Dynamic Parameter Encoding for Genetic Algorithms *

Nicol N. Schraudolph **Richard K. Belew**
nici@cs.ucsd.edu *rik@cs.ucsd.edu*
Computer Science & Engineering Department
University of California, San Diego
La Jolla, CA 92093-0114

July 20, 1992

Abstract

The common use of static binary place-value codes for real-valued parameters of the phenotype in Holland's genetic algorithm (GA) forces either the sacrifice of representational precision for efficiency of search or vice versa. Dynamic Parameter Encoding *(DPE) is a mechanism that avoids this dilemma by using convergence statistics derived from the GA population to adaptively control the mapping from fixed-length binary genes to real values. DPE is shown to be empirically effective and amenable to analysis; we explore the problem of* premature convergence *in GAs through two convergence models.*

1 Motivation

Genetic Algorithms (GAs) are capable of efficiently searching complex fitness landscapes utilizing methods inspired by biological evolution [7]. The work reported here used the GAucsd[1] software which implements a popular form of the GA, applying the operators of selective reproduction, mutation and crossover to a fixed-length binary genome. While the common restriction to fixed-length genetic representations in GA experiments is rooted in convention and convenience, theoretical arguments have been made in support of binary alphabets [4, p. 80–82].

If the function to be optimized by the GA is defined over a continuous domain, its real-valued parameters can only have approximate genotypic representations as binary codes of a given length. The choice of code length determines not only the resolution of the GA search for this gene and the precision of its result[2], but also the size of the space of representations the GA has to search.

Unfortunately the GA searches long place-value codes rather inefficiently: in the early phases of the search a lot of computational effort is invested in evaluating the least significant digits of the gene. The optimal value for these digits, however, will typically depend on the more significant ones, and little useful information can be obtained from sampling the former until the latter have converged to

*Reprinted from *Machine Learning* 9, 9–21 (1992).

[1] GAucsd is our variant of the GENESIS package [5] supporting — among other enhancements — user-transparent DPE. It is a public domain program and can be obained via Internet file transfer or electronic mail from the first author.

[2] We use the term *resolution* to denote the number of alleles in a gene, and *precision* for the distance (on the real line) between neighboring alleles.

their optimal value. Likewise, once this has happened there is no need to spend any more search effort on the most significant digits — the plain GA though will pay equal attention to all digits of the representation throughout its search, thus wasting its *implicit parallelism*[3] on an inherently sequential search problem.

Moreover, in the absence of meaningful fitness information early in the search, the least significant bit positions will converge to an arbitrary value due to *genetic drift*, a process aggravated by *genetic hitchhiking* — incidental associations with highly successful alleles in the more significant bits (see Section 5.2). Once these bits have converged the GA cannot search them efficiently anymore; thus the GA search progressing sequentially from the most significant bits can only achieve a limited resolution before it is beached by the premature convergence spreading from the less significant section of the gene.

In this situation one strategy to obtain a high-precision result might be to use a very crude precision at first and allow the GA to converge. At this point the resulting population could be *re-encoded* onto the same set of bits such that search is restricted to the identified region, and the GA restarted. This process could then be iterated until the result has the desired precision. In essence, this is what Dynamic Parameter Encoding (DPE) does autonomously and for each real-valued gene in parallel. By adaptively controlling the encoding scheme in this way, DPE demonstrates that a sacrifice of search resolution need not necessarily lead to an identical loss of precision in the search result. This correspondence is just an artifact of the assumption — implicit in many implementations of the GA — that the encoding scheme remains constant throughout the search.

The next two sections of this paper describe the DPE algorithm and demonstrate its effectiveness, respectively. We then contrast DPE with related approaches to the problem of real value representations in GAs, and the ARGOT strategy [11] in particular. This comparison in turn motivates an analysis of GA convergence which attempts to quantify the conditions under which the DPE algorithm succeeds.

