Improving the Locality Properties of Binary Representations

Adrian Grajdeanu and Kenneth De Jong

Department of Computer Science
George Mason University
Fairfax, VA 22030

Abstract. Choosing representations and operators that preserve locality between genotype and phenotype space is an important goal in EA design. In the GA literature there has been considerable discussion of this issue with respect to the choice between standard binary encoding and Gray codes. In this paper we argue that an important and unappreciated aspect of such discussions is the degree to which locality preservation is isotropic in phenotype space (i.e., independent of location in phenospace). We show that using a traditional bit-flip mutation operator with either of these two representations results in rather weak isotropic locality. These insights lead to the design of a new binary mutation operator that increases isotropic locality. The results from an initial set of experiments supports the hypothesis that this improvement in isotropic locality leads to improvements in GA performance as well.

1 Introduction

It is well-known in the EC community that the choice of representation and reproductive operators is critical to success of an application. One of the desirable features of such choices is that locality is preserved when mapping between the internal representation space (genospace) and the external application space (phenospace). The classical example of this in the GA literature involves the decision as to how best to map a problem into a binary representation. The most straightforward approach to representing ordered sets of phenotypic objects internally as binary strings is to assign the strings in order according to their binary value: 00...00, 00...01, 00..10, ..., 11...10, 11...11. So, for example, the interval [0.0,1.0] would be represented internally as binary strings whose length is dictated by the desired level of precision ϵ with the string 00...00 representing the real number 0.0 and the string 11...11 representing the real number 1.0. The most common mutation operator for binary representations is bit-flip mutation in which individual bits are stochastically selected to be flipped (switched from zero to one or vice versa). As a consequence, Hamming distance is the most natural distance metric in genospace while Euclidean distance is the most natural one for real-valued phenospaces.

The notion, then, of locality preservation is that small steps in one space correspond to small steps in the other space. Clearly, this is true for real numbers such as 0.0 and 0.0+ϵ and their corresponding strings 00...00 and 00...01.

K. Deb et al. (Eds.): GECCO 2004, LNCS 3102, pp. 1186–1196, 2004.

However, it is clearly not true for the real numbers whose binary representations are 01...11 and 10...00. In this case, achieving a small step in phenospace requires a large step in genospace. In the GA literature the traditional name for this phenomenon is a "Hamming cliff".

A standard way of resolving this issue is to switch to a Gray code representation in which adjacent points in phenospace are assigned bit strings that differ in only a single bit position (see, for example, [1], [2] or [3]). Hence, every small step in phenospace corresponds to a small step in genospace, but the reverse is clearly not true since there are single bit flips that result in large steps in phenospace.

The effect this can have on EA performance is easily seen by using a family of artificially constructed Hamming cliff landscapes. This parameterized family of landscapes, $hc[a, b, \alpha]$, is defined as follows. Given a real-valued interval $[a, b]$ and a parameter α whose value lies somewhere in $[a, b]$, then let:

$$hc[a, b, \alpha](x) = \begin{cases} x - (\alpha + 1) + b - a, x <= \alpha \\ x - (\alpha + 1), x > \alpha \end{cases}$$

By varying α one can position the global maximum anywhere in $[a, b]$. In particular, setting $\alpha = (b-a)/2$ places it immediately to the right of a large Hamming cliff. Figure 1 illustrates this by plotting $hc[0, 31, 16](x)$. Higher dimension versions are easily constructed by summing n copies of a 1-dimensional version:

$$HC[a, b, \alpha, n](x) = \sum_{i=1}^{n} hc[a, b, \alpha](x)$$

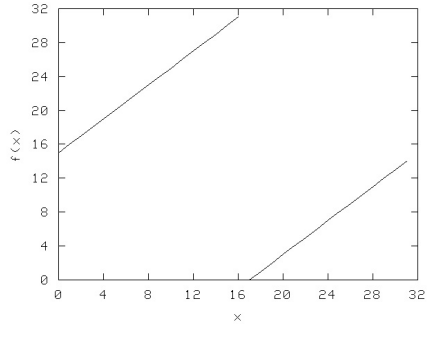

Fig. 1. Example of a 1-dimensional Hamming cliff: $hc[0, 31, 16](x)$.

