A Performance Assessment of Modern Niching Methods for Parameter Optimization Problems

Jean-Paul Watson
Computer Science Department
Colorado State University
Fort Collins, CO 80523 USA
e-mail: watsonj@cs.colostate.edu

Abstract

Niching genetic algorithms (NGAs) are designed to locate multiple fitness function optima. Numerous NGAs exist, but an accurate picture of their relative strengths and weaknesses remains elusive. The variety of performance measures and experimental methodologies makes accurate comparison difficult. Test functions are also limited in number, and possess structural regularities. Furthermore, most NGAs require determination of one or more parameters, but the issue of parameter sensitivity is rarely explored. Here, we study the performance and sensitivity of several NGAs using a common experimental methodology. We consider several new non-uniform test functions, in addition to those commonly used. Finally, NGA researchers have almost exclusively used binary variable encodings. We also analyze NGA performance under both binary and gray encodings.

1 Introduction

No widely accepted methodology for performing comparative analyses of niching genetic algorithms (NGAs) currently exists. Researchers use a wide variety of both experimental methodologies and performance measures, preventing thorough comparisons. Commonly used test functions are limited in number and complexity. Thus, NGAs run the risk of becoming over-specialized to these test functions and there is no guarantee that performance will transfer to more difficult, unstructured problems. Most NGAs also require the specification of parameters, and effective, operational heuristics for parameter value determination are often not available. Without strong guidelines, the empirical sensitivity of NGAs to parameter settings is an important predictor of NGA performance in real-world settings.

This study defines an experimental methodology for NGA evaluation and analyzes the performance and sensitivity of several modern NGAs using this methodology. The methodology focuses on 'operational' performance measures, which do not rely on a-priori knowledge of the fitness function. The experiments consider NGA performance on both existing and newly designed multimodal test functions. Existing test functions contain known regularities, and it is often unclear how NGAs will behave on highly non-uniform problems. Despite the advantages of gray encoding for unimodal function optimization (Whitley et al., 1998), binary encodings remain the nearly exclusive choice of NGA researchers. We also investigate whether gray encodings confer any advantages to NGAs. Finally, we limit the scope of our study to the performance of NGAs on *parameter optimization* problems in which the phenotypes are real-valued.

2 The Niching Genetic Algorithms

The majority of modern NGAs are based either on DeJong's (1975) crowding scheme or Goldberg and Richardson's (1987) sharing scheme. In crowding NGAs, new individuals enter the population by replacing similar individuals; selection and reproduction proceed normally. Sharing NGAs are identical to traditional genetic algorithms, with the exception that the fitness of each individual is reduced before selection occurs; the degree of reduction is a function of the number of similar individuals in the population. Due to the goals of this study and limited space, we describe below each NGA in terms of it's 'critical' niching parameters. Finally, all described distance measures operate in the phenotypic space because we are dealing with parameter optimization problems.

We consider two crowding-based NGAs; Mahfoud's (1992) Deterministic Crowding (DC) and Harik's (1995) Restricted Tournament Selection (RTS). In terms of niching ability, DeJong's crowding mechanism was proved relatively ineffective in prior studies (Mahfoud, 1992). DC has no niching parameters; children compete with their parents for insertion into the population according to deterministic rules. RTS, a variant of a steady-state GA (Whitley, 1989) has one niching parameter: the window size w. Whenever a new individual is produced, w individuals are selected at random (with replacement) from the population. The new individual and the nearest of the w samples then compete in a binary tournament, and the winner enters the population. In our experiments we define $w = numpeaks_{est} \cdot peak_{mult}$ where $numpeaks_{est}$ represents an estimate of the number of fitness peaks and $peak_{mult}$ represents the sampling rate multiplier.

We also consider Goldberg and Richardson's original sharing scheme and three sharing-based derivatives. The standard sharing algorithm (SH) requires specification of a single niching parameter, σ_{share}. The fitness of each individual is derated by a function of the number of individuals within a radius equal to σ_{share}. Accurately estimating σ_{share} for functions with peaks varying in fitness, extent, and shape is extremely difficult; this is the major drawback to SH. Furthermore, the computation of the derated fitness is $O(n^2)$ for population sizes equal to n. Miller and Shaw (1996) developed dynamic niche sharing (DYS) to reduce the time complexity. DYS requires an estimate of the number of peaks, $numpeaks_{est}$, in addition to σ_{share}. Using $numpeaks_{est}$, DYS is able to improve on the time complexity of SH. We did not explore variants of DYS incorporating mating restriction schemes.