2 The DPE Algorithm

For the purpose of evaluation, each parameter of a fitness function f has to be recovered from its representation as an l-bit binary gene. The first step of this decoding process is to interpret the gene as an unsigned binary (possibly Gray-coded) integer i. A real-valued parameter in the *search interval* $[a, b)$ can then be derived through the linear[4] transformation

$$T(i) = a + (b-a)\frac{i+X}{2^l} \tag{1}$$

where X is a random value with uniform density over $[0, 1)$ included in order to eliminate potential aliassing effects caused by the quantization error inherent in the representation. This means that T maps each allele onto a subinterval of length $(b-a)/2^l$ instead of just a single point on the real line; the effective allele fitness measure is therefore the Monte Carlo integral of f over the corresponding subinterval.

The GA can identify the optimum of f in $[a, b)$ only if it is located in the subinterval whose fitness integral is optimal. The number of subintervals must be large enough for this to happen, thus imposing a lower bound on the length l of the gene — we call this the

[3] This implicit parallelism [7] occurs in the evaluation of low-order schemata, ie. exactly where it is not useful in case of a standard place-value representation.

[4] Other (e.g. logarithmic) representation schemes can be implemented by suitably post-processing $T(i)$.

minimal resolution required by f for searching this gene.

The length of each subinterval *at the end of the search* determines the precision of the result obtained. For static encoding schemes a *minimal precision* requirement therefore imposes another lower bound on l. In most cases this is the bound that limits the efforts to improve the GA search by reducing gene length: often comparatively coarse search resolution would be sufficient to identify the optimal subinterval but we need to know more precisely where in that subinterval the optimum is located. DPE addresses this need by remapping the gene such that the available search resolution is reused on a finer scale in an optimal subinterval once that has been identified. By applying this process repeatedly DPE facilitates final results of any desired precision independent of gene length, as long as the minimal resolution requirement is met.

DPE treats each real-valued parameter of f identically and independently; Figure 1 illustrates its operation for one such gene. Each generation the DPE algorithm constructs a histogram of the population over the quarters of the current search interval formed by the two most significant bits of the gene. By summing over neighboring quarters we obtain the population counts for three overlapping *target intervals*, then take their exponential traces. Such exponential historic averaging — whose time constant is a free parameter of DPE — makes these counts less sensitive to the noise inherent in GA search than the raw target histogram, hence more reliable as an indicator of stable convergence. If the largest count exceeds a given *trigger threshold* we consider the population converged on the corresponding target interval and invoke a *zoom operator*.

The zoom operation restricts subsequent GA search to the target interval by suitably altering the decoding transformation T (Equation 1). The binary gene of every individual in the population is then manipulated such that those lying inside the target interval are left in place by the zoom[5]. Those lying outside are "folded" onto the target interval in the process; this method is less disruptive than random re-initialization in that it preserves the existing schema frequencies.

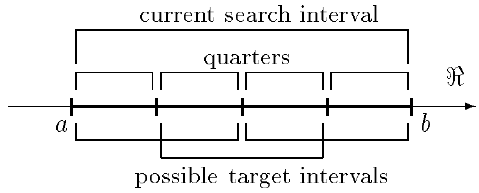

Figure 1: Intervals used by the DPE zoom.

Note that the overlap of target intervals eliminates the risk of zooming into the wrong target interval due to small sampling errors in case the optimum is close to a target interval boundary. Since the zoom operator doubles the precision of the gene, each allele can be assigned to one of two new subintervals during the remapping; this choice is made by an independent coin toss for each individual, i.e. a random re-initialization of the gene's least significant bit.

3 Empirical Results

To verify the efficiency of the DPE algorithm we checked its performance on a classical GA test function suite [2]; see Table 1 for number of genes d, length l (in bits) of each gene, and

[5]This is achieved with a bit shift left plus some additional manipulations in the two most significant bit positions.

the name of each test function to be minimized.

