A Comparison of the Effects of Dominance on Evolutionary Programming and Genetic Algorithms

Arnold L. Patton, Terrence W. Dexter, W. F. Punch III and E.D. Goodman
Genetic Algorithms Research and Application Group (GARAGe)
A714 Wells Hall, Computer Science Dept.
Michigan State University
E. Lansing MI, 48824
pattona@cps.msu.edu, dexterte@cps.msu.edu, punch@cps.msu.edu, goodman@egr.msu.edu
http://isl.cps.msu.edu/GA/

Abstract

Genetic Algorithms (GA) and Evolutionary Programming(EP) are both search techniques predicated upon simulating some of the processes outlined in evolutionary theories. Both techniques claim the capacity to find global minima (or alternately maxima) in large search spaces. Genetic Algorithms employ sexual reproduction among fit parents using genotypic operators such as crossover, bit mutation, and inversion. On the other hand, Evolutionary Programming espouses a form of asexual phenotypic reproduction whereby each candidate parent undergoes a zero-mean Gaussian mutation. Previous studies have shown that EP is capable of producing more precise solutions than GAs for a number of simple functions. This paper investigates another class of problem spaces for which GA outperforms EP, specifically problems where portions of a given solution have a greatly varying impact on the overall score. A simple representative problem is tested against typical EP and GA implementations with special attention paid to testing the effects of search parameters on both EP and GA performance. Finally, the relative performance and ease of use of the two paradigms is compared.

Introduction

A growing number of claims have been made regarding the use of asexual *phenotypic* Gaussian mutation as being superior in comparison to crossover and other *genotypic* operators commonly employed in Genetic Algorithms (GAs). Such reports invariably support use of Gaussian mutation in an algorithmic search method dubbed Evolutionary Programming (EP) as outlined in [Fogel94]. It is not the position of this paper to dispute such claims. EP certainly works impressively on a number of problems. Indeed many of our results show that on some problems EP may have considerable advantage over a GA approach. However, as we will show, there are equal difficulties and disadvantages in using EP which set it on par with GA, rather than above.

Specifically, this paper explores a class of problems which poses great difficulty for Evolutionary Programming but which remains tractable for Genetic Algorithms. The effects of parameter *dominance*, where parameters differ greatly as to the magnitude of their impact on the problem solution, will be the focus of our study here. A simple representative problem is created as a basis for examination. The results of direct analysis of representative EP and GA implementations on this problem are given and conclusions offered.

Background

Others have undertaken comparisons of different facets of GA and EP approaches on a number of problems both abstract and practical ([Baeck96],[Fogel94],[Fogel95]). Most of these concentrate on direct comparison of representative implementations of the two paradigms; however, [Fogel90] takes a novel approach comparing the fundamental operators of GA and EP by allowing simultaneously evolved solutions produced by each operator to compete for the limited resources of a single population, the ostensible assumption being that the offspring of the operator consistently producing the greatest advancement would dominate the population. While this type of approach is aesthetically appealing, it may not produce a valid comparison since crossover is by definition strongly influenced by the make-up of the entire population, whereas the Gaussian mutation used in EP is not. Therefore, we chose to return to head-to-head comparisons of "equivalent" GA and EP implementations allowing each to operate on independent populations. This more closely matches the method in [Fogel94].

Experimental Design

The following section details the class of problems to be studied and introduces the two representative implementations used for comparison.

Test Problem

Consider a vector $\boldsymbol{X}$ of size $\boldsymbol{n}$, each element of which is multiplied by some constant value $\boldsymbol{C_i}$. The resulting vector is compared to a target vector $\boldsymbol{T}$, and the sum squared error is assigned as the score of $\boldsymbol{X}$. If we assign the constants $\boldsymbol{C_i = c^{\,i}}$, then we call $\boldsymbol{c}$ the *dominance factor* of this problem. Thus, the score of vector $\boldsymbol{X}$ may be denoted:

$$\sum_{i=0}^{n} \left(t_i - \left(x_i \cdot c^{\,i} \right) \right)^2$$

Note that when $\boldsymbol{c}$ is 1.0, this problem is equivalent to the non-pleiotropic correlation matrix problem in [Fogel90].