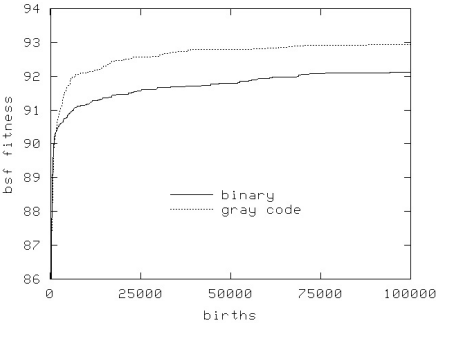

Fig. 2. Best-so-far performance of a standard GA with binary and Gray coding on $HC[0, 31, 16, 3](x)$.

Figure 2 illustrates how the performance of a standard GA, in terms of average best-so-far (bsf) curves, can be improved by switching from a standard

binary representation to a Gray code representation on a 3-dimensional version: $HC[0, 31, 16, 3](x)$. By contrast, when $\alpha = b$, no Hamming cliffs serve as barriers to the global optimum and both representations work equally well (Figures 3 and 4).

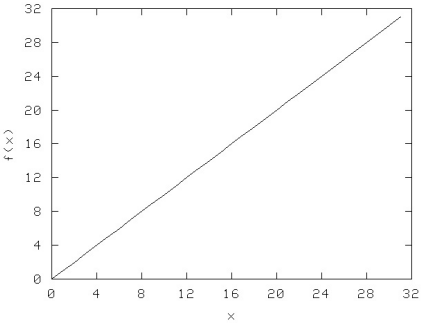

Fig. 3. The $hc[0, 31, 31](x)$ landscape.

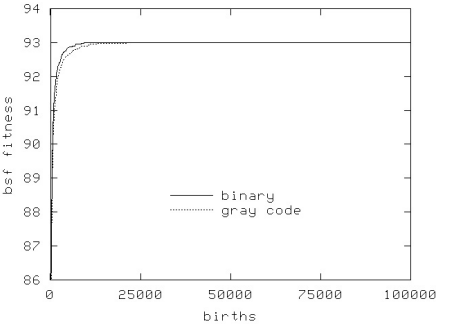

Fig. 4. Best-so-far performance of a standard GA with binary and Gray coding on $HC[0, 31, 31, 3]$.

Experiments such as these suggest that switching to Gray coding can improve performance when Hamming cliff barriers are encountered, and Gray coding doesn't seem to hurt performance much when no such barriers exist. Unfortunately, one doesn't need to look far to find counter examples. The Schwefel function [4] is a standard EC benchmark, and we shall consider it on a 2-dimensional $[0, 5000] \times [0, 5000]$ landscape, designated here as $Schwefel[0, 5000, 2]$. Figure 5 shows the negative effect that using a Gray code has on the average best-so-far performance of the same standard GA on this landscape.

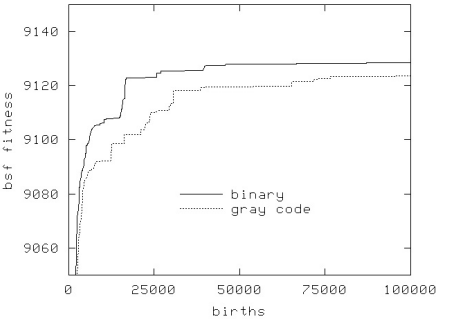

Fig. 5. Best-so-far performance of a standard GA with binary and Gray coding on the Schwefel[0.5000,2] landscape.