Table 1: De Jong's GA test function suite.

fn	d	l	Comment
f1	3	10	paraboloid
f2	2	12	Rosenbrock's saddle
f3	5	10	step function
f4	30	8	x^4 with noise ($\sigma = 1$)
f5	2	17	Shekel's foxholes

In a large empirical study Schaffer et al. have identified values for population size, crossover and mutation rate that produce good online performance on this test suite [9]; we are using their results (which are almost identical to earlier recommendations by John J. Grefenstette) here, and also follow their practice of Gray coding all genes [1]. To keep the experiment as straightforward as possible we did not follow the common practice of using an *elitist* selection strategy[6], and kept the following parameters invariant unless noted otherwise:

Population Size:	30
Crossover Rate:	0.95
Mutation Rate:	0.005
Generation Gap:	1.0
Scaling Window:	1
Trigger Threshold:	0.9
Trace Time Constant:	10

The search resolution was held constant at $l = 3$ for all experiments using DPE to facilitate comparison of results across the test suite; the control experiments used the values given in Table 1. The results are shown in Figure 2, which traces $\bar{f}$, the average value of

[6] Later control experiments with elitist strategy produced very similar results.

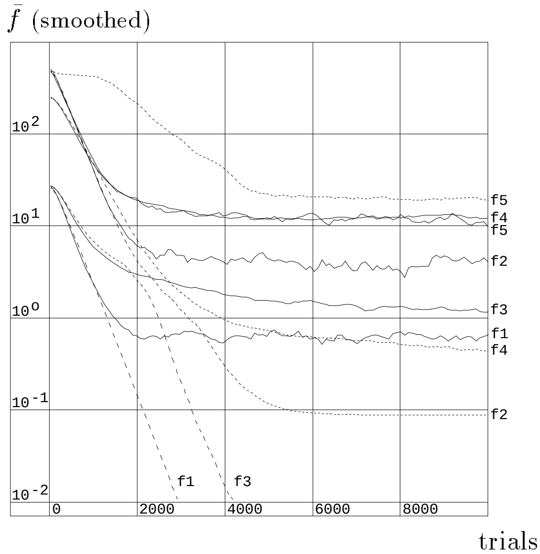

Figure 2: Performance for De Jong's functions with DPE (dashed) vs. without (solid).

the objective function in the population[7] over 10 identical runs (with different initial random number seed) and 10000 evaluations of each function; experiments using DPE (dashed) are compared to those that did not (solid) on a logarithmic scale. Other experiments showed that the DPE parameters can be optimized further for each individual function; a few such cases are reported below.

The positive effect of DPE is most evident on functions f1 and f3. For them the experiments both with and without DPE show a similar initial performance improvement; without DPE, however, the curve soon bottoms out as it reaches a level where the mutation noise becomes dominant. With DPE, each zoom operation reduces the phenotypic effect of a muta-

[7] For display purposes this performance measure was smoothed through an exponential trace with a time constant of 10 generations.

tion, thus keeping mutation noise in check.[8] The result is continuous performance improvement over many orders of magnitude: the average function values after 10000 trials with DPE were $6 \cdot 10^{-10}$ and $4 \cdot 10^{-7}$ for f1 and f3, respectively.

For f2 and f4 DPE initially manages to significantly improve performance, too. After 4000 evaluations, however, it levels out in both cases, albeit at a far better level than without DPE. This happens because both functions have their optima located on an extremely flat ridge (f2) or plateau (f4). In such a case, once DPE gets close to the optimum the fitness differences become so small that mutation noise prevents sufficient convergence to trigger the zoom operation.

Lowering the mutation rate does reduce mutation noise but simultaneously exacerbates the problem of premature convergence. Although measures favoring exploration over exploitation (e.g. large populations, high crossover rates, low selective pressure) can be used to combat premature convergence, they inherently slow down the search process — the gained precision is thus bought at a high computational price.

Figure 3 illustrates that even with more than tripled population size and crossover rate[9] no choice of mutation rate enables the plain GA to optimize the quadratic function f1 — trivial by optimization standards — to any high precision. Note that the initial rate of performance improvement is now significantly less than in Figure 2 due to the more exploration-oriented parameter settings. For comparison, DPE with trigger threshold 0.8 and a trace constant

