An analysis of the locality of binary representations in genetic and evolutionary algorithms

Hrishee Shastri and Eitan Frachtenberg

Department of Computer Science, Reed College, Portland, OR, United States of America

ABSTRACT

Mutation and recombination operators play a key role in determining the performance of Genetic and Evolutionary Algorithms (GEAs). Prior work has analyzed the effects of these operators on genotypic variation, often using locality metrics that measure the sensitivity and stability of genotype-phenotype representations to these operators. In this paper, we focus on an important subset of representations, namely nonredundant bitstring-to-integer representations, and analyze them through the lens of Rothlauf's widely used locality metrics. Our main research question is, does strong locality predict good GEA performance for these representations? Our main findings, both theoretical and empirical, show the answer to be negative. To this end, we define locality metrics equivalent to Rothlauf's that are tailored to our domain: the *point locality* for single-bit mutation and *general locality* for recombination. With these definitions, we derive tight bounds and a closed-form expected value for point locality. For general locality we show that it is asymptotically equivalent across all representations and operators. We also reproduce three established GEA empirical results to understand the predictive power of point locality on GEA performance, focusing on two popular and often juxtaposed representations: standard binary and binary-reflected Gray.

INTRODUCTION

Genetic and Evolutionary Algorithms (GEAs) solve optimization and search problems by codifying a population of possible solutions, evaluating their fitness in solving the problem, and iteratively modifying them in an attempt to improve their fitness. The digital manifestation of the solutions are called genotypes, and their interpretations into the specific problem domain are called phenotypes. The function that maps from genotypes to phenotypes is simply called the *representation*, and it can have a significant impact on the success and speed of the GEA to approximate optimal solutions (*Doerr, Klein & Storch, 2007*; *Jansen, Oliveto & Zarges, 2013*). Consequently, many empirical and theoretical studies investigated the effects of representation on GEA performance under different operators that modify genotypes. The No Free Lunch Theorems can be interpreted to show that no single representation has a performance advantage for all optimization problems (*Wolpert & Macready, 1997*).

Corresponding author
Eitan Frachtenberg, eitan@reed.edu

A key component of evolution, both biological (*Mitchell-Olds, Willis & Goldstein, 2007*) and computational (*Goldberg, Deb & Clark, 1992*), is the variation of genotypes. Perhaps the most common operator for variation in GEAs is mutation, which may be combined with another operator, recombination (*Bäck, Fogel & Michalewicz, 2018*). There are various implementations of the mutation operator, but typically they embody a localized change to an individual genotype. Contrast this with recombination, which requires two or more individuals and often involves nonlocal changes to the genotypes.

One of the most commonly used variation operators is point mutation, a simple mutation operator which randomly changes one allele at a time. There exist various forms and parameters for point mutation that control the magnitude of the genotypic change. By controlling this magnitude, mutation can be used in a search GEA both for exploration—sampling many disparate parts of the search space—and for exploitation—thoroughly searching in a localized subspace (*Črepinšek, Liu & Mernik, 2013*).

An important property of mutation that can increase the predictability and interpretability of the GEA is to have "strong locality," which we informally define here as the property that small variations in the genotype lead to small variations in the phenotype (*Pereira et al., 2006*). Strong locality implies better control of the GEA, because tuning mutation for a certain magnitude of changes in the genotype—the inputs to the search—leads to an expectation of the magnitude of changes in the phenotype— the outcome of the search. Note that the mutation operator is intricately tied to the representation. It is the combination of the mutation operator and representation that determines the magnitude of phenotypic change. Often, the discussion of locality assumes a fixed mutation operator—such as uniform bit flips or Gaussian differences—and focuses on the representations, such as standard binary or Gray.

In his seminal book, *Rothlauf (2006)* presented a theoretical framework for comparing representations based on three properties: redundancy, scaling, and locality. He proposed two metrics to quantify the locality of representations, one specifically for point mutation (simply called "locality" in the book) and one for any variation operator (called "distance distortion").

This paper focuses on these metrics when applied to a widely used subclass of representations, namely nonredundant translations from bitstring genotypes to nonnegative integer phenotypes, such as Gray encoding (*Whitley, 1999*). More precisely, we focus on representations that are one-to-one mappings from bitstrings to integers in the range $[0:2^\ell)$. These representations are very common in practice to represent integers and fixed-point values in GEAs, as well as vectors of such values. One example of such a representation is standard binary encoding (SB), where the ith bit b adds a value of exactly 2^b to the phenotype. Another example is binary-reflected Gray encoding (BRG), where the genotypes of successive phenotypes vary only by one bit (*Bitner, Ehrlich & Reingold, 1976*).