Software

For the purposes of this comparison we used two software packages, a modified version of GALLOPS[Goodman94] by E. D. Goodman from the Michigan State University's GARAGe for the genetic algorithm, and PEP (Plain Evolutionary Programming) by N. Saravanan from Wayne State's ENCORE for evolutionary programming. Both of these packages are based upon Goldberg's SGA[Goldberg89] system and should provide a reasonably level playing field. Further, the GALLOPS package was modified to use the identical form of population initialization as PEP to allow for more direct comparisons.

Discussion

This section discusses issues which arise from our choice of test problem as well as from the two paradigms directly. First the issue of the validity of the proposed class of test problems is addressed, after which the relative difficulties of using both EP and GA implementations are discussed.

Problem Validity

One of the benefits of using artificially constructed problems is that we can control the parameters of the problem and focus on a single area of interaction between the problem space and the search engine. The inherent disadvantage in choosing an abstract problem is the requirement to show how the features being studied have direct bearing on non-trivial problem spaces. It may be difficult to locate real-world problems with provably large factors of parameter dominance.

However, since parameter interaction for many problem spaces of interest to GA and EP search are not

directly measurable (otherwise inductive search might be a better candidate), we cannot without a priori knowledge assume that such dominance factors are not a characteristic. In fact, since such measurements are rarely taken, large dominance factors may be commonplace. Further, depending on the specific characteristics of a given problem instance, problems in which parameter interactions are of different orders (i.e. squared, cubed, etc.) may display similar dominance effects.

Difficulties in using EP

While the question of encoding is often paramount in GA implementations, encoding in EP is essentially irrelevant since EP assumes that all parameters are continuous and real-valued. PEP uses a 64-bit floating point value for each parameter in the problem statement. The bias toward quantitative values in EP stems from the definition of the EP mutation operator, which precludes problem statements which require nominal parameters (i.e. parameters for which there is no conceptual "mean"). Further, while it is possible to limit the ranges of parameter values in the initial population, there is no clear method for constraining parameter values beyond the first generation.

Perhaps the most striking difficulty in using EP is selection of the mutation rate. PEP allows the standard deviation of the mutation applied to be set relative to the error rate (score) of an individual, or as a fixed rate for all individuals, or as a sum of the two. The same mutation rate is applied equivalently to each parameter in a given solution. For the test problems used here, a reasonable estimate of at least the magnitude of the best mutation rate might be surmised from the values given in a particular problem instance (not as easy a task as one might assume); however, we cannot rely on such tactics since they imply knowledge about the problem domain which should not be assumed. This difficulty in choosing mutation rates extended to both fixed as well as relative mutation rates despite claims that constant "small" mutation rates are capable of producing reasonable results in EP [Fogel94].

Except where noted, the mutation rates for these experiments were chosen by narrowing the range through successive trials for each problem instance to arrive at an estimate of the *best* mutation rate. Doubtless, the choice of mutation rate in these tests is a possible source of bias (as these successive preliminary tests may be seen as giving somewhat of an advantage to the EP implementation); however, the authors were unable to determine any mutation rate selection method which would be less biased.

Difficulties in using GA

Whereas in EP the issue of genotypic encoding is sidestepped, in a GA implementation the choice of encoding is perhaps the most crucial influence on the actual success or failure of the GA [Goldberg89]. GA makes no assumptions about the form of the solution, so nominal as well as quantitative information may be encoded; hence the GA approach lends itself easily to more advanced systems such as Classifier Systems[Holland75], and Genetic Programming[Koza92]. However, this flexibility can be blessing as well as curse - especially since there is little consensus over the *ideal* encoding for a given problem, other than vague references to the basic GA schema theorem [Goldberg89].

For purposes of keeping the evaluation between GA and EP as unbiased as possible, we chose to directly interpret the GA genome as successive 64-bit floating point values. Thus the problem space for both implementations is similar, though this encoding may not be the best possible choice for the GA. The crossover and mutation operators, however, treat the genome as a sequence of bits allowing crossover and bit mutation to occur at any point along the genome.

As in EP, choosing the rates of mutation and crossover are reasonably important; however, the ranges are much more limited and the effects of a poor decision are less pronounced than with the EP mutation rate. Throughout these experiments, the GA crossover rate is fixed at 90% (simple one point crossover), and the bit-mutation probability is 1/1000 bits. Use of inversion was not tested.