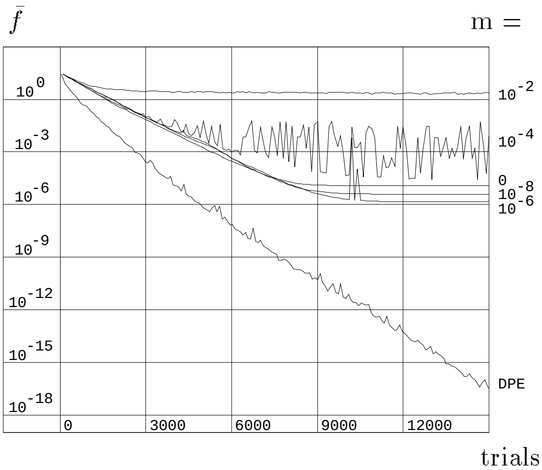

Figure 3: Performance of plain GA on f1 for various mutation rates, contrasted with DPE performance.

of 12 generations supplies average results well below 10^{-16} (below 10^{-21} in some runs) in the same number of trials.

Returning to Figure 2, we note that with our default set of parameters DPE actually worsens the performance for f5, the only multimodal function in the test suite. However, this merely demonstrates that there is no universally optimal parameter setting for the GA[6]; in this case three bits per gene do not provide enough search resolution for DPE to evaluate which of f5's 25 optima is the best. Indeed, when allowed a resolution of six bits per gene[10], DPE consistently converges on the global optimum within 18000 trials, whereas the plain GA's performance is still an order of magnitude away after 25000 evaluations (Figure 4).

Note that over the first 13000 trials (where no performance advantage is visible) DPE

[8] A more elaborate example of this approach is the adaptive control of mutation variances in evolutionary strategies [10].

[9] Population size 100, crossover rate 3.0 — that is, three independent two-point crossovers for each pair of parents.

[10] To compensate for the enlargened search space we also increased the trace time constant to 50 generations; all other parameters remained the same.

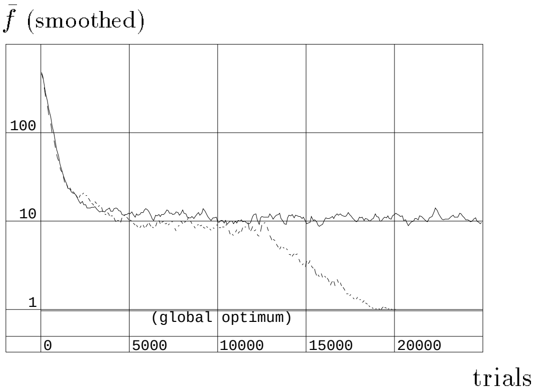

Figure 4: Double resolution search of f5 with DPE (dashed) vs. without (solid).

gradually eliminates all non-global optima from consideration; once the search space has become unimodal DPE rapidly homes in on the target and the population promptly converges to the global optimum. This strategy is quite appropriate in that it puts most of the computational effort into the hard problem of deciding which basin of attraction is the best, then wastes little time on the comparatively easy task of converging on the identified optimum.

Since De Jong's test suite provides with f5 only one multimodal function, we performed additional tests on the following problem [10]:

$$f7(\vec{x}) = -\sum_{i=1}^{10} x_i \sin\sqrt{|x_i|}\,, \quad |x_i| \leq 500. \qquad (2)$$

The best of the over 10^9 local optima of this function are located far from the global optimum yet are almost as attractive (within 3% of optimal), making it especially hard for DPE to prune this search space.

A "fast" zoom operator (trace time constant: two generations) with a high trigger threshold of 0.95, however, was still able to consistently identify the optimal basin of attraction of f7 within 8000 evaluations in experiments using a population size of 100. This can be achieved with the plain GA just as well, but any low-order bits in the genome will have converged by that time, making further progress to the actual optimum (via mutation only) very difficult. DPE makes the transition from global to local search far more gracefully, locating the optimum to any desired precision.

4 Previous Work

We are by no means alone in suggesting that certain aspects of genetic encoding schemes in general, and those for real numbers in particular, should be under adaptive control of the GA itself — Deb for instance has investigated floating point optimization using a GA with "messy" representations as advocated by Goldberg [3].

