Effects of Diploid/Dominance on Stationary Genetic Search

Proceedings of the Fifth Annual Conference on Evolutionary Programming. San Diego: MIT Press.

F. Greene
6920 Roosevelt NE #126
Seattle, WA 98115
bgreene@accessone.com

Abstract

Evidence suggesting that a gene-fitness based approach to diploid/dominance may provide resistance to premature convergence is presented. The algorithm used is compatible with any scalar and stationary fitness function and is based on well established population genetics theory.

Introduction

In nature the vast majority of animals and plants utilize a *diploid* chromosome structure. Discussions of its use in genetic algorithms can be found in (Holland, 1975) and (Goldberg, 1989). More recently, (Smith, 1992) described experiments that follow Holland's one string position per allele representation. Those results showed some increased ability to recall previous string element values in a non-stationary fitness environment using diploidy, along with a careful analysis of expected diploid vs. haploid behavior.

Diploid/dominance expression in biology suggests that two homologous alleles are paired and, through a hidden mechanism, the dominant gene is phenotypically expressed. Only if both genes are of the *recessive* phenotype, from an observational standpoint, is the recessive gene expressed. In (Hollstien, 1971), a genetic algorithm was used to implement such a "mechanism" by increasing the cardinality of the chromosome bit string from 2 to 3. If a "2" occurred at a given string position, it was interpreted as a "1", but only if paired with a "1" or "2" in its homologue. Assignment of dominance can be modified during the course of a GA run with a random "dominance shift" operator.

The approach described here is based on principles in the field of population genetics and biochemistry. The biological scenario corresponds to a primitive (e.g., prokaryote) organism in which recombination routinely takes place *within* genes. This differs conceptually from the common GA model where recombination is only allowed *between* genes. The difference being to what the word gene and allele refer: In the traditional case gene or allele usually refers to a single string position -- here a gene or allele may be a contiguous string sub-section, potentially including the entire string in a simple single-locus organism.

Natural Diploid/Dominance

It is widely understood in biology that diploid dominance occurs for very simple reasons. Dominance can be resolved at the gene level through the viability of the resulting protein, which may be a structural protein, r-RNA, or enzyme. This is most readily seen to occur with a substrate limited enzyme reaction. In this case, one working gene is enough to produce a viable protein and having two may cause a relatively insignificant increase in the amount of protein product, or gene fitness (Crow, 1983). The first clear discussion of enzyme dominance is in (Muller, 1950). Even prior to this, however, the *behavior* of single-allele diploid dominance behavior was made clear by (Haldane, 1927) who was aware of the mathematics involved. A relevant discussion of the following results can also be found in (Li, 1954, pg. 283).

A recessive *mutant* allele when paired with one of its own kind might have fitness compared to the other two possible allele combinations as follows:

AA	Aa	aa
1	1	1-s

where $0 \le s \le 1$, "a" is the mutant (actually any dysfunctional version of the gene) allele and "A" is the normal, functional or *wild type*. "Normal" genes may have differing genotypes so long as they are functional. If the heterozygote fitness is equal to the homozygote AA fitness (as it is in this case), then we can write the "A" gene frequency for the next generation as:

$$p_A^{'} = \frac{p_A(p_A + (1-p_A))}{p_A^{\;2} + 2p_A(1-p_A) + (1-p_A)^2(1-s)}(1-\mu), \qquad (1)$$

where μ is the rate at which A's mutate to a's, and an infinite population size is assumed. Considering that a given string position change may or may not result in gene dysfunctionality, μ can be greater or less than p_m, the string position-wise mutation rate. Solving for the steady state proportion of a's, or p_{SS}:

$$1 - s(1-p_{ss})^2 = 1-\mu,$$

resulting in:

$$p_{ss} = \begin{cases} \sqrt{\mu / s} & for\ \ s > \mu \\ 1 & or\ \ s \le \mu \end{cases} \qquad (2).$$

Equation 2 results from the fact that a's only get exposed to selection according to their independent and joint occurrence in the population. The upper bound is never reached unless *s* is so small that it cannot overcome the mutation rate. It is assumed that the rate of dysfunctional to functional gene mutations is negligible. By contrast a *haploid* organism with fitnesses:

A	a
1	1-s

would have p_{SS}= μ / s . Similar quadratic relationships are derived in (Smith, 1992) for binary string values.