Two sharing-based approaches, the adaptive clustering algorithm (YG) of Yin and Germay (1993) and the coevolutionary sharing algorithm (CS) of Goldberg and Wang (1998), attempt to directly avoid estimation of σ_{share}. YG is based on an adaptive form of k-means clustering. Clusters of individuals correspond to niches and individual fitness is derated by a function of member cluster size. The clustering mechanism requires specification of the 'refining' parameter d_{min} and the 'coarsening' parameter d_{max}. Clusters must be separated by a distance d_{min} and new clusters are formed from individuals more than d_{max} from existing clusters. In our experiments, we define $d_{max} = dmult \cdot d_{min}$, where $dmult$ is some integer constant. Finally, YG requires specification of the initial number of clusters $init_{clusters}$.

Inspired by a model of monopolistic competition, CS uses two coevolving populations: customers and businessmen. Customers are 'served' by the nearest businessman. Using a sharing-like function, customer fitness is derated in proportion to the total number of other customers served by the nearest businessman. Thus, there is pressure to find businessmen serving relatively few customers. The customer population evolves under a traditional GA. In contrast, the businessmen attempt to maximize the number of customers served; having more customers yields higher fitness. To prevent convergence of the businessman population to a single global optimum, businessmen must be separated by a distance of at least d_{min}. The businessman population evolves via an imprint mechanism in which the best customers are 'converted' into businessmen. For each businessman, up to n_{limit} customers are selected at random. The first customer (if any) that is both more fit and at least d_{min} away from the other businessmen then replaces the original businessman in the population. Because it influences the number of peaks CS is able to maintain, we also treat the businessmen population size $popsize_{bus}$ as a critical niching parameter.

Finally, we also consider random sampling (RND) as a niching method. RND serves as a baseline which any NGA should be able to outperform. RND operates simply by randomly generating some fixed number x of points in the domain of the fitness function.

3 The Test Functions

We consider 8 one- and 8 two-dimensional test functions. Four test functions in each category are taken from the literature, with the remaining four constructed for the purposes of this study. The sinusoidal functions M1-M4 (Mahfoud, 1995) are the most commonly used 1-D NGA test functions. We found only two commonly used 2-D NGA test functions: Shekel's Foxholes and the modified Himmelblau's function (both are described in (Mahfoud, 1995)). In addition, we consider the 2-D functions F6 and F7 defined by Bohachevsky et. al. (1986) although they have not, to our knowledge, been previously used in NGA studies.

The motivation for constructing new multimodal test functions is two-fold. First, existing NGA test functions are limited in terms of both difficulty and irregularity. To find more difficult test functions NGA researchers typically turn to Goldberg's massively multimodal deceptive binary test functions, commonly referred to as M7 and M8 (Mahfoud, 1995). Shekel's Foxholes is the most complex parameter optimization problem used in NGA studies. Yet, peak separation

and fitness in all of these test functions is very regular. Real-world functions rarely exhibit these characteristics and NGA performance may not transfer to fitness functions with more complex and irregular structures. Second, by using a small, limited test suite we run the risk of developing over-specialized NGAs, with performance failing to transfer to other problems.

We used cubic and bi-cubic spline functions (Press et al., 1988) to hand-construct the additional one and two-dimensional test functions, respectively. We denote the 1-d functions by CSP1-4 and the 2-d functions by BSP1-4. CSP1 and BSP1 are shown in Figures 1 and 2, respectively[1]. CSP1 and BSP1 exhibit highly irregular surfaces with a large number of peaks scattered throughout the domain. In contrast, CSP2 and BSP2 contain high-fitness peaks with several low-proximity peaks in close proximity. Finally, peak fitness in CSP3 and BSP3 varies over a very small range while the landscapes of CSP4 and BSP4 are dominated by a single large, high-fitness peak.

4 The Performance Measures

Historically, NGAs are primarily judged by their ability to locate *all* fitness function peaks. To quantify this goal, one commonly used measure simply counts the number of peaks containing individuals in their basins of attraction which possess fitness' equal to or greater than 80% of the peak fitness (Miller and Shaw, 1996). After such filtering, Miller and Shaw (1996) define the *maximum peak ratio* statistic as

$$peak_{ratio} = \frac{\sum_{i=1}^{p} f_i}{\sum_{i=1}^{N} f_i}$$

where p is the number of identified peaks, N is the total number of peaks, and f_i is the fitness of the i_{th} peak. In contrast, Mahfoud (1995) uses a slightly more operational measure. Instead of identifying those individuals 'near' the peak, each individual hill-climbs to the nearest peak, and the total number of evaluations required is recorded.