Shaefer in particular has influenced our work with his GA-based function optimization strategy ARGOT [11], whose many heuristics include an operator very much like the zoom operator described here. However, ARGOT also features an "inverse zoom" operator that expands the search space if there is little convergence. While such an ability to backtrack would certainly be very useful as an insurance against misguided zoom operations we believe it impossible to establish a well-founded, general trigger criterion for this operator.

Consider the experiment reported in Figure 4 for instance. Although convergence was very low for the first 13000 trials, this did not mean that the global optimum of f5 wasn't in the search space — it was just hard to find. Other examples can easily be found to show that the absence of convergence is indeed neither a necessary nor a sufficient indicator of the situation that warrants an inverse zoom, viz. the absence of the global optimum from the current search space. Moreover, *any* potential trigger criterion for the inverse zoom has to be based upon available information. However, since the GA does by definition not provide any information about the fitness landscape outside its search space, we are forced to conclude that in general there is no indicator for the presence of a global optimum outside the search space.

Two more approaches closely related to ARGOT and DPE have recently come to our attention: the Adaptive Search Space Scaling algorithm applied to medical image registration [8], and the technique of Delta Coding [12]. Like ARGOT, these methods employ a form of inverse zoom that makes implicit assumptions (such as piecewise monotonicity) about the function being searched, thus sacrificing generality for performance. By contrast, the heuristic DPE employs for deciding when to trigger a zoom operation is based on the same termination criterion typically used with the plain GA: stable convergence of the population. In this sense DPE converges to the global optimum if and only if the plain GA does, and is therefore best characterized as a succession of GAs over search spaces iteratively refined through bisection.

This strong reliance on the plain GA makes DPE highly amenable to standard GA analysis techniques such as schema analysis; the price to pay for it is the irreversibility of the zoom operator as argued above. Consequently, DPE also inherits the GA's *premature convergence* problem, which we address by applying the zoom operator conservatively. In fact, much of the work reported here can be viewed as an attempt to make the zoom operator *safe* — cf. the overlap of target intervals and the smoothing of convergence statistics. In order to provide guidance in setting the DPE algorithm's parameters such that DPE zooms as rapidly as possible without converging prematurely we now take a closer look at the mechanics of GA convergence.

5 Models of Convergence

With or without DPE, the GA can converge to the global optimum only if the gene length provides sufficient resolution, so that the size of the subspace corrsponding to each genotype is small enough for its fitness integral to reliably predict its optimum. By presenting two models of convergence in GAs we now attempt to provide similar guidelines for setting the two parameters DPE introduces, namely trigger threshold and trace time constant.

5.1 Expected Convergence Level

The zoom operator should be triggered whenever a gene has converged on one of the three target intervals. In the presence of mutation, however, convergence is never complete; rather, the opposing forces of selection and destructive mutations lead to an equilibrium in which the frequency of any schema depends on both its fitness and the mutation rate. In order to formulate a sound trigger criterion for the zoom operator it is therefore necessary to take the mutation rate m into account.

Consider a single bit position anywhere in the genotype and let r_0 be the population relative schema fitness of the zero allele in that position, i.e. $r_0 = f_0/\bar{f}$. In each generation the proportion p_0 of these zero alleles in a GA population will change by a factor of r_0 through fitness-proportional selection. Mutation will then destroy a fraction m of the zeroes; however, it will also create new zeroes through back-mutation. These effects[11] com-

[11] We deliberately ignore the linkage effects of crossover here; they are the focus of the next section.

bine to yield the expected change of proportion of zero alleles during one generation:

$$\Delta p_0 = (r_0 - 1)p_0 - m r_0 p_0 + m(1 - r_0 p_0) \quad (3)$$

Equilibrium is reached when $\Delta p_0 = 0$. Substituting into Equation 3 and solving for p_0 we find that the equilibrium proportion $\tilde{p}_0$ of zero alleles in the population as a function of mutation rate is

$$\tilde{p}_0(m) = \frac{m}{(2m-1)r_0 + 1} \quad (4)$$

Since the relative schema fitness r_0 depends on the population fitness average $\bar{f}$, which in turn depends on the composition of the population, Equation 4 describes the level of allele convergence only implicitly. It can, however, be recast in terms of the absolute fitness ratio $r = f_0/f_1$ by using the fact that $\bar{f} = p_0 f_0 + (1-p_0) f_1$ to obtain