The value "s" can be interpreted as the death rate per generation for recessive "a" alleles in the above haploid model. The average haploid mutant will then persist in the population for 1/s generations. In the diploid case, "a"'s only get subjected to selection when paired with each other, so the corresponding diploid persistence is $1/(s \cdot p_{ss})$ generations. The ratio of diploid to haploid persistence is therefore:

$$r = \begin{cases} \sqrt{s / \mu} & for\ \ s \ge \mu \\ 1 & for\ \ s < \mu \end{cases} \qquad (3)$$

and may be significantly greater than one. In general, if multiple alleles are considered, this behavior can occur at all allele positions independently. In summary there is *no requirement for multiple alleles* in an organism to gain the protection indicated above to a low fitness diploid homologue.

If the heterozygote fitness is reduced by a non-zero value of "h" as follows:

AA	Aa	aa
1	1-hs	1-s,

the square root behavior is retained for small values of h. (Felsenstein, 1995, pg. 109) shows that a sufficient condition for this behavior is $h << \sqrt{\mu / s}$. For $h >> \sqrt{\mu / s}$ P_{ss} can be approximated by μ / hs until for values of h approaching 1, the persistence of mutants in succeeding generations are identical to that stated above for haploids.

Completely recessive behavior (h → 0) will be modeled here with the fitness function:

$$f = Max(f_1, f_2) \qquad (4)$$

where, the f_i are the problem defined fitnesses of each homologous chromosome, or in this case allele. For a given individual, it is apparent that its fitness with equation (4) will be relatively low only if *both* homologues have relatively low fitness. Conversely, if even one has a relatively high fitness the lower-fitness allele is *completely* shielded from selection, corresponding to the above heterozygote condition with h = 0. Use of equation (4) was described in (Greene, 1994) using a non-stationary 0-1 knapsack problem. That report is a comparison with specific results in (Smith, 1992), which utilized Hollstien's triallelic approach. In contrast, the purpose here is to present the use of equation (4) with a typical, stationary fitness function.

Methods

The diploid chromosome model used here has two chromosomes with one gene and one corresponding allele value each. The allele values are defined to be the fitness values of the two strings, which are evaluated according to the problem at hand. The haploid chromosome with the higher fitness then determines the fitness and survivability of the diploid individual.

In biology, crossover occurs prior to gamete formation. Here the opposite is done -- namely the gametes are randomly paired up and crossover is then done, so that crossover computation is not wasted on unused gametes. In addition, biology typically has a maximum recombination rate of 0.5 whereas here the rate can be, and with the Genitor algorithm *is*, 1.0. More details of the algorithm can be found in (Greene, 1994).

The second homologue of each individual in the *initial* diploid population is generated by making a *direct copy* of each randomly generated first homologue, as opposed to randomly generating the 2nd homologue. As a result the diploid experiments have an initial allele diversity that is exactly equal to their haploid counterparts.

Multiple Trial, Multimodal Test Function

The Genitor program (Whitley, 1989) was used with modifications to support diploidy (Greene, 1994). Values of p_m= .05 and a linear fitness bias of 1.025[1] were used. Ten trials of the test problem were made using different randomly generated populations. These results used the random number generator supplied with Borland C++ 3.1 which is multiplicative congruential with period 232. The same random number seed is used for a given diploid-haploid comparison, so each diploid and haploid population of equal size initially contain exactly the same chromosome values. As the diploid GA runs, the initially identical homologues will increasingly differ, and there will be an increase in diversity as this occurs. For this reason, diploid populations of size N=100 are compared in terms of search time efficiency both to haploid populations of size N= 100 and N= 200.

A multimodal/partially deceptive test function was devised that is both fast to evaluate and easy to define and therefore replicate. The fitness function to be minimized is:

$$f(x) = x \bullet [\alpha - \cos(\pi T x)], \quad 0 \le x < 2^{30} \quad (5)$$

The difficulty of the problem can be adjusted with the α parameter, if necessary, so that it is in a range that is neither too difficult (in which case neither haploid or diploid reach a solution in a reasonable time) or too easy (in

[1] A linear bias defines the weighting by which the ranked population of individuals are randomly selected for mating. A bias of b= 1.025 corresponds to growth rate ≈ 2.8 [Goldberg, 1991 #43, pg. 84].

which case no difference is seen.) All experiments here used α= 1.1. Equation 5 has monotonically decreasing minima (corresponding to fitness maxima) with decreasing binary chromosome value x and a global minimum (or fitness maximum) at x= 0 as shown in Figure 1. Other functions considered were Goldberg's concatenated "order-3" and "order-5" deceptive functions (Goldberg, 1993, pg. 10). In initial experiments, the order-3 problem was "too easy" and the order-5 too difficult to see a difference between haploid and diploid. That is, haploid vs. diploid comparisons either converged in about the same, small number of function evaluations or didn't converge to a solution in a reasonable time. Equation 5 is intended to model a well designed, but moderately difficult application, where there are multimodal and deceptive regions in the search space.