Experiments

For the first round of experiments presented here, a vector of five values was chosen. In the first set, a dominance factor of 1.0 was tested, while the dominance factor was set to 16.0 in the second. In each experiment for a given population size, ten random target vectors were chosen which would cause the solution vector to consist entirely of values between 0 and 1. Both the GA and EP implementations were tested on these ten targets and the best solution found in the first 500 generations was recorded for each. Figures 1-4 show the results of the average error for the ten targets for each paradigm for each population size as well as the best of the ten values. For the 1.0 dominance factor tests, the EP mutation rate factor was set at 1.0, while a rate of 1.0e-9 proved optimal for the 16.0 dominance factor tests. Both the GA and EP populations were seeded with uniformly selected random values between -1.0 and 1.0. (The initialization from PEP was ported directly into GALLOPS to make better comparisons; however, no effort was made to ensure that initial populations were identical.)

A similar experiment was carried out to examine the effects of problem size on GA and EP solution rates under high dominance. For this battery of tests, the size of the problem vector was increased from 1 to 10 values in length. The population size was selected at 2000 to be fair to

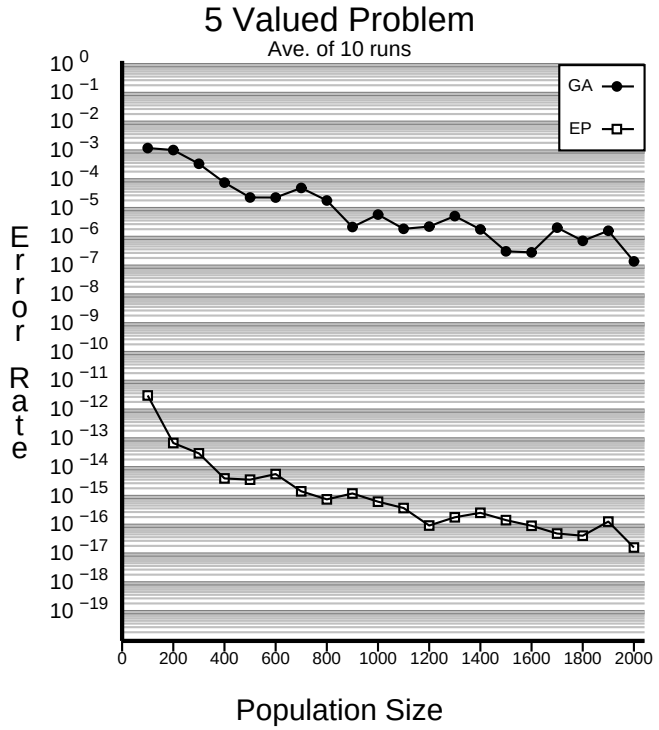

Figure 1. Effects of Population size w/ Low Dominance (Ave. of 10)

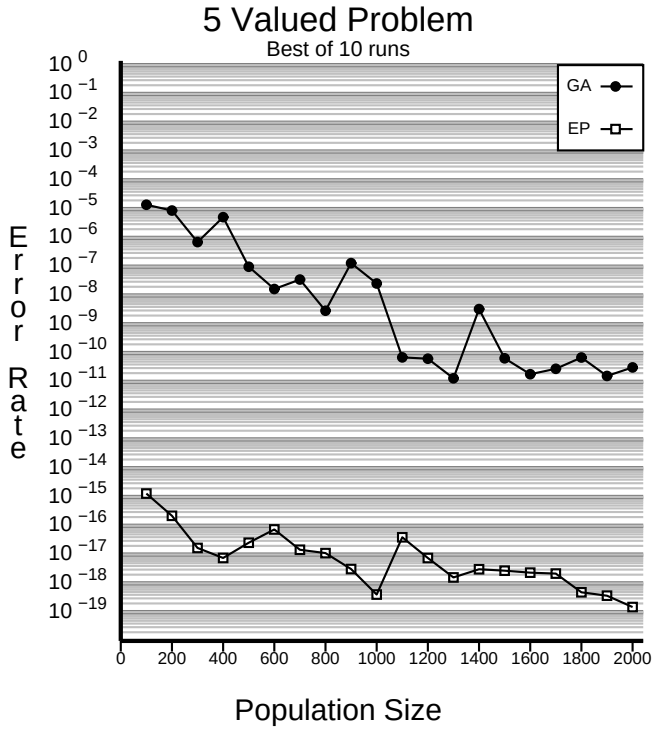

Figure 2. Effects of Population size w/ Low Dominance (Best of 10)