$$r_0 = \frac{1}{p_0 + \frac{1-p_0}{r}} \quad (5)$$

This equation can be substituted into Equation 4 to yield an explicit prediction of the expected level of allele convergence, shown in Figure 5 for various values of r, assuming — without loss of generality — that the zero allele conveys a fitness disadvantage, i.e. $r < 1$. Also shown (in dashed lines) are the approximate solutions obtained by using Equation 4 directly with a fixed r_0; the figure demonstrates that at low values of $\tilde{p}_0(m)$ this does provide a good estimate since then $\bar{f} \approx f_1$ and hence $r_0 \approx r$.

For a Gray-coded gene the three target intervals of the DPE zoom operator can be expressed as first-order schemata $0\#\cdots$, $1\#\cdots$ and $\#1\cdots$. The above analysis therefore applies to the DPE algorithm and can be used to specify the trigger threshold for the zoom operator: if DPE is to prune those parts of the search space whose absolute schema fitness is less than r, a zoom to a target subinterval should be triggered whenever the proportion of the population outside that subinterval falls below the corresponding value of $\tilde{p}_0(m)$ at the given mutation rate.

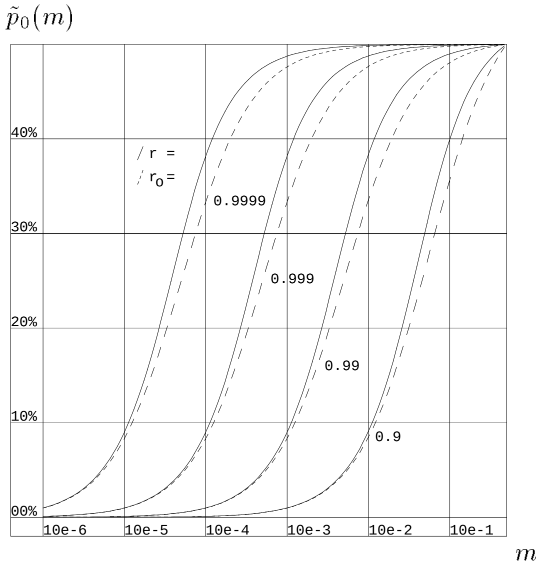

Figure 5: equilibrium proportion $\tilde{p}_0(m)$ of losing allele for various r (solid) and r_0 (dashed).

Note that in standard binary coding (as opposed to Gray coding) the center zoom target interval is the union of two second-order schemata, $01\cdots$ and $10\cdots$. Since a crossover after the first bit between two representatives of these schemata moves both outside the target interval, high levels of convergence will be much harder to achieve for the center interval than for the other two targets. This disruptive effect of crossover complicates the above analysis considerably and would necessitate the introduction of a separate DPE trigger threshold (dependent on both crossover and mutation rate) for the center target interval. To avoid such undesirable complications the use of Gray code for DPE-mapped genes is highly recommended; it is also generally preferable as it reduces *Hamming cliffs* that could mislead the GA search [1].

5.2 Genetic Hitchhiking

In the previous section we showed that the DPE trigger threshold must not be set too high since this would make it impossible for DPE to eliminate slightly suboptimal areas of the search space from consideration. On the other hand it is clear that the threshold should not be set too low either in order to be sufficiently immune to noise. Disregarding the non-cumulative effects of *stochastic variance* and *quantization noise* introduced by the various genetic operators, we now demonstrate how the main contributor to the GA's noise problem, *sampling error*, can distort convergence statistics.

Sampling error is the systematic and cumulative error in convergence statistics resulting from the fact that the GA can only explore a very small subspace of the entire search space during its search. This means that the GA has only empirical — and potentially misleading — estimates of the true schema fitness averages to work with. Crossover reduces sampling error by continually recombining genome fragments in novel ways, allowing more accurate estimates of the true schema fitness averages despite of a limited population size.