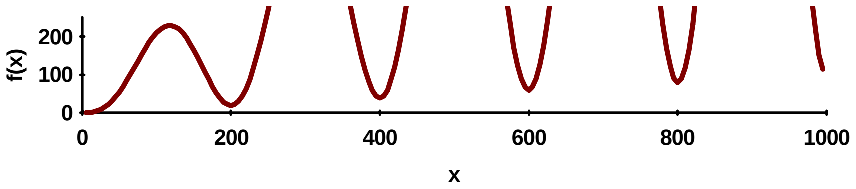

Figure 1. Fitness function showing decreasing local minima towards global optimum (α=1.1, T=100.)

Results

Results are shown below as the ratio of average fitness function *evaluations*. This takes into account the diploid implementation used here which makes two function evaluations per Genitor iteration. With half-period T= 100, the evaluation of 10 independent initial populations for each haploid and diploid experiment resulted in ratios of average required evaluations, to zero error, as shown in Table 1. For the multiple trial haploid experiment having twice the population size of its diploid N=100 counterpart, the ratio between the average number of required function evaluations for diploid and haploid populations was approximately 4.5 to 1. In order to compare larger and smaller haploid population sizes with diploid N= 200, haploid experiments with N= 100 and N= 400 were done. The ratio of average function evaluations in these cases were 4.9 and 1.0 to 1, respectively.

Numerator → ↓ Denominator	Haploid, N=100	Haploid, N=200	Diploid, N=100	Diploid, N= 200
Haploid, N= 200	1.1	-	-	-
Diploid, N=100	4.9	**4.5**	-	-
Diploid, N= 200	5.8	5.3	1.2	-
Haploid, N= 400	5.0	4.5	1.0	0.9

Table 1. Ratio of Average Fitness Function Evaluations, 10 Trials, T= 100

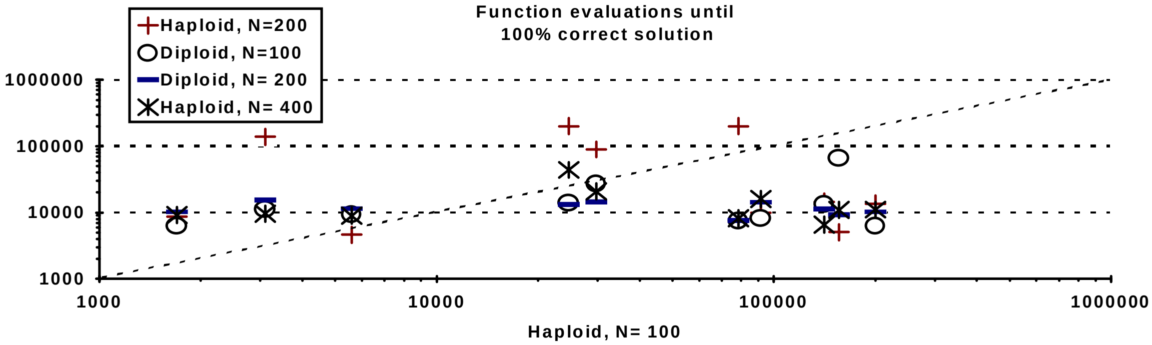

Figure 2 . Scatter gram of GA evaluation time for five combinations of haploid and diploid population sizes: Ten trials were done on each combination using equation (5) with T=100.

The longest convergence times occur with haploid N= 100 and N= 200, as shown in Figure 2, with 3 trials in each case exceeding 100,000 evaluations. The scatter of haploid N=200 is roughly symmetrical with that of haploid N=

100. Diploid N=100 is consistently better than haploid N=100 except at convergence times less than 10000 (for which the initial population distributions were apparently favorable to both haploid and diploid.) Having noted that the same initial number generator seeds were used for each haploid N= 100 and 200, and diploid N= 100 comparison, there appears to be protection against the largest haploid convergence times using diploid that is not merely due to the diploid doubling of string material.

Multiple Trial, Multimodal Test Function: Comparisons with Two Additional Values of T
To further assess robustness of the approach, the previous experiment was repeated with two additional values of T, and their efficiencies are summarized in Table 2. In addition, the percent of trials in each set of ten that converged (to the global optimum) within 400,000 trials are given. As we might expect from equation (3) diploid *tends* to outperform haploid in terms of efficiency, and usually does no worse.