A survey of the EC literature produces a similar collection of mixed results For example, [5] presents a statistical comparison of binary and Gray encodings that suggests that Gray encodings are generally superior while an analysis in [6] concludes just the opposite. Attempting to bring the theory more in line with empirical experience, Whitley argues ([7]) for the benefit of Gray codes by showing that Gray codes induce fewer local optima than the standard binary encoding on a certain class of problems, arguably those of interest to the application oriented researchers. By contrast, using a Markov model to study relative performance of binary and Gray coding in genetic algorithms, it was shown in [8] that Gray coding does not necessarily improve performance for functions which have fewer local optima in Gray representation than in binary. In a subsequent paper Whitley shows that there is a complementary bias in the Gray and binary neighborhood structures and explores the possibility of using a combination of the two [9].

The results presented in this paper approach this representation/operator choice issue from a somewhat different perspective. Our sense is that a critical feature of such choices is the extent to which they result in locality preservation *uniformly* throughout phenospace, i.e., the notion of isotropic locality. We show that both binary encodings when used in conjunction with standard bit-flip mutation exhibit weak isotropic locality. This leads to the design of a generalized bit-flip mutation operator which, when used with a standard binary encoding, has better isotropic locality than either the binary or Gray code with the standard bit-flip mutation operator. In addition, we present the results of a preliminary set of experiments that shows a corresponding improvement in GA performance as well.

2 Locality Properties of Encodings

Ideally, the choice of internal representation and reproductive operators would result in a distance-preserving mapping between genospace and phenospace. This is difficult to achieve in general. A less ambitious goal is to choose a mapping that is locality preserving (i.e., small steps in genospace produce small steps in phenospace and vice versa). We noted above that the primary motivation for choosing a Gray code representation over the standard binary encoding was to improve on this locality property. In this section we explore in more detail how this is achieved.

2.1 Phenotypic Effects of 1-Bit-Flip Mutation

The simplest way to obtain insight into locality issues is to focus on the effect that a one-bit-flip mutation in genospace has in phenospace. Suppose, for example, that the phenospace consists of the interval [0.0,1.0] and the desired precision results in a 9 bit encoding. Then, for each of the 2^9 phenotype values, one can plot its change in value as a result of each one of the 9 possible 1-bit-flip mutation.

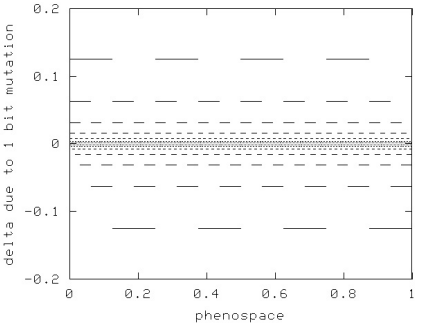

Fig. 6. A scatter plot for 1-bit-flip mutation with a standard binary encoding.

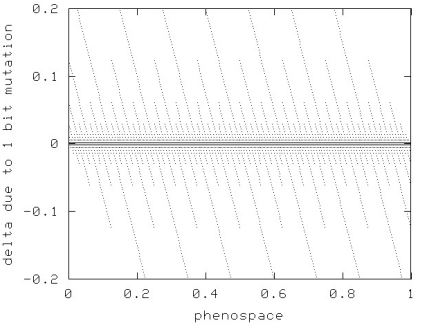

Fig. 7. A scatter plot for 1-bit-flip mutation using Gray code.

This analysis is presented in Figure 6 for the standard binary encoding and in Figure 7 for the Gray code representation.

In both cases single bit flips can produce large changes in phenotype values. However, since we are focusing on locality issues, we have zoomed in on the small phenotypic changes. Hence the Y range depicted is the $(-0.2, 0.2)$ sub-range and not the entire range of $(-1, 1)$. The degree of isotropic locality is represented by the continuity of the horizontal lines. The lack of continuity indicates a dependency on location (i.e., non-isotropy), while a continuous horizontal line depicts the ability of produce a particular change in phenotype value *regardless* of location in phenospace. Figures 6 and 7 show clearly that the Gray code produces much more consistent locality (i.e., better isotropy) for *small* changes in phenotypic value, but at the expense of decreased isotropy as the size of the change in phenotype values increases.