These two measures address different but equally important aspects of NGA performance. The $peak_{ratio}$ measure captures the proportion of peaks identified by an NGA. In contrast, Mahfoud's measure captures the proximity of individuals to the fitness peaks. In real-world applications, the exact function structure is unknown - post-processing of the population is often required to uniquely identify the fitness peaks. In our experiments we use the $peak_{ratio}$ statistic, with the modification that we consider a peak as 'identified' if any individual resides in its' basin of attraction, *independent of fitness*. We also measure the average number of function evaluations required to hill-climb individuals to their corresponding peaks, denoted by $evals_{hcavg}$.

To compute $evals_{hcavg}$, we use a steepest-ascent phenotypic hill-climber, using a fixed step size determined by 1) the number of bits used to encode the variables and 2) the range of the variable. This hill-climber overcomes three deficiencies present in Mahfoud's (1995) hill-climber. First, we feel a performance measure should be parameterless; Mahfoud's hill-climber requires specification of a single, initial step size. Second, and because of this parameter, Mahfoud's hill-climber cannot guarantee that an individual will not 'jump basins'; all peaks may *not* be properly identified. Finally, the algorithm fails on multi-dimensional functions in which ridges are pervasive, as it does not attempt simultaneous changes in two or more variables.

We also use two additional, secondary NGA performance measures. The first, denoted by $evals_{tot}$, measures the number of function evaluations required for population convergence. Several methods to determine NGA convergence have been introduced, the simplest of which terminates an NGA after a fixed number of function evaluations have been performed. However, this approach makes it difficult to accurately assess the number of evaluations actually *required* by an NGA to identify fitness function peaks. Following Beasley et. al. (1993) and Mahfoud (1995) , we use a form of *halting window* to determine NGA convergence. If we denote the current generation by t_0 and the prior generations by $t_{-1}, t_{-2}, \cdots, t_{-(N-1)}$ (with N the size of the halting window), we can halt an NGA if the difference in average fitness at t_0 and $t_{-(N-1)}$ is not more than some value *inc*. This mechanism halts an NGA when the average *raw* (as opposed to derated) fitness stagnates. Mahfoud (1995) used $N = 5$ in his experiments. Our preliminary experiments showed that the NGAs examined often found substantial improvements beyond this limitation, and instead use $N = 20$. We take *inc* as $\frac{1}{1000}$ of the fitness function range. We also found that requiring a strict increase in raw fitness causes premature convergence in many situations and instead terminate the NGAs when the *absolute* difference in average fitness at t_0 and $t_{-(N-1)}$ is not more than *inc*. Finally, we found that convergence in reasonable time periods is only possible when using low-noise selection operators such as SUS.

Researchers have expended significant effort in the design of NGAs capable of both locating and maintaining

[1]The spline definition files and evaluation code (in C++) are available from the author.

fitness peaks. For example, Mahfoud (1992) showed that the number of peaks identified by DeJong's crowding mechanism substantially decreased with an increase in the number of total evaluations. Thus, our other secondary performance measure, $peak_{maint}$, represents the ability of an NGA to locate and *maintain* individuals at the fitness peaks for extended periods of time.

Most researchers simply track the number of peaks maintained by an NGA as a function of the number of evaluations performed(Deb and Goldberg, 1989); to our knowledge NGA peak maintenance has not been explored in conjunction with a convergence-based termination criterion. After NGA convergence, we measure the $peak_{ratio}$ and perform an additional 100 generations of processing. Then, we record the new $peak_{ratio}$ and denote the difference by $peak_{maint}$. Positive values of $peak_{maint}$ indicate the NGA was prematurely converged, while negative values suggest that too strong a termination criterion was used. Thus, $peak_{maint}$ provides an indirect measure of the sensitivity of the algorithm to N, the size of the halting window.

5 Experimental Methodology

Table 1 defines a set of values for each 'critical' niching parameter described in Section 2. For test functions taken from the literature, the values are moderate deviations from commonly used settings. More judgment was required in determining values for our spline-based test functions. For each NGA, we performed 30 independent trials for all possible combinations of the following independent variables: 1) encoding (either binary or gray), 2) population size (100, 300, or 500), and 3) critical parameter value(s). For each set of 30 trials, the dependent variables were the mean $peak_{ratio}$, $evals_{hcavg}$, $evals_{tot}$, and $peak_{maint}$. We also measured the performance of RND for $peak_{ratio}$ and $evals_{hcevals}$, with the number of samples equal to the NGA population size. Given a specific encoding and population size, we next summarized the results from all the corresponding parameter combinations by computing a mean and standard deviation for each performance measure. Selected results are recorded in Tables 2 through 5.