Sampling error can cause premature convergence — with DPE, in form of a misguided zoom operation — and is particularly prominent in a phenomenon known as *genetic hitchhiking*, referring to the fact that a "hitchhiker" gene that does not convey any fitness advantage may still become predominant by exploiting an incidental correlation with a highly successful "booster" gene.

To demonstrate this phenomenon, we model a GA operating on a 2-bit genome. The first bit is the *booster* gene that conveys a high reproductive advantage $R = f_{1\#}/f_{0\#}$ upon the individuals carrying it. The second bit is the *hitchhiker* that is completely neutral with respect to selection, i.e. $f_{00} = f_{01}$ and $f_{10} = f_{11}$.

We assume that initially a certain proportion p_i of the population have the booster allele, and that all of these also carry the hitchhiker: $p_{11} = p_i$, $p_{10} = 0$. Of the remainder of the population, half have the hitchhiker allele: $p_{00} = p_{01} = (1 - p_{11})/2$. From these initial conditions we follow the evolution of allele frequencies through time by repeatedly applying selection, mutation and crossover to the population.

Selection is assumed to be proportional to the population-relative schema fitness, i.e. the allele frequencies p' after selection are given by $p'_{xy} = r_{xy} p_{xy}$, where

$$r_{00} = r_{01} = \frac{1}{p_{0\#} + R\,p_{1\#}} \tag{6}$$

and $r_{10} = r_{11} = R\,r_{00}$.

Applying mutation at a per-bit rate of m next we find that

$$p''_{xy} = m^2 + (1-2m)(p'_{xy} + m(p'_{\bar{x}y} + p'_{x\bar{y}})) \tag{7}$$

Crossover between the two bit positions, applied with a probability of c, finally gives us

$$p'''_{xy} = p''_{xy} + c(p''_{\bar{x}y}p''_{x\bar{y}} - p''_{xy}p''_{\bar{x}\bar{y}}) \tag{8}$$

After examining the new allele frequencies p''' we can iterate the model by initializing the next generation ($p_{xy} \leftarrow p'''_{xy}$). The typical result is an evolution of the hitchhiker allele frequency as shown in Figure 6:

1. Due to its initial correlation with the booster allele, the hitchhiker frequency rises rapidly at first, with a slope dependent mostly on R, but also p_i.

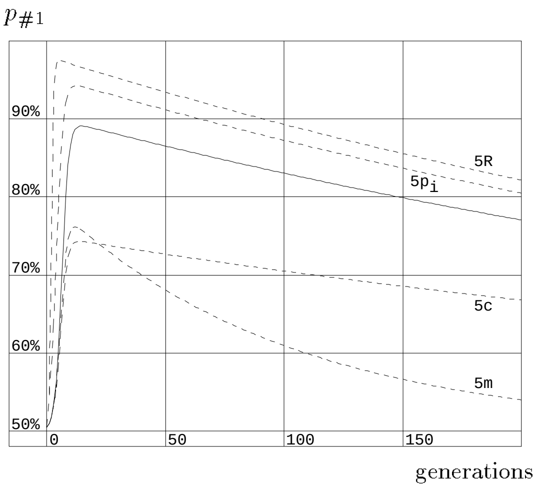

Figure 6: Evolution of modelled hitchhiker allele frequency for $R = 2$, $c = 0.02$, $m = 0.001$ and $p_i = 0.01$ (solid); variations (dashed) for fivefold increase of R, c, m and p_i, respectively.

2. At a point jointly determined by R, c, m and p_i crossover manages to decouple the hitchhiker allele from the booster, and its frequency peaks out.

3. The decay of the hitchhiker allele back to its equilibrium proportion of 50% is achieved by mutation alone and hence very slow at realistic mutation rates.[12]

The slow decay unfortunately means that this model can not be used to derive some lower bound on the DPE trace time constant, since in a typical multidimensional genome the abundant linkage forces will tend to overwhelm mutation, and the duration for which the hitchhiker stays above threshold will thus be dominated by the frequency of such hitchhiking events. We can, however, use the maximum proportion of the hitchhiker as a lower bound on the trigger threshold. Even so it is clear that this model is still far from reality; we consider it but a first step towards a fuller understanding of the premature convergence problem.