2.2 Quantifying Isotropic Locality

The scatter plots in the previous section are visually suggestive that a key difference between the two encodings is the degree to which the ability to take small steps in phenospace varies as a function of where one is in phenospace. The reverse of this, the notion of isotropic locality is - more formally - the ability to take certain steps independent of location in phenospace. We can quantify the degree to which an encoding is isotropic by calculating the probability of taking a step of size δ using 1-bit-flip mutation for each point in phenospace, and then looking at the mean and variance of these probabilities. This approach is illustrated in Figures 8 and 9 using the same setup as in the previous section.

For clarity Figures 8 and 9 zoom in on the first 31 possible (positive) changes in phenotypic value. They use box plots to display the mean, standard deviation, and maximum and minimum values of the probability distributions for each of the included positive changes δ. Hence, the degree of isotropic locality is represented by the variances of the probability distributions (high variance means low isotropy).

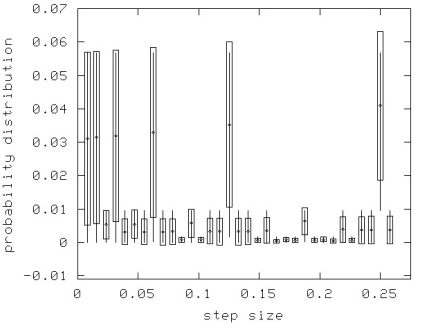

Fig. 8. Probability distribution for small step sizes δ using 1-bit-flip under the standard binary encoding.

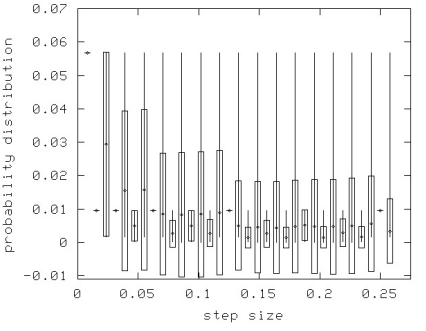

Fig. 9. Probability distribution for small step sizes δ using 1-bit-flip under the Gray code representation.

By comparing these two figures one can see clearly the isotropic differences between the two encodings. For the standard binary encoding (Figure 8) notice the high variances associated with the first, second, fourth, etc. values for δ which correspond precisely to the existence of Hamming cliffs. By contrast the Gray code probability distributions for the same step sizes have zero variance (Figure 9).

Further comparison of these two figures shows that the improvements in locality obtained for Hamming cliffs using the Gray encoding are obtained at the expense of a reduction elsewhere. Notice how the variances oscillate as one increases the value of δ with the maximum probability values remaining *uniformly* high throughout the entire range.

These observations raise the question as to whether it is possible to improve isotropy in a more consistent and uniform manner that incorporates the good features of both representations while avoiding the bad ones. We answer that question in the affirmative in the next section.

3 A Bit Level Mutation Operator with Isotropic Locality

One possible approach to answering this question would be to invent a new type of binary representation. In this paper we explore an alternative approach, namely, by retaining the standard binary representation and modifying the bit-flip mutation operator.

Using the standard binary encoding of an interval $[a, b]$ at a particular level of precision results in an internal representation of points in $[a, b]$ as bit strings of length l. When a particular bit k is mutated, the classical bit-flip has the phenotypic effect in $[a, b]$ of adding or subtracting a quantity proportional to 2^k. The choice of addition or subtraction is dictated by the current value of bit k and this is precisely the reason for the lack of isotropy noted in the previous

section. This fact suggests that isotropy could be improved by simply breaking this coupling.

We achieve this decoupling by modifying the standard bit-flip mutation operator as follows. When bit k is selected to be mutated, the binary value 2^k is genotypically added/subtracted independently of the current value of bit k. The choice of addition vs. subtraction is determined by flipping an unbiased coin. This results in exactly the same effects as standard bit-flip operation if bit k is a zero and addition is selected, or if bit k is a one and subtraction is selected. The remaining two cases are handled by performing standard binary addition or subtraction with overflow and underflow conditions ignored.