T= 100:	Haploid, N=100 / 100%	Haploid, N=200 / 100%
Diploid, N=100 / 100%	4.9	4.5
T= 200:	Haploid, N=100 / 80%	Haploid, N=200 / 90%
Diploid, N=100 / 100%	> 2.0	> 1.2
T= 500:	Haploid, N=100 / 60%	Haploid, N=200 / 60%
Diploid, N=100 / 100%	> 2.6	> 2.6

Each Heading: Experiment / % Trials Converging to Zero Error

Table 2. Summary of three experiments, ten trials, each using equation (5) and three values of T.

Conclusions

Based on equation (3) an appropriately designed GA problem that follows biology *should* show improved efficiency with diploid vs. haploid. On average, the diploid implementation described here does no worse and at times does significantly better than haploid in terms of the number of function evaluations needed to locate the global optimum. No diploid trials exceeded 400,000 function evaluations. By comparison, six haploid N= 100 and five haploid N= 200 trials exceeded this pre-determined limit. The results are generally consistent with the expected relative persistence of mutant alleles in diploid vs. haploid populations given by equation (3).

These results suggest that an improvement in GA efficiency with diploid dominance is possible, compared to haploid. It should be noted that improvement may not occur in a given problem if the problem difficulty is either too high or too low for the chosen population size, selective pressure or mutation rate. A similar caveat exists if a sufficient range of chromosome string element values does not exist in the initial population, as suggested by the multiple trials. These same statements could be said to apply to any GA enhancement, however. The results should be of interest in that for a problem of modest GA difficulty, improvement in the form of some protection against premature convergence appears to occur using the $\text{Max}(f_1,f_2)$ approach to diploid/dominance. Finally, it should be pointed out that diploidy, particularly as implemented here, does not preclude improvements to other aspects of GA's, such as crossover, chromosome encoding, etc.

Most GA's use a scalar fitness function that corresponds to the definition of "single allele" used here. A multiple allele (locus), approach, in this sense, could correspond to a vector fitness function where the alleles have "sub"-fitnesses that could be mapped to a scalar possibly according to their importance. If the multiple fitnesses were not independent by design, then additional interactions such as gene regulation would be possible.

Future work may simulate migration by infusing mutant (e.g., random) chromosome values, either periodically or more artificially when premature convergence is detected. Migration can introduce new alleles into an otherwise prematurely converged population. In a diploid population such potentially and likely recessive alleles can be shown to persist longer, in general, than in a haploid population. Further improvement might also be seen with time varying fitness, although doing this complicates fitness function design. The $\text{Max}(f_1,f_2)$ approach is currently being used in a genetic programming application using real chromosomes.

Bibliography

Crow, J. F. (1983). Genetics Notes (8 ed.). Minneapolis: Burgess.

Felsenstein, J. (1995). Theoretical Evolutionary Genetics. Seattle: ASUW Publishing.

Goldberg, D. E. (1989). Genetic Algorithms in Search, Optimization & Machine Learning. Menlo Park: Addison-Wesley.

Goldberg, D. E., Deb, K., Kargupta, H., Harik, G. (1993). Rapid, accurate optimizationo of difficult problems using fast messy genetic algorithms (IlliGAL No. 93004). Univerity of Illinois.

Greene, F. (1994). A method for utilizing diploid/dominance in genetic search. In World Conference on Computational Intelligence, ICEC-1 (pp. 439-444). Orlando, FL: IEEE.

Haldane, J. B. S. (1927). A mathematical theory of natural and artificial selection. Part V: Selection and mutation. Proc. Cambridge Philosophical Society, 23, 838-844.

Holland, J. H. (1975). Adaptation in Natural and Artificial Systems. Ann Arbor: The University of Michigan Press.

Hollstien, R. B. (1971) Artificial genetic adaptation in computer control systems. Doctoral Dissertation, University of Michigan.

Li, C. C. (1954). Population Genetics. Chicago: University of Chicago Press.

Muller, H. J. (1950). Evidence of the precision of genetic adaptation. In C. C. Thomas (Eds.), The Harvey Lecture Series (pp. 165-229). Springfield, Illinois:

Smith, R. E., Goldberg, D.E. (1992). Diploidy and dominance in artificial genetic search. Complex Systems, 6, 21-285.

Whitley, D. (1989). The Genitor algorithm and selection pressure: why rank-based allocation of reproductive trials is best. In Proceedings of the Third International Conference on Genetic Algorithms, (pp. 116-123). George Mason University: Morgan Kaufmann.