We encode all variables with 15 bits. For both non-critical niching and GA parameters, we take common settings from the literature. With the exception of CS, all NGAs were configured with two-point crossover; CS uses one-point crossover. We use SUS selection (Baker, 1987) for SH, DYSH, the CS customer population, and YG. RTS uses binary tournament selection; DC has a built-in selection mechanism. Crossover probabilities for RTS, SH, DYSH, CSN, and YG were 0.9, 1.0, 1.0, 1.0, and 0.9, respectively. We chose the triangular sharing function ($\alpha = 1.0$) for both SH and DYSH. We configured all algorithms with a standard mutation operator, p=0.0001 per bit. For RTS, we use a sampling rate multiplier $peak_{mult}$ of 4. For CS, we take n_{limit} as twice the businessman population size $popsize_{bus}$; for all test functions we consider $popsize_{bus}$ values of 15, 30, and 45. Finally, for YG we take the initial number of clusters ($clusters_{init}$) as twice the actual number of fitness peaks and use *dmult* values of 2 and 3.

func	RND	DC	RTS	SH	DYS	CS	YG
M1	1630	109	20	948	903	661	858
M2	1619	87	32	875	852	677	612
M3	1668	27	30	740	736	661	598
M4	1644	27	43	745	695	636	534
CSP1	802	22	28	716	700	607	665
CSP2	1094	382	58	825	819	872	627
CSP3	1723	142	68	1738	1828	1505	1718
CSP4	2338	928	95	2495	2868	2390	1837
HBL	5990	540	387	6012	5117	4755	5185
FOX	3891	50	288	605	517	433	459
F6	3667	202	116	2803	3018	1836	2359
F7	3267	208	196	2936	3114	1840	2604
BSP1	3382	77	118	2964	3111	1667	2599
BSP2	4834	496	150	4225	4331	2010	3974
BSP3	3322	175	142	3351	3436	2023	3333
BSP4	3682	269	145	3243	3335	1516	2623

Table 3: The mean $evals_{hcavg}$ taken over all combinations of niching parameters. Binary encoding was used and the population size was 300.

6 Mean Algorithm Performance

In this section, we compare the performance of the various NGAs, and discuss the support for two specific experimental hypotheses. First, we expected the NGAs to generally perform worse on the test functions constructed specifically for this study, for reasons discussed in Section 3. In preliminary experiments with sharing-based NGAs, we found that under-estimation of the 'separation' parameters such as σ_{share} or d_{min} result in good $peak_{ratio}$ performance but very poor $evals_{hcavg}$ performance; large proportions of individuals are maintained far from fitness peaks. Thus, we expected the sharing-based NGAs to outperform both DC and RTS in terms of $peak_{ratio}$ but not in terms of $evals_{hcavg}$.

First we consider the generally poor performance of DC. Mahfoud's (1995) study demonstrated the poor performance of DC on Shekel's Foxholes; more difficult parameter optimization problems were not considered. The data in Table 2 suggests this result was not anomalous; DC underperforms all of the other NGAs

	RTS	SH	DYS		CS	YG
func	$numpeaks_{est}$	$sigma_{share}$	$numpeaks_{est}$	$sigma_{share}$	d_{min}	d_{min}
M1	3-11, 2	0.02-0.2, 0.02	3, 5, 7	0.02-0.2, 0.02	0.005-0.05, 0.005	0.02-0.2, 0.02
M2	3-11, 2	0.02-0.2, 0.02	3, 5, 7	0.02-0.2, 0.02	0.005-0.05, 0.005	0.02-0.2, 0.02
M3	3-11, 2	0.02-0.2, 0.02	3, 5, 7	0.02-0.2, 0.02	0.005-0.05, 0.005	0.02-0.2, 0.02
M4	3-11, 2	0.02-0.2, 0.02	3, 5, 7	0.02-0.2, 0.02	0.005-0.05, 0.005	0.02-0.2, 0.02
CSP1	11-21, 2	0.2-2.0, 0.2	11, 15, 19	0.2-2.0, 0.2	0.05-0.5, 0.05	0.2-2.0, 0.2
CSP2	8-16, 2	0.2-2.0, 0.2	9, 12, 15	0.2-2.0, 0.2	0.05-0.5, 0.05	0.2-2.0, 0.2
CSP3	4-14, 2	0.2-2.0, 0.2	5, 7, 9	0.2-2.0, 0.2	0.05-0.5, 0.05	0.2-2.0, 0.2
CSP4	6-16, 2	0.2-2.0, 0.2	6, 9, 12	0.2-2.0, 0.2	0.05-0.5, 0.05	0.2-2.0, 0.2
HBL	3-11, 2	2.75-5.25, 0.25	2, 4 ,6	2.25-5.25, 0.25	0.25-2.5, 0.25	3.0-5.5, 0.25
FOX	20-30, 2	3-13, 1	21, 25, 29	3-13, 1	1-10, 1	1.0-10.0, 0.2
F6	11-21, 2	0.06-0.46, 0.04	11, 15, 19	0.06-0.46, 0.04	0.01-0.1, 0.01	0.25-0.75, 0.05
F7	17-27, 2	0.1-1.0, 0.1	17, 23, 29	0.1-1.0, 0.1	0.01-0.1, 0.01	0.25-0.75, 0.05
BSP1	16-26, 2	0.2-2.0, 0.2	17, 21, 25	0.2-2.0, 0.2	0.05-0.5, 0.05	0.2-2.0, 0.2
BSP2	8-16, 2	0.2-2.0, 0.2	8, 11, 14	0.2-2.0, 0.2	0.05-0.5, 0.05	0.2-2.0, 0.2
BSP3	4-14, 2	0.2-2.0, 0.2	17, 21, 25	0.2-2.0, 0.2	0.05-0.5, 0.05	0.2-2.0, 0.2
BSP4	6-16, 2	0.2-2.0, 0.2	15, 18, 21	0.2-2.0, 0.2	0.05-0.5, 0.05	0.2-2.0, 0.2