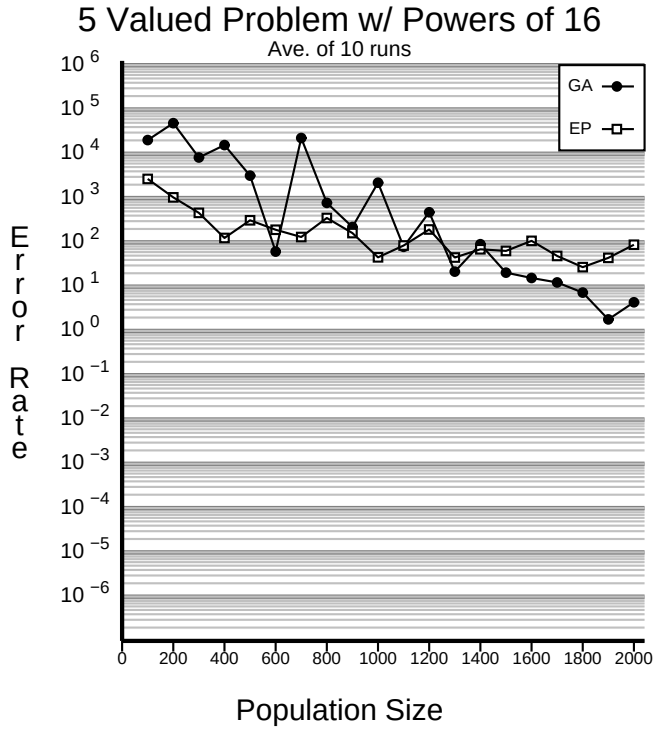

Figure 3. Effects of Population size w/ High Dominance (Ave. of 10)

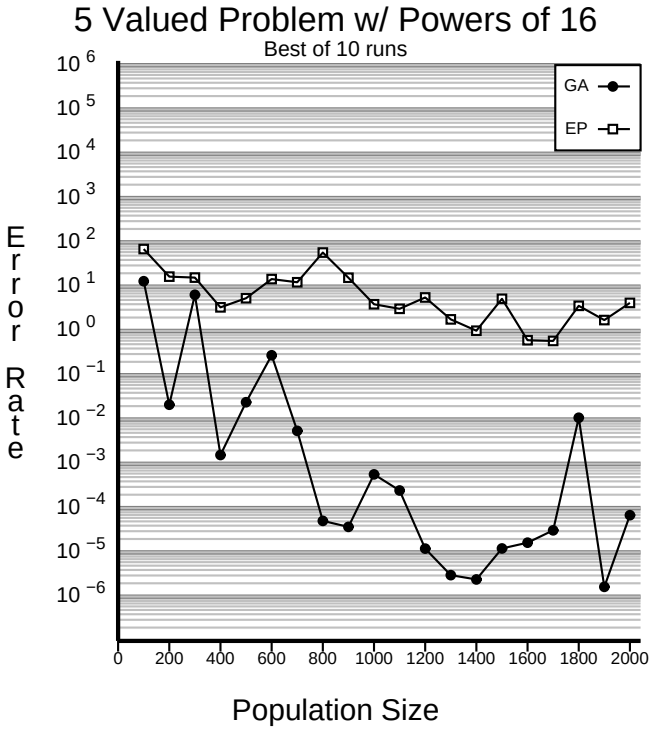

Figure 4. Effects of Population size w/ High Dominance (Best of 10)

