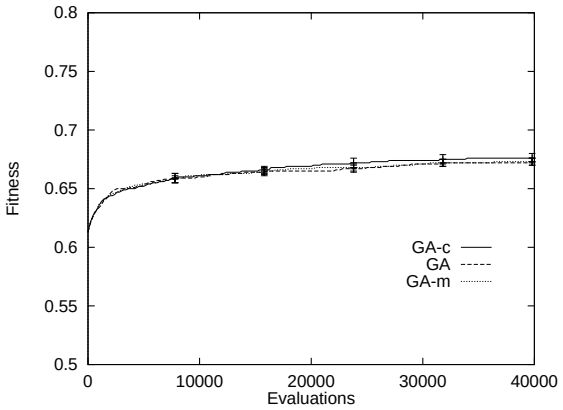

Figure 3: Average best-so-far curve for GA, GA-c, and GA-m on NK problems with high epistasis.

6.2 Initial L-SAT Experiments

For our initial L-SAT experiments we fixed the number of variables V to 100, the length of the clauses L to 3, and the population size to 100. As discussed earlier, this allowed us to vary the degree of epistasis in a manner similar to the NK problems by varying the number of clauses C from 200 (low epistasis) to 2400 (high epistasis).

We were somewhat surprised to observe nearly identical trends. Figures 4–6 illustrate the typical behavior observed for low epistasis (200 clauses), medium epistasis (1200 clauses) and high epistasis (2400 clauses).

As before, we ran additional experiments to test the sensitivity of these results to various GA parameter settings. Again, the observed trends were insensitive to changes in population size, and the trends were amplified by increasing the mutation rate.

We also tried decreasing the mutation rate from its default value (which in this case corresponded to a

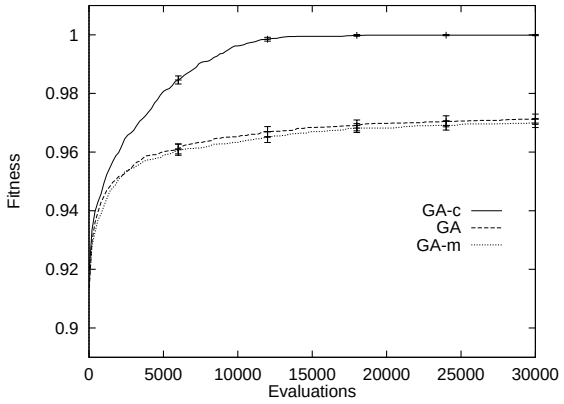

Figure 4: Average best-so-far curves for GA, GA-c, and GA-m on L-SAT problems with low epistasis.

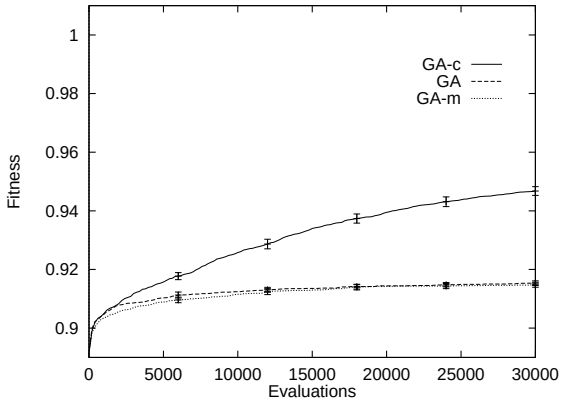

Figure 5: Average best-so-far curves for GA, GA-c, and GA-m on L-SAT problems with medium epistasis.

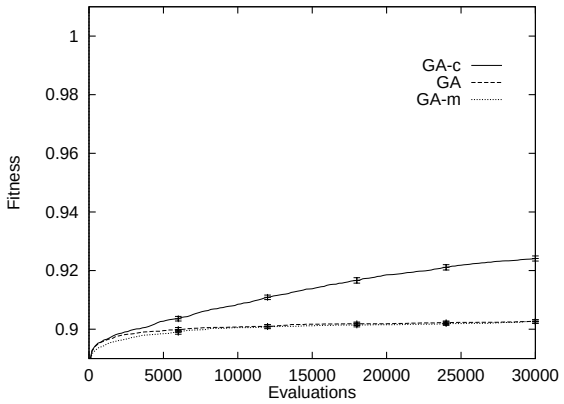

Figure 6: Average best-so-far curves for GA, GA-c, and GA-m on L-SAT problems with high epistasis.