Table 1: Critical niching parameter values, per test function

func	RND	DC	RTS	SH	DYS	CS	YG
M1	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	0.90±0.15	1.0±0.0	1.0±0.0
M2	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	0.89±0.14	0.97±0.07	0.81±0.21
M3	1.0±0.0	1.0±0.0	1.0±0.0	1.0±0.0	0.87±0.16	1.0±0.0	0.98±0.05
M4	1.0±0.0	1.0±0.0	0.99±0.02	1.0±0.0	0.87±0.17	1.0±0.0	0.91±0.19
CSP1	0.99±0.02	0.67±0.04	0.86±0.07	0.98±0.0	0.89±0.12	0.96±0.03	0.96±0.03
CSP2	0.94±0.01	0.51±0.05	0.75±0.01	0.88±0.05	0.82±0.11	0.88±0.10	0.64±0.18
CSP3	1.0±0.0	0.80±0.09	0.85±0.07	1.0±0.0	0.92±0.09	0.90±0.19	1.0±0.0
CSP4	1.0±0.0	0.79±0.02	0.84±0.02	0.96±0.06	0.88±0.08	0.75±0.09	0.88±0.08
HBL	1.0±0.0	1.0±0.0	1.0±0.0	1.0 ±0.0	0.96±0.06	1.0±0.0	1.0±0.0
FOX	1.0±0.0	0.28±0.03	0.93±0.04	1.0±0.0	1.0±0.0	0.90±0.15	0.99±0.02
F6	1.0±0.0	0.10±0.0	0.94±0.03	0.97±0.03	0.96±0.08	0.63±0.10	0.74±0.17
F7	0.99±0.02	0.05±0.0	0.83±0.01	0.98±0.03	0.98±0.03	0.57±0.09	0.87±0.08
BSP1	0.97±0.02	0.56±0.04	0.81±0.01	0.96±0.03	0.96±0.03	0.65±0.16	0.90±0.09
BSP2	1.0±0.0	0.66±0.07	0.66±0.02	0.95±0.12	0.95±0.12	0.47±0.20	0.98±0.03
BSP3	0.94±0.02	0.39±0.04	0.80±0.04	0.95±0.01	0.95±0.01	0.59±0.24	0.95±0.01
BSP4	0.98±0.02	0.52±0.09	0.83±0.01	0.91±0.16	0.92±0.15	0.33±0.08	0.84±0.17

Table 2: The mean and standard deviation of $peak_{ratio}$ taken over all combinations of niching parameters. Binary encoding was used and the population size was 300.

in terms of $peak_{ratio}$. The relatively poor performance of DC underscores the need to test with a wide range of non-uniform test functions, and to consider both difficult binary *and* parameter optimization problems in empirical evaluations. Given that DC is parameterless, it likely cannot overcome these deficiencies.

Next, we consider the relative performance of the following NGA classes: sharing-based, crowding-based, and random sampling. In terms of $peak_{ratio}$, Table 2 shows RND as the best NGA. With a population size of 300, test functions with fewer than 30 peaks, and a limited domain, each peak will almost always contain an individual within its basin of attraction, so this result is unsurprising. Of the remaining two classes, sharing-based NGAs tend to outperform crowding-based NGAs for the $peak_{ratio}$ metric, confirming our hypothesis.