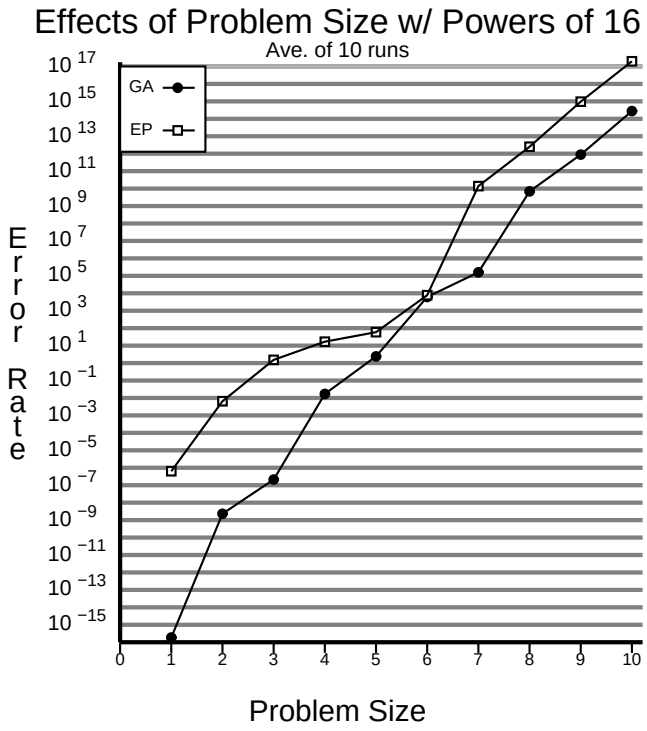

Figure 5. Effects of Problem size w/ High Dominance (Ave. of 10)

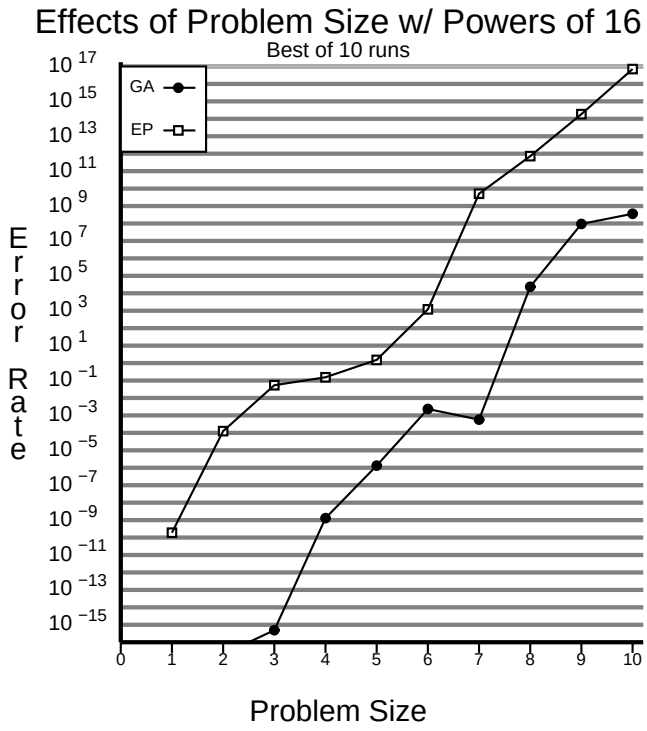

Figure 6. Effects of Problem size w/ High Dominance (Best of 10)

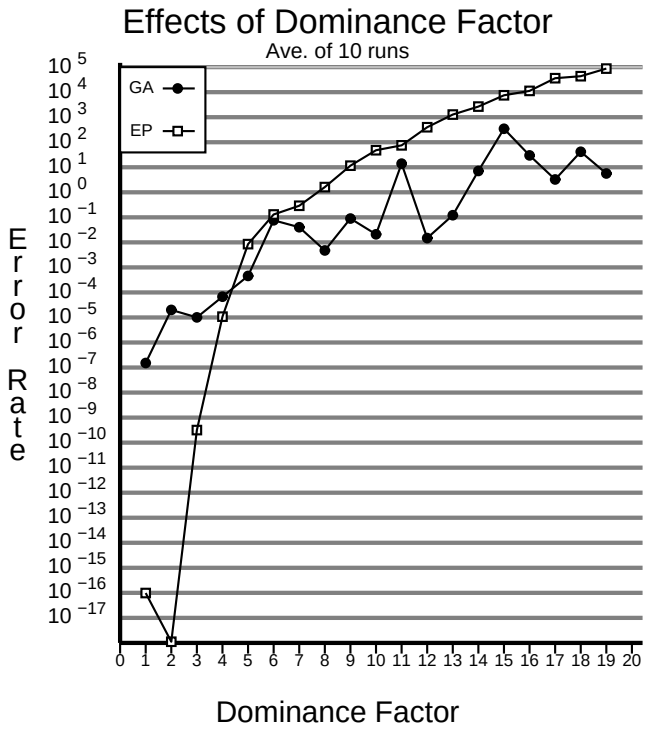

Figure 7. Effects of Dominance Factor (Ave. of 10)