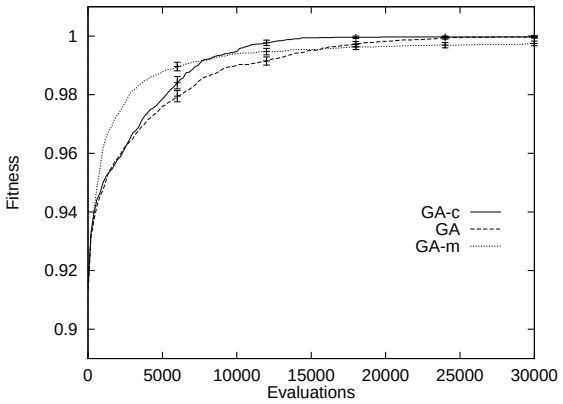

Figure 7: Average best-so-far curves for GA, GA-c, and GA-m using a 0.001 mutation rate on L-SAT problems with low epistasis.

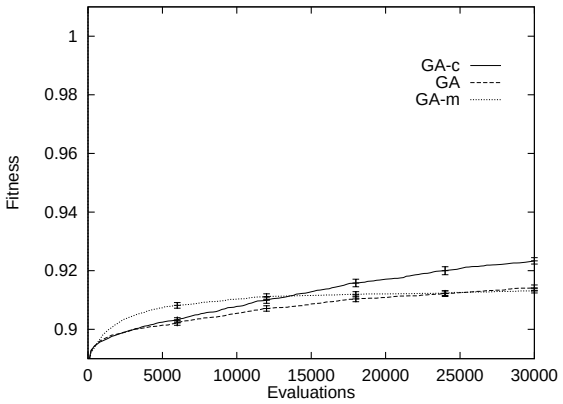

Figure 8: Average best-so-far curves for GA, GA-c, and GA-m using a 0.001 mutation rate on L-SAT problems with high epistasis.

mutation rate of 0.01) by an order of magnitude to 0.001. This produced rather striking changes in GA behavior, resulting in trends much closer to our original expectations.

Figures 7 and 8 are typical of what we saw, namely, that GA-m has the initial advantage, but is ultimately overtaken by GA-c and GA. As epistasis increases, the initial advantage of GA-m lasts for an increasing number of evaluations.

These results suggested a somewhat different explanation for the observed trends. These simple generational GAs have no *explicit* form of elitism; rather, there is an *implicit* form of elitism in the sense that some of the offspring produced may be identical copies

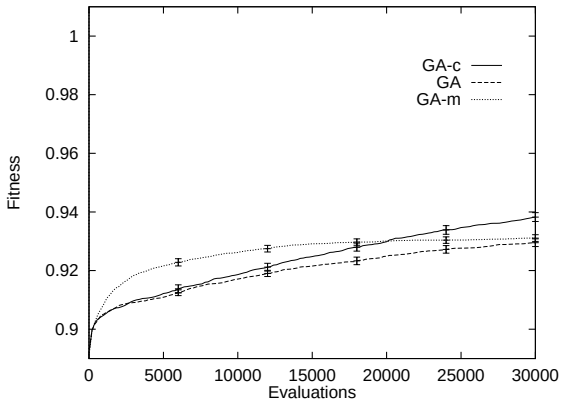

Figure 9: Average best-so-far curves for GA, GA-c, and GA-m using a 0.001 mutation rate and 1.0 crossover rate on L-SAT problems with medium epistasis.

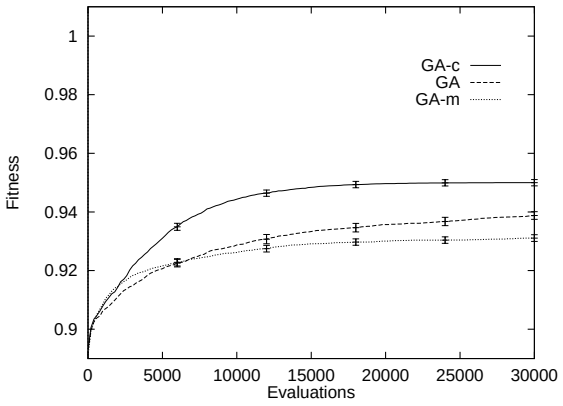

Figure 10: Average best-so-far curves for GA, GA-c, and GA-m using a 0.001 mutation rate and 0.2 crossover rate on L-SAT problems with medium epistasis.

of their parents. The performance of these GAs depends on a proper balance when producing the next generation between exploration via genetically different offspring and exploitation via genetically identical offspring. The default mutation rate results in little or no cloning, which is apparently too high an exploration rate for these simple, non-elitist GAs.