[12] There are variants of GA search employing very high mutation rates; they are not considered here since they either use non-convergent populations, or converge by a different mechanism.

6 Discussion

The DPE algorithm described here provides a useful extension of the GA search procedure to real-valued parameters of arbitrary precision. By tracking the convergence of a population and using it to restrict subsequent search, DPE efficiently finds precise parameter values using a minimal number of bit positions. The experiments reported here — and others we have performed — suggest that DPE can often help the GA find better solutions faster.

Compared to other algorithms adapting the GA search space, DPE is more conservative and hence amenable to standard GA analysis techniques. We have analyzed the role mutation plays as an inhibitor of convergence as well as ways in which sampling error can fool DPE into prematurely converging on suboptimal regions, but the latter analysis is far from complete. Further progress will require a better understanding of the conditions leading to premature convergence — an essential step towards a more general theory of GA search.

Acknowledgements

We thank the Cognitive Computer Science Research Group at UCSD for their commentary on early drafts of this paper. This work was partially supported by Los Alamos National Laboratories under INCOR contract CNLS/89-423.

References

[1] Richard A. Caruana and J. David Schaffer. Representation and hidden bias: Gray vs. binary coding for genetic algorithms. In *Proc. 5th Int. Conf. Machine Learning*, pages 153–161, San Mateo, CA, 1988. Morgan Kaufmann.

[2] Kenneth A. De Jong. *An Analysis of the Behavior of a Class of Genetic Adaptive Systems*. PhD thesis, Dept. of Computer and Comm. Sciences, Univ. of Michigan, Ann Arbor, MI, 1975. Univ. Microfilms No. 76-9381.

[3] Kalyonmoy Deb. Binary and floating point optimization using messy genetic algorithms. Technical Report 91004 (dissertation), The Clearinghouse for Genetic Algorithms, Univ. of Alabama, Tuscaloosa, AL, 1991.

[4] David E. Goldberg. *Genetic Algorithms in Search, Optimization & Machine Learning*. Addison-Wesley, Reading, MA, 1989.

[5] John J. Grefenstette. A user's guide to GENESIS. Technical Report CS-84-11, Vanderbilt Univ., Nashville, TN, 1984.

[6] William E. Hart and Richard K. Belew. Optimizing an arbitrary function is hard for the genetic algorithm. In Richard K. Belew and Lashon B. Booker, editors, *Proc. 4th Intl. Conf. Genetic Algorithms*, pages 190–195, San Mateo, CA, 1991. Morgan Kaufmann.

[7] John H. Holland. *Adaptation in Natural and Artificial Systems*. The Univ. of Michigan Press, Ann Arbor, MI, 1975.

[8] Venkat R. Mandava, J. Michael Fitzpatrick, and David R. Pickens, III. Adaptive search space scaling in digital image registration. *IEEE Trans. Medical Imaging*, 8(3):251–262, September 1989.

[9] J. David Schaffer, Richard A. Caruana, Larry J. Eshelman, and Rajarshi Das. A study of control parameters affecting online performance of genetic algorithms for function optimization. In J. David Schaffer, editor, *Proc. 3rd Intl. Conf. Genetic Algorithms*, pages 51–60, San Mateo, CA, 1989. Morgan Kaufmann.

[10] H. P. Schwefel. *Numerical Optimization of Computer Models*. Wiley, Chichester, 1981.

[11] Craig G. Shaefer. The ARGOT strategy: Adaptive representation genetic optimizer technique. In John J. Grefenstette, editor, *Genetic Algorithms and their Applications: Proc. 2nd Intl. Conf.*, pages 50–58, Hillsdale, NJ, 1987. Lawrence Erlbaum Associates.

[12] D. Whitley, K. Mathias, and P. Fitzhorn. Delta coding: An iterative search strategy for genetic algorithms. In Richard K. Belew and Lashon B. Booker, editors, *Proc. 4th Intl. Conf. Genetic Algorithms*, pages 77–84, San Mateo, CA, 1991. Morgan Kaufmann.