For example, suppose we are representing $[a, b]$ using 6-bit genotypes. Then, whenever bit 3 of a genotype is selected for mutation, a value of 001000 will be added to or subtracted from the binary value of the genotype undergoing the mutation, with the choice between addition or subtraction being determined for each such mutation by the flip of an unbiased coin. Hence, if the genotype undergoing a mutation at bit 3 is 010101, the result of applying the new (decoupled) mutation operator would be 011101 if addition were selected and 001101 in case of subtraction. Similarly, mutating the fifth bit of a genotype results in the addition or subtraction of 000010.

If multiple bits in the same genotype are selected for mutation, the application of this operator is sequential and cumulative (one can prove that the order in which bits undergo mutation is irrelevant). Extending this operator to multi-dimensional problems requires that each dimension be mapped into an independent binary gene allowing this new mutation operator to handle each gene (dimension) independently at the bit string level. In other words, overflow/underflow conditions do not propagate beyond gene boundaries.

3.1 Locality Properties of Decoupled Mutation

To see whether or not this decoupling idea truly improves isotropic locality, we performed the same analysis as we did earlier for bit-flip mutation using standard binary and Gray encodings. Figures 10 and 11 present the results.

By comparing these two figures with the earlier ones (Figures 6 - 9), one can see clearly the rather dramatic improvement in isotropic locality as reflected by the horizontal bars in Figure 10 and the lack of variance in the box plots in Figure 11. What remains to be seen is whether this improvement has a positive effect on performance.

4 Empirical Studies

To assess the effects that improved isotropic locality has on performance, we performed an initial set of empirical studies using a standard generational GA (popsize=100, 2-point crossover, fitness-proportional selection, and a mutation probability of $1/l$). We varied the representation and mutation operator as follows:

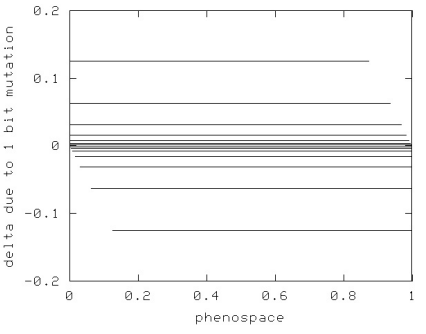

Fig. 10. A scatter plot for decoupled mutation with a standard binary encoding.

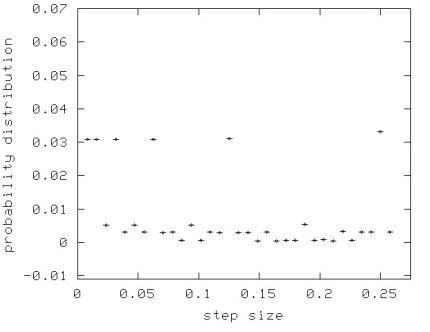

Fig. 11. Probability distribution for small step sizes δ using decoupled mutation with the standard binary encoding.

- Study 1 used the standard binary encoding and the standard bit-flip mutation operator.
- Study 2 used a Gray code representation and the standard bit-flip mutation operator.
- Study 3 used the standard binary representation with the decoupled mutation operator.

The set of landscapes used for these initial studies were HC[0,31,16,3], HC[0,31,31,3], and $Schwefel[0, 5000, 2]$. In all cases, 100 independent runs were performed and the results were averaged to obtain both mean and variance.

4.1 Hamming Cliff Landscape Results

Figures 12 - 15 present the results on the artificial Hamming cliff landscapes in terms of the effect on best-so-far performance curves. Figures 12 and 13 plot just the average best-so-far curves. What is striking is that the effects of the decoupled mutation operator operating on a standard binary representation are nearly identical to those of standard bit-flip mutation operating on a Gray code representation. Figures 14 and 15 include one standard deviation error bars to indicate the statistical advantage that both have over and EA using bit-flip mutation with a standard binary representation.