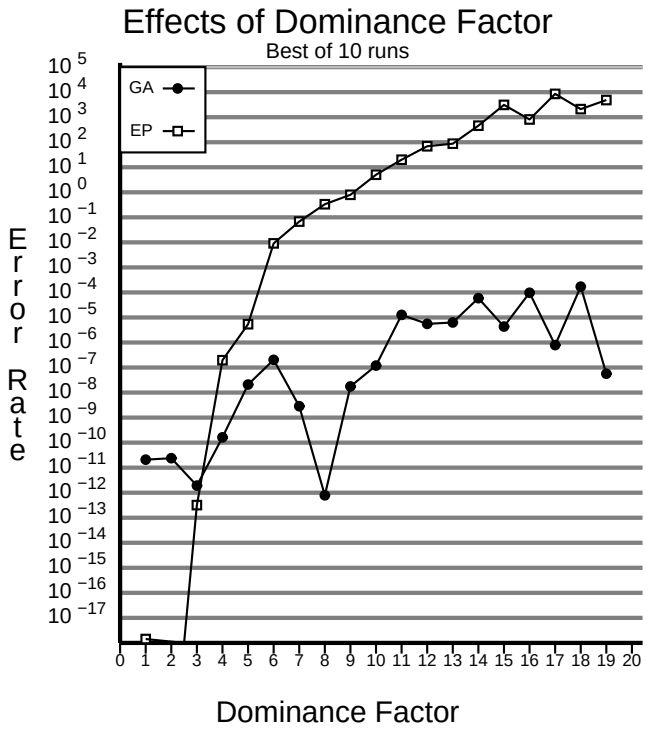

Figure 8. Effects of Dominance Factor (Best of 10)

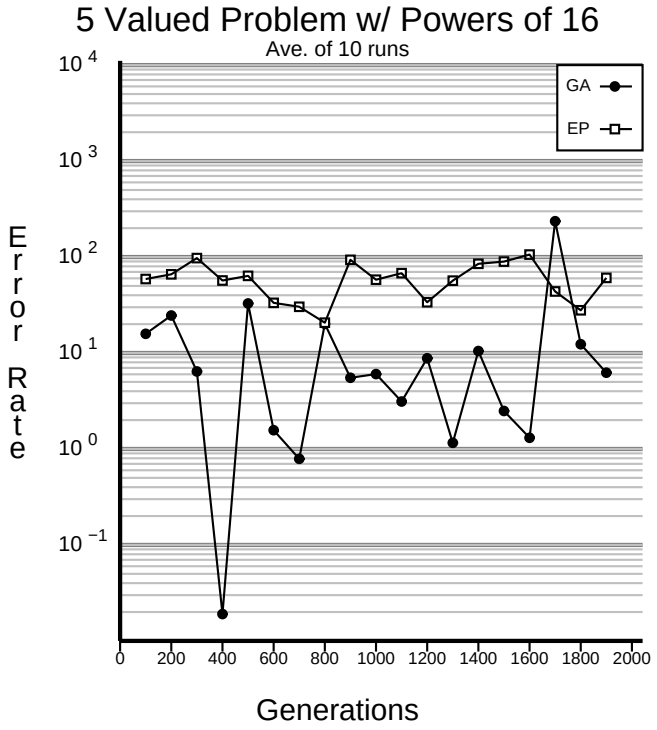

Figure 9. Effects of Run Time w/ High Dominance (Ave. of 10)