Table 3 clearly shows crowding-based NGAs are superior in terms of $evals_{hcavg}$ performance, often by a substantial margin. A more surprising result was that the $evals_{hcavg}$ performance of sharing-based NGAs is often comparable to the performance of RND. Clearly, there appears to be a inverse relationship between $peak_{ratio}$ and $evals_{hcavg}$ performance for crowding and sharing-based NGAs.

Miller and Shaw (1996) conjectured DYS would perform worse than SH as deviations from the 'optimal' parameter settings increased. However, Table 2 suggests the performance of DYS and SH are nearly iden-

tical, supporting general use of the faster DYS algorithm in more general situations. YG has generally good $peak_{ratio}$ performance, but is very sensitive to the choice of test function. Finally, CS substantially under-performed the other sharing-based NGAs. Finally, because of the inconsistent or poor performance of many NGAs, we found inconclusive support from Tables 2 and 3 for our hypothesis that NGAs should perform worse on our newly developed test functions. However, RTS, SH, and DYS generally perform worse in terms of $peak_{ratio}$ on the newly developed test functions.

We next consider NGA performance in terms of the two secondary measures $evals_{tot}$ and $peak_{maint}$. As shown in Table 4, sharing-based NGAs require fewer total evaluations for convergence than crowding-based NGAs. More interestingly, the $evals_{tot}$ for sharing-based NGAs appears insensitive to the choice of test function. In contrast, crowding-based NGAs generally require substantially more total evaluations than their sharing-based counterparts and show more sensitivity individual test functions.

Finally, all NGAs maintained the identified peaks 100 generations after convergence. Researchers have expended significant effort in designing algorithms capable of not only locating but maintaining individuals at fitness peaks. No negative values of $peak_{maint}$ were observed, most were near 0.02, and the largest was 0.12. Thus, even with a window size of $N = 20$, these NGAs can still locate additional peaks after apparent convergence of the population. While researchers appear to have solved the problem of peak maintenance, work still remains in the area of identifying population convergence in NGAs. We did not consider the impact of the additional 100 generations on performance measures other than $peak_{ratio}$.

7 Algorithm Sensitivity

Previously, we argued that NGA performance should be measured in terms of *mean* performance over a 'reasonable' range of parameter values. By defining performance in operational terms, this paradigm also allows us to assess the sensitivity of an NGA to the choice of parameter value(s). Sensitivity is a crucial aspect of NGA performance because it provides a measure of algorithm reliability. If NGAs are very sensitive to parameter values, achieving good performance on real-world problems may be problematic. We equate sensitivity with performance standard deviations. Table 2 records the $peak_{ratio}$ standard deviations. Below, we compare NGA sensitivities and briefly discuss approaches to parameter value selection which can mitigate the the impact of high sensitivities.

func	DC	RTS	SH	DYS	CS	YG
M1	719	633	482	546	457	473
M2	1050	998	469	524	451	524
M3	580	675	491	538	446	510
M4	601	824	481	535	449	518
CSP1	891	733	423	426	419	425
CSP2	618	828	470	507	438	522
CSP3	854	924	417	449	439	452
CSP4	561	1045	448	517	718	498
HBL	493	549	399	461	378	430
FOX	558	829	505	509	474	518
F6	653	899	448	462	428	497
F7	797	963	409	417	410	454
BSP1	989	952	428	421	468	471
BSP2	1134	2019	427	427	456	460
BSP3	1078	772	401	401	480	412
BSP4	958	841	434	443	458	509

Table 4: The mean $evals_{tot}$ taken over all combinations of niching parameters. Binary encoding was used and the population size was 300. Reported values are the number of evaluations * 100

RND is clearly the least sensitive 'NGA'. In general, the sensitivity of the two crowding-based NGAs and SH is relatively low. For RTS, conservative estimates of the number of peaks results in lower sensitivity, although run-time in increased. DC is parameterless, so further limiting the sensitivity is impossible. Finally, unless the σ_{share} parameter is severely over-estimated SH sensitivity remains low.

Both DYS, CS, and YG are often highly sensitive to selection of parameter values, with standard deviations occasionally exceeding 0.2. Conceptually, we can partially attribute this increase to the introduction of an additional niching parameter. For CS, larger values of $popsize_{bus}$ improve $peak_{ratio}$ but also slightly increase $evals_{hcavg}$; the majority of CS sensitivity stems from the d_{min} parameter. YG only exhibits high levels of sensitivity on certain test functions. We currently have no explanation for this behavior, and are investigating various hypotheses.