4.2 Schwefel Landscape Results

Unlike the artificial Hamming cliff landscapes, the Schwefel function provides a more realistic landscape on which to evaluate performance. Figure 16 shows the average best-so-far curves of the 3 studies on $Schwefel[0, 5000, 2]$. If we compare that with Figure 5, we see a rather striking result. On this landscape the decoupled mutation operator performed somewhat better than bit-flip mutation

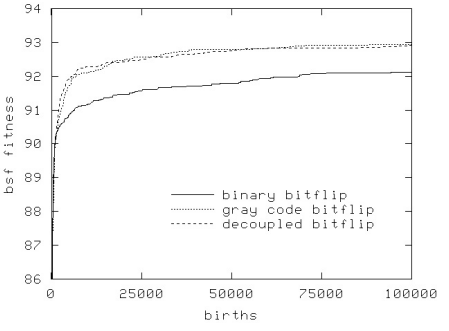

Fig. 12. Best-so-far performance on HC[0,31,16,3]

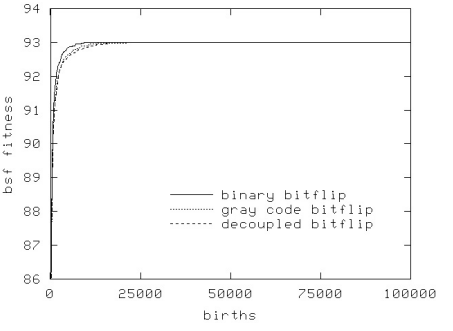

Fig. 13. Best-so-far performance on HC[0,31,31,3]

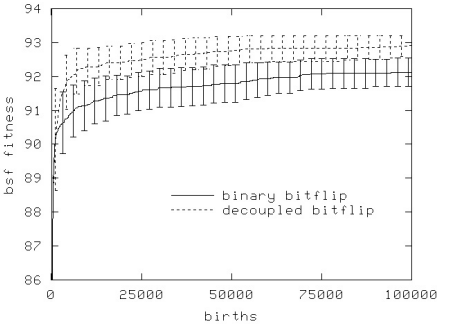

Fig. 14. Best-so-far performance on HC[0,31,16,3]

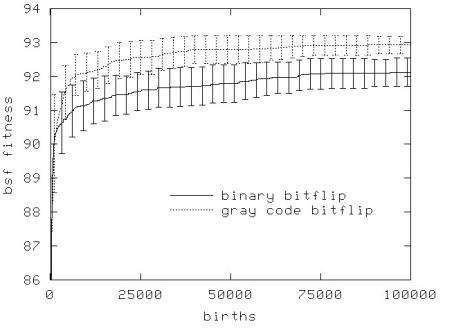

Fig. 15. Best-so-far performance on HC[0,31,16,3]

with a standard binary representation, and much better than bit-flip mutation on a Gray code representation.

Figures 17 and 18 include one standard deviation error bars and suggest why this is the case. The improved uniformity of the locality properties of decoupled mutation allow significantly more opportunity for local exploitation than bit-flip with a binary representation, resulting in faster and more consistent convergence to the global optimum. By contrast, the locality improvements of the Gray code representations are obtained at the cost of higher bias towards global exploration - with much less effectiveness.

5 Conclusions

We have illustrated some statistical properties of the effects of 1-bit-flip mutation under binary and Gray code encodings. These findings better highlight the trade-off introduced by adopting a Gray code mapping in order to improve operator locality, and result in a better understanding as to when Gray code mappings fail

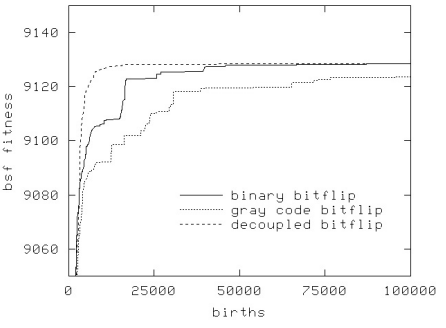

Fig. 16. Best-so-far performance on Schwefel[0,5000,2].