To test this hypothesis we performed an additional set of experiments in which we increased and decreased the rate of crossover, and we observed similar changes in behavior. Increasing the crossover rate from 0.6 to 1.0 significantly degraded performance as shown in figure 9, while reducing the crossover rate to 0.2 significantly improved performance as seen in figure 10.

At this point it became clear what was going on. Typical operator rates are too explorative for a simple, generational non-elitist GA resulting in distorted performance curves that, in turn, mask the more subtle effects of epistasis.

6.3 Summary

We were quite pleased at this point with the experimental methodology using the problem generators to study these effects systematically. We found the NK and L-SAT generators to be useful, but not without problems. Table space requirements for the NK models restrict experiments involving large values of N and K. The L-SAT generator has the advantage that it permits the exploration of much higher levels of epistasis, and allows one to vary polygeny and pleiotropy independently. Both generators have the property that the variance in the fitness values decreases as the amount of epistasis increases. This results in more difficult fitness landscapes for simple GAs using proportional selection, independent of the effects of epistasis, resulting in additional masking of the effects of epistasis. We are currently developing a "multimodal" problem generator that allows one to increase epistasis without significantly reducing fitness variance. We describe this idea briefly in the next section.

7 Multimodal Generator

The idea is to generate a set of P random N-bit strings, which represent the location of the P peaks in the space. To evaluate an arbitrary bit string, first locate the nearest peak (in Hamming space). Then the fitness of the bit string is the number of bits the string has in common with that nearest peak, divided by N.

$$f(chrom) = \frac{1}{N} \max_{i=1}^{P} \{N - \text{Hamming}(chrom, Peak_i)\}$$

Problems with a small/large number of peaks are weakly/strongly epistatic. Figures 11 and 12 illustrate some of our preliminary results with respect to the performance of GA, GA-m, and GA-c on 1-peak and 500-peak problems, with a 0.001 mutation rate, 0.6 crossover rate, and a population size of 100. Note that the range of fitness values is not greatly affected by increasing the number of peaks, thus addressing the issue concerning the"flattening" of the NK and L-SAT landscapes as epistasis increases. What is most noticeable is the severe drop in performance of GA-c and

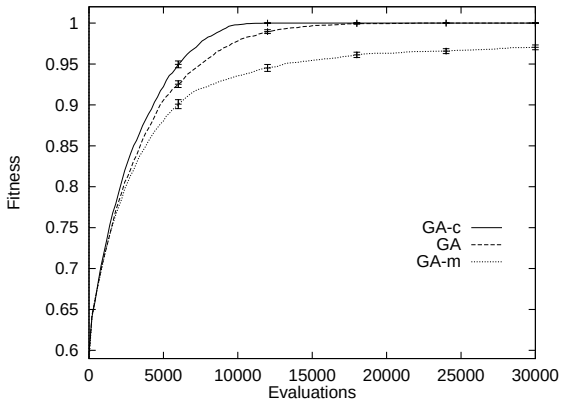

Figure 11: Average best-so-far curves for GA, GA-c, and GA-m on 1-peak problems.

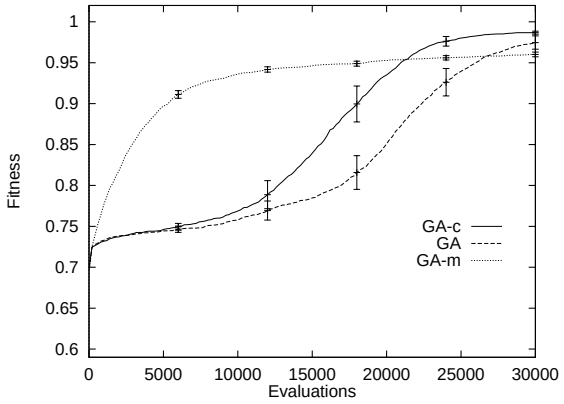

Figure 12: Average best-so-far curves for GA, GA-c, and GA-m on 500-peak problems.