8 Gray versus Binary Coding

Binary variable encoding remains the exclusive choice of NGA researchers. To our knowledge, no research has studied the impact of encoding on NGA behavior. In contrast to traditional GAs, an NGA operates both in the genotypic and real-valued phenotypic spaces. Crossover and mutation manipulate the traditional bit-string encodings. However, selection and replacement operate in the phenotypic variable space.

Such operations in the phenotypic space have an unknown impact on schema processing and other mechanisms which are generally assumed to drive search in a traditional GA. If the underlying representation has little impact on search quality, we can begin to hypothesize that in reality NGAs operate primarily in the phenotypic space with crossover and mutation serving simply to generate new points in the search space.

func	DC	RTS	SH	DYS	CS	YG
M1	0.0	0.0	0.0	0.0	0.0	0.11
M2	0.0	0.0	0.0	0.01	0.01	0.06
M3	0.0	0.0	0.0	0.03	0.0	0.02
M4	0.0	0.0	0.0	0.04	0.0	0.03
CSP1	0.12	0.0	0.0	0.04	0.0	0.03
CSP2	0.03	0.01	0.01	0.0	0.06	0.06
CSP3	0.13	0.02	0.12	0.01	0.0	0.06
CSP4	0.03	0.08	0.01	0.0	0.01	0.02
HBL	0.0	0.0	0.0	0.01	0.0	0.0
FOX	0.0	0.06	0.0	0.01	0.1	0.01
F6	0.0	0.01	0.02	0.04	0.18	0.05
F7	0.0	0.03	0.0	0.0	0.22	0.13
BSP1	0.02	0.0	0.01	0.01	0.10	0.02
BSP2	0.03	0.0	0.01	0.0	0.01	0.02
BSP3	0.09	0.01	0.0	0.0	0.01	0.0
BSP4	0.05	0.05	0.0	0.01	0.02	0.08

Table 5: The absolute difference in mean $peak_{ratio}$ under binary and gray encodings. population size was 300, and the mean is taken over all combinations of input parameters.

Table 5 shows the absolute difference in mean $peak_{ratio}$ for each NGA under both binary and gray variable encodings, on all test functions. All results were obtained using a population size of 300. With few exceptions the performance under the two encodings is very nearly equivalent and rarely exceeds 0.10. Thus, the interaction effect between encoding and NGA performance, in terms of the $peak_{ratio}$ measure, is at best subtle. Considering that NGA selection and replacement mechanisms operate in the phenotypic space (for parameter optimization problems), we feel the apparent lack of interaction effect *suggests* the possibility that NGA search operators significantly interfere with the underlying search mechanisms of a traditional GA. One interesting extension to this study involves eliminating the bit-encoding from these NGAs, using instead real-valued representations and operators from the evolutionary strategy (ES) community (Beck, 1996). Eliminating the bit-representations and leveraging prior work on ES analysis may result in a better understanding of how NGAs work on parameter optimization problems.

9 Conclusions

The use of a variety of performance measures and experimental methodologies makes accurate comparisons of NGA performance difficult. Furthermore, available test functions are limited in both number and topology; the fitness peaks are typically uniform in both separation and shape. Because NGA performance may not transfer to more difficult problems, we constructed a number of highly non-uniform test functions. We then measured the performance of a number of modern NGAs over a wide range of parameters and test functions.

In our experiments, we consider NGA performance in terms of both the number of peaks identified and the proximity of individuals to those peaks. For both crowding and sharing-based NGAs, we see a form of duality between these measures. Crowding-based NGAs fail to find as many peaks as sharing-based NGAs but individuals are closer to the fitness peaks. The issue of peak proximity is rarely addressed but if we consider both measures the sharing-based NGAs often perform worse than *random* search.

Most NGAs require the specification of one or more niching-related parameters and researchers almost exclusively report performance only for specific parameter values. We instead measure NGA performance over a range of parameter values, which enables us to assess sensitivity of the algorithms. The crowding-based NGAs and standard sharing algorithm exhibited low sensitivity. In general, the sensitivity of multiple-parameter NGAs is high, potentially limiting their applicability in real-world environments. Finally, we considered the interaction of coding and NGA performance. Performance varies little between binary and gray encoding, suggesting that NGAs may significantly interfere with GA search mechanisms.

Given the relatively strong performance of random search, we can also raise the question of whether NGAs are really useful for identifying multiple fitness peaks of parameter optimization problems. Work is currently in progress to determine under what conditions random search can outperform NGAs, in addition to extending the basic analysis to binary test functions.