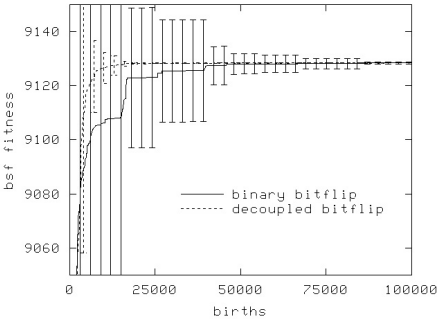

Fig. 17. Best-so-far performance on Schwefel[0,5000,2].

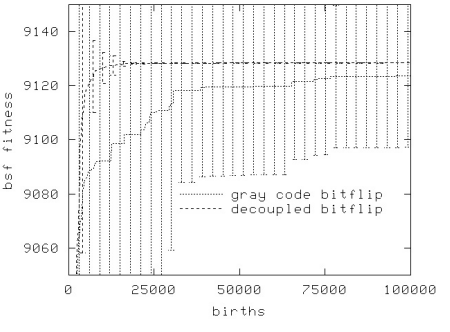

Fig. 18. Best-so-far performance on Schwefel[0,5000,2]

to outperform the standard binary mapping. This, in turn, inspired the design and implementation of the decoupled bit-flip mutation operator that has better isotropic locality properties. That is, the probability distribution of inducing a certain step size δ from a single application of the 1-bit-flip mutation operator, taken for all points in phenospace has zero variance. This property is necessary for the absence of any Hamming cliff anomalies. In addition the decoupled bit-flip mutation has a negative exponentially modulated propensity towards global exploration. This property is inherited from the classical bit-flip mutation and is unlike the Gray code mapping which has the same propensity uniformly modulated. Because of this, the decoupled bit-flip mutation is able to maintain a more effective balance between exploitation and effective exploration throughout the run.

6 Future Work

We are considering several continuations of this work, grouped in two major categories. The first area concerns itself with further study and better under-

standing of various properties and dynamics of the decoupled bit-flip mutation as introduced. Clearly, the results presented here are preliminary in nature and additional insights are likely to be obtained by extending this analysis and empirical study to various other landscapes. The second area revolves around extending the decoupled bit-flip operator to other phenospaces beyond the bounded numerical parameters. We are particularly interested in identifying the essential statistical properties of this operator that would permit the creation of a formal mechanism of such extension. An immediate first step is addressing rank ordered spaces, but a much more interesting extension would be to address partially ordered phenotypes.

Acknowledgments. This research was performed using the facilities of the Evolutionary Computation Laboratory at George Mason University and supported by ONR Grant N000140110193.

References

1. Hollstein, R.B.: Artificial Genetic Adaptation in Computer Control Systems. PhD thesis, University of Michigan (1971)
2. Caruana, R., Schaffer, J.D.: Representation and hidden bias: Gray vs. binary coding for genetic algorithms. In: Proceedings of the 5th International Workshop on Machine Learning. (1988) 153–161
3. Davis, L.: The Handbook of Genetic Algorithms. Van Nostrand Reinhold, New York (1991)
4. Schwefel, H.P.: Numerical Optimization of Computer Models. John Wiley & Sons, Ltd., Chichester (1981)
5. Hinterding, R., Gielewski, H., Peachey, T.C.: The role of mutation in the optimization of numerical functions by genetic algorithms. Technical report, Department of Computer and Mathematical Sciences, Victoria University of Technology, Victoria, Australia (1995)
6. Rothlauf, F.: The influence of binary representation of integers on the performance of selectorecombinative genetic algorithms. Technical Report 1, University of Bayreuth, Dept. of Information Systemsa (2002)
7. Whitley, D.: A free lunch proof for gray versus binary encodings. In: Proceedings of the Genetic and Evolutionary Computation Conference. (1999) 726–733
8. Chakraborty, U.K., Janikow, C.Z.: An analysis of gray versus binary encoding in genetic search. Information Sciences **156** (2003) 253–269
9. Whitley, D., Barbulescu, L., Watson, J.P.: Local search and high precision gray codes: Convergence results and neighborhoods. In: Foundations of Genetic Algorithms 6. (2000) 295–311