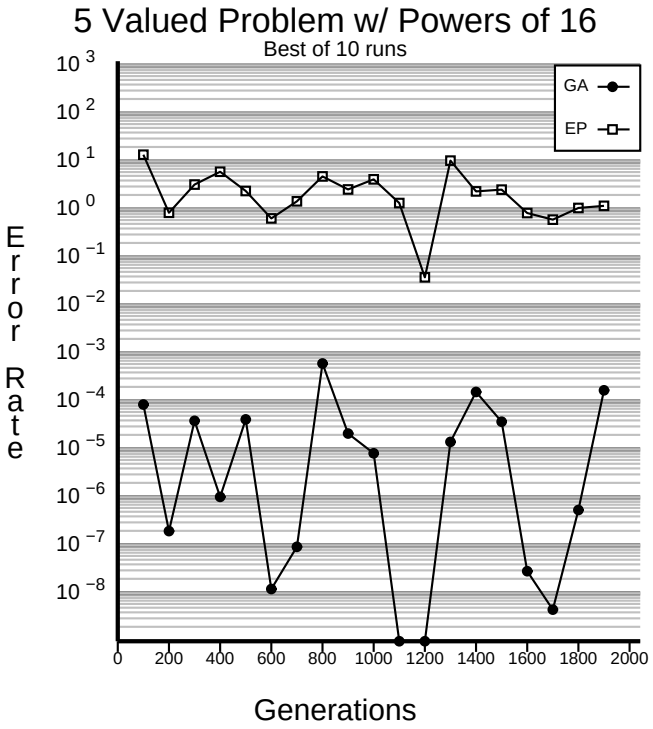

Figure 10. Effects of Run Time w/ High Dominance (Best of 10)

	Pop. Size	1-pt Crossover Rate	Bit Mutation Rate	EP Mutation Scaling	Genera-tions	Problem Size	Dominance Factor
Figs. 1&2	Varies	0.9	0.0001	1.0	500	5	1.0
Figs. 3&4	Varies	0.9	0.0001	1.0e-9	500	5	16.0
Figs. 5&6	2000	0.9	0.0001	1.0e-9	500	Varies	16.0
Figs. 7&8	2000	0.9	0.0001	Varies	500	5	Varies
Figs. 9&10	2000	0.9	0.0001	10e-9	Varies	5	16.0

Table 1: Parameter Settings

the GA which is more strongly influenced by population size. All other factors remained identical to the previous set of experiments. The results are displayed in figures 5 and 6.

To test the response of the two paradigms to increasing magnitudes of parameter dominance, both systems were exposed to a series of problem sets with target vectors of length five and dominance factors increasing from 1.0 to 19.0. All other factors remained as before, except for the EP mutation rate which was varied in inverse proportion to largest constant multiplier being applied to the vector. The results are given in figures 7 & 8.

Finally, to examine the overall effects of search time on the capabilities of the two algorithms, a series of tests were conducted offering from 100 to 2000 generations to each. The average and best of each ten problems were recorded in figures 9 and 10. Note that each point represents a group of ten new problem sets rather than the same ten problems 100 generations later. Table 1 summarizes the parameter settings used in each of these data gathering efforts.

Results

First, we must agree that under conditions of low dominance and moderate problem size EP clearly is capable of producing more precise solutions than an equivalent GA as demonstrated in figures 1 & 2. Further, [Fogel94] demonstrates that EP outperforms GA on the standard DeJong test functions[DeJong75]. This section draws upon the given results to examine the effects of dominance on both paradigms and further discusses the effects of space (population size), search time, and increased problem space size on both paradigms when dominance is high.

Effects of Dominance

Figures 3 & 4 provide evidence that as the dominance factor increases the GA takes a lead over EP in its ability to find an answer. The reason for this turnabout is simply that the EP is unable to tune its mutation rates finely enough to be able to zero in on the values for parameters with lesser impact on the solution. In other words, the noise being applied to the large impact parameters masks any improvement being made on those of lower magnitude. Note that no amount of rescaling of the mutation being applied can rectify this situation. Either the mutation rates become so small that the EP takes an inordinate amount of time to reach a solution (which would create a greater risk of the convergence to a local minima if one existed), or the rate remains high enough to mask further resolution of the parameters. The effect of this masking increases as the level of dominance between parameters increases.

Naturally, we could tailor the mutation rates for EP on a per parameter basis; however, this would require a priori information about the problem domain which cannot be assumed. Attempts to reduce the mutation rate effects through cooling, etc. during the EP evaluation cannot succeed because they overlook the crux of the problem which is that the *noise* being applied to dominant parameters must be reduced while the mutation of less dominant parameters proceeds apace.