References

Baker, J. E. (1987). Reducing bias and inefficiency in the selection algorithm. In *Proceedings of the Second International Conference on Genetic Algorithms.*

Beasley, D., Bull, D. R., and Martin, R. R. (1993). A

sequential niche technique for multimodal function optimization. *Evolutionary Computation*, 1(2):101–125.

Beck, T. (1996). *Evolutionary Algorithms in Theory and Practice*. Oxford University Press.

Bohachevsky, I. O., Johnson, M. E., and Stein, M. L. (1986). Generalized simulating annealing for function optimization. *Technometrics*, 28(3).

Deb, K. and Goldberg, D. E. (1989). An investigation of niche and species formation in genetic function optimization. In *Proceedings of the Third International Conference on Genetic Algorithms*.

DeJong, K. A. (1975). *An analysis of the behavior of a class of genetic adaptive systems*. PhD thesis, University of Michigan.

Goldberg, D. E. and Richardson, J. (1987). Genetic algorithms with sharing for multimodal function optimization. In *Proceedings of the Second International Conference on Genetic Algorithms*.

Goldberg, D. E. and Wang, L. (1998). *Genetic Algorithms and Evolutionary Strategies in Engineering and Computer Science*, chapter Adaptive Niching Via Coevolutionary Sharing. John Wiley and Sons.

Harik, G. R. (1995). Finding multimodal solutions using restricted tournament selection. In *Proceedings of the Sixth International Conference on Genetic Algorithms*.

Mahfoud, S. W. (1992). Crowding and preselection revisited. In *Parallel Problem Solving From Nature 2*.

Mahfoud, S. W. (1995). A comparision of parallel and sequential niching methods. In *Proceedings of the Sixth International Conference on Genetic Algorithms*.

Miller, B. L. and Shaw, M. J. (1996). Genetic algorithms with dynamic niche sharing for multimodal function optimization. In *IEEE International Conference on Evolutionary Computation*.

Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling, W. T. (1988). *Numerical Recipes in C*. University of Cambridge Press.

Whitley, D. (1989). The GENITOR algorithm and selective pressure: Why rank based allocation of reproductive trials is best. In *Proceedings of the Third International Conference on Genetic Algorithms*.

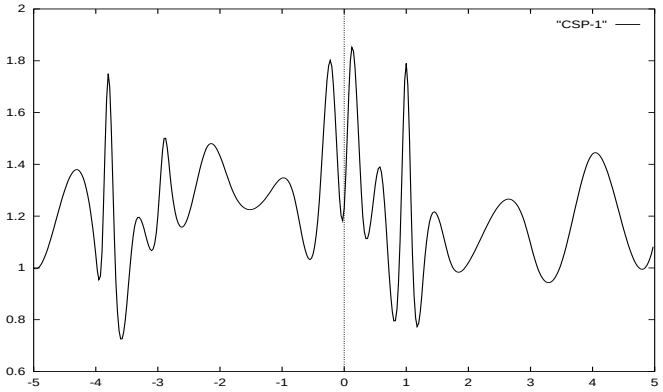

Figure 1: The One-Dimensional Cubic Spline Test Function CSP1

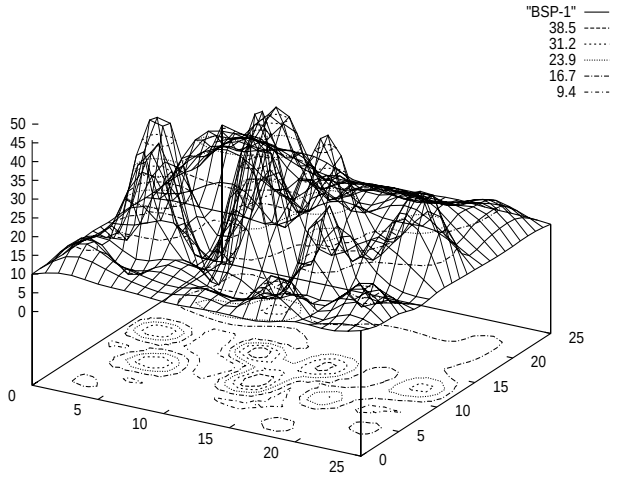

Figure 2: The Two-Dimensional Bi-Cubic Spline Test Function BSP1

Whitley, D., Rana, S., and Heckendorn, R. (1998). *Genetic Algorithms and Evolutionary Strategies in Engineering and Computer Science*, chapter Representation Issues in Neighborhood Search and Evolutionary Algorithms. John Wiley and Sons.

Yin, X. and Germay, N. (1993). A fast genetic algorithm with sharing scheme using cluster analysis methods in multimodal function optimization. In *Proceedings of the International Conference in Austria on Artificial Neural Nets and Genetic Algorithms*.