GA for the 500-peak problems, while the GA-m curves are almost identical in the two figures. This provides strong confirmation of the increasing initial advantage of GA-m as epistasis increases.

8 Summary and Conclusions

There are a number of important observations that can be made from this initial use of problem generators to study the effects of epistasis on GAs. The most important observation is to note how subtle the effects are, and how easy it is to reach hasty and erroneous conclusions. Crossover and mutation do not operate in a vacuum. Rather, they are components that interact in complex ways with the other components of an evolutionary system. As a consequence, as we have seen, even something as simple as changing operator rates has side effects that can mask some of the more subtle effects of epistasis.

On the other hand, these initial results do support the notion that the relative advantage of crossover over mutation is reduced as epistasis increases. What is somewhat surprising, however, is the continued effectiveness of crossover at higher levels of epistasis than commonly believed. Clearly, more careful experiments will be required understand and to quantify this better.

Clearly, there is more work to be done here. We are currently using this methodology to find "optimal" settings for the operators involved. By doing so we should be able to keep the exploration/exploitation balance constant across experiments and perhaps see more clearly the effects of epistasis.

In addition, we are looking at the effects of epistasis on other EAs. As these preliminary results suggest, the effects could be quite different for rank-based selection, explicit forms of elitism, etc.

References

Belew, R. K. and M. Vose (Eds.) (1996). *Foundations of Genetic Algorithms (FOGA4)*. Morgan Kaufmann.

Booker, L. B. (1992). Recombination distributions for genetic algorithms. In *Proceedings of the Second Foundations of Genetic Algorithms Workshop*, pp. 29–44. Morgan Kaufmann.

Davidor, Y. (1990). Epistasis variance: A viewpoint of ga-hardness. In *Proceedings of the First Foundations of Genetic Algorithms Workshop*, pp. 23–35. Morgan Kaufmann.

De Jong, K. A. (1975). *Analysis of Behavior of a Class of Genetic Adaptive Systems*. Ph. D. thesis, University of Michigan, Ann Arbor, MI.

De Jong, K. A. and W. M. Spears (1989). Using genetic algorithms to solve np-complete problems. In J. D. Schaffer (Ed.), *Proceedings of the Third International Conference on Genetic Algorithms*, pp. 124–132. Morgan Kaufmann.

De Jong, K. A. and W. M. Spears (1992). A formal analysis of the role of multi-point crossover in genetic algorithms. *Annals of Mathematics and Artificial Intelligence 5*(1), 1–26.

Fogel, D. B. (1995). *Evolutionary Computation*. IEEE Press.

Goldberg, D. E. (1989). *Genetic Algorithms in Search,*

Optimization, and Machine Learning. Addison-Wesley.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems*. University of Michigan Press.

Kauffman, S. A. (1989). Adaptation on rugged fitness landscapes. In D. L. Stein (Ed.), *Lectures in the Sciences of Complexity*, Volume 1, pp. 527–618. Addison Wesley.

Miller, R. G. (1986). *Beyond ANOVA, basics of applied statistics*. John Wiley & Sons.

Mitchell, D., B. Selman, and H. Levesque (1992). Hard and easy distributions of sat problems. In *Proceedings of the Tenth National Conference on Artificial Intelligence*, pp. 459–465. AAAI Press/The MIT Press.

Salomon, R. (1996). Reevaluating genetic algorithm performance under coordinate rotation of benchmark functions. *BioSystems 39*, 263–278.

Schaffer, D. and L. Eshelman (1991). On crossover as an evolutionarily viable strategy. In R. K. Belew and L. B. Booker (Eds.), *Proceedings of the Fourth International Conference on Genetic Algorithms*, pp. 61–68. Morgan Kaufmann.

Spears, W. M. (1992). Crossover or mutation? In *Proceedings of the Second Foundations of Genetic Algorithms Workshop*, pp. 221–237. Morgan Kaufmann.

Wolpert, D. H. and W. G. Macready (1995). No free-lunch theorems for search. Technical Report 95-02-010, Santa Fe Institute.