For the GA on the other hand, increased dominance only minimally affects the overall performance characteristics. From examination of sample runs, it appears that as the dominance factor increases, the crossover operator allows the GA to quickly "lock in" the range of the most dominant parameter which in turn reduces the noise effects masking the resolution of the lower order parameters. These results are similar to those achieved by Goldberg in considering the "counting ones" problem versus direct "binary encoding" of values[Goldberg89].

Effects of Space

Figures 1-4 show the effects of increasing the population size on the ability of both GA and EP to reach an effective solution. Both systems show increased precision as additional resources are added; however, the GA shows a slightly better payback for the increase in space. Preliminary analysis shows that this is primarily due to the heavy reliance which crossover places on the initial population.

Effects of Time

Surprisingly, increasing the search time for either paradigm beyond an initial 100 generations had little consistent effect on the level of precision. Neither data set in figures 9 & 10 show any obviously separable trends. Under high dominance EP quickly reaches the correct range of the most dominant parameters and then stalls out due to masking effects or greatly reduced effective mutation rate. Likewise GA manages to lock in the best value for dominant feature very early in its exploration, after which the magnitude of later modifications depends solely upon the values which are still present in the population.

Effects of Problem Space Size

Figures 5 & 6 demonstrate that both GA and EP degrade uniformly as the problem size increases. Neither paradigm gains an advantage from increased problem size; nor does it appear that increased problem size negates the effects of dominance.

A Note About Consistency

Through comparison of the average and best solution values for each of the above tests (or through compari-

son of the standard deviation data which was omitted for the sake of clarity), it becomes clear that EP tends toward a much more consistent level of performance for any given constraints. Analysis of individual GA evaluations show that this is largely due to the dependence of the GA paradigm upon the initial population. In general EP solutions can be expected to be within a few orders of magnitude of each other while GA results may vary greatly in magnitude even for multiple evaluations of the same problem.

Conclusions

EP and GA are both equally valid approaches for searching large problem spaces. Each has qualities which are best suited toward certain classes of problems. While EP may dominate as a tool for problems where portions of the solution contribute equally toward the overall value of an individual, it falls behind GA on problems where parameter dominance is a factor. Unfortunately, there is no clear method for predeterminatly selecting which algorithm might be best suited to a given problem space; therefore, consideration of either or both approaches may be appropriate for any given problem.

Future Work

It may be possible to adapt a form of Gaussian mutation for use within a GA framework; thereby potentially freeing the GA from its dependence on initial population. Our future research will focus on the development and subsequent analysis of such an operator.

Both GA and EP make claims of being able to escape local minima. A quantitative comparison of the conditions and influences which enable or hinder each system in escaping local minima might provide insight into the relative merits of the two claims.

Finally, it would most instructive to apply EP to a "real world" domain where GA approaches already exist, or conversely to apply GA to one for which an established EP approach exists. Thus we might be able to take such comparisons beyond the realm of the theoretical and into the practical.

References

Baeck, Thomas (1996) Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming and Genetic Algorithms, Oxford University Press, New York.

DeJong, K. A. (1975) Analysis of the Behaviors of a Class of Genetic Adaptive Systems. Ph.D. dissertation, University of Michigan.

Fogel, D.B. and J.W.Atmar, (1990) "Comparing Genetic Operators with Gaussian Mutations in Simulated Evolutionary Processes Using Linear Systems", Biol Cybern 63:111-114.

Fogel, D.B. and L.C. Stayton, (1994) "On the Effectiveness of Crossover in Simulated Evolutionary Optimization", BioSystems 32:171-182

Fogel, L.J. and D.B. Fogel, (1994b) "A Preliminary Investigation on Extending Evolutionary Programming to Include Self-adaptation on Finite State Machines", Informatica 18:387-398.

Fogel, D.B. (1995) "A Comparison of Evolutionary Programming and Genetic Algorithms on Selected Constrained Optimization Problems", Simulation 64:397-404.

Goldberg, D. E., (1989) Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, Reading, MA.

Goodman, E., (1994) "An Introduction to GALOPPS--The 'Genetic ALgorithm Optimized for Portability and Parallelism' System", Tech. Report., Michigan State University.

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

Koza, J.R. (1992) Genetic Programming: On the Programming of Computers by Means of Natural Selection, MIT Press.