The main purpose of this paper is to study the relationship between locality and GEA performance for these representations. We explore this relationship both theoretically and empirically, and show that despite some contrary claims in the literature, strong locality does not imply good GEA performance in this domain. For example, the poor performance
[1]We define locality metrics and optimality precisely later in 'Theoretical Results on Point Locality', but for now, point locality means the mean difference in phenotype value when genotype bits are flipped randomly, and optimal point locality means that no other representation in this domain exhibits lower point locality.

of SB in some simple GEA benchmarks has been attributed to its weak locality (*Rothlauf, 2006*). In contrast, we claim that Gray encodings do not exhibit general and point locality advantages over SB under the considered domain and assumptions. In fact, we prove that both SB and BRG exhibit optimal point locality under the same benchmark conditions.[1]

Summary of contributions and paper outline

After surveying related work in 'Related Work', we examine the theoretical properties of both locality metrics. We build on Rothlauf's definitions for locality and distance distortion and redefine them specifically for the domain of binary-integer representations, which we call point locality and general locality, respectively. This lets us contribute proofs and precise computations for the following properties:

- A tight lower bound of $\frac{2^{\ell}-1}{\ell}$ for any such representation on bitstrings of length ℓ.
- A tight upper bound of $2^{\ell-1}$ for any such representation on bitstrings of length ℓ.
- A proof that the point localities of SB and BRG encodings are both identical and optimal.
- An existence proof and construction algorithm for suboptimal Gray encodings.
- A closed form for the expected value of point locality across random representations.
- A lower bound and asymptotic limit for general locality.

'Theoretical Results on Point Locality' presents these results in higher detail than in previous work (*Shastri & Frachtenberg, 2020*). To put these theoretical results in context, we also attempt in 'Experimental Results' to faithfully reproduce three distinct past experiments from three different types of GEAs that empirically compared SB's performance to BRG's. These studies found BRG to generally outperform SB, and in some cases hypothesized that this better performance is the result of stronger locality (*Hinterding, Gielewski & Peachey, 1995*; *Rothlauf, 2006*). Here, we recreate these established findings of superior BRG performance and then examine them in the newly proven perspective of the equivalent locality of the two representations. Our experiments are therefore by design not new, but rather a framework to delve deeper into the reasons for performance differences between representations.

Then, in 'Discussion', we discuss these results and produce additional experimental data and alternative explanations to BRG's superior performance. A separate contribution of this section is a discussion of the role of the GEA's simulation time on dynamically varying the representation, showing that different phases of the simulation can benefit from different locality properties of the representation. Finally, we conclude with suggestions for future work in 'Conclusion and Future Work'.

RELATED WORK

The representation in a GEA is tightly linked to its performance, which led to numerous studies working to formalize and measure the effects of representation locality (*Galván-López et al., 2011*; *Gottlieb et al., 2001*; *Gottlieb & Raidl, 1999*; *Jones & Forrest, 1995*; *Manderick, 1991*; *Ronald, 1997*). Many of these studies quantify locality in different ways and apply it to different phenotype classes, such as floating-point numbers (*Pereira et al., 2006*), computer programs (*Galván-López et al., 2011*; *Rothlauf & Oetzel, 2006*),

permutations (*Galván-López & O'Neill, 2009*), and trees (*Hoai, McKay & Essam, 2006*; *Rothlauf & Goldberg, 1999*). The focus of these approaches is often to measure the effect of genotypic changes on fitness distances (*Gottlieb & Raidl, 1999*). A more general approach is to instead measure the effect on phenotypic distances, because it provides a way to measure the locality of a representation that is independent of the fitness function (*Rothlauf, 2003*).

The foundational locality definitions for our study and several others (*Chiam, Goh & Tan, 2006*; *Galván-López et al., 2010*) come from Rothlauf's treatise on the theory of GEA representations (*Rothlauf, 2006*). Rothlauf defines the locality d_m as:

$$d_m \stackrel{\text{def}}{=} \sum_{d_{x,y}^g = d_{min}^g} |d_{x,y}^p - d_{min}^p| \quad \text{(ibid. Eq. 3.23)}, \tag{1}$$

[2] This definition has changed from the book's first edition to better match reader intuition and for simpler computation.

where for every two distinct genotypes x, y, $d_{x,y}^g$ is the genotypic distance between x and y, and $d_{x,y}^p$ is the phenotypic distance between their respective phenotypes, based on our choices of genotypic and phenotypic spaces. Similarly, d_{min}^g and d_{min}^p represent the minimum possible distance between genotypes or phenotypes, respectively.[2] For example, for nonredundant representations of integers as bitstrings (the focus of this paper), genotypic distances are measured in Hamming distance and phenotypic distances use the usual Euclidean metric in $\mathbb{N}$.

This definition "describes how well neighboring genotypes correspond to neighboring phenotypes" (ibid. p. 77), which is valuable for measuring small genotypic changes that typically result from mutation. Extending this notion to include large genotypic changes (e.g., from a recombination operator), Rothlauf defines the *distance distortion* d_c as:

$$d_c \stackrel{\text{def}}{=} \frac{2}{n_p(n_p-1)} \sum_{i=1}^{n_p} \sum_{j=i+1}^{n_p} |d_{x_i,x_j}^p - d_{x_i,x_j}^g| \quad \text{(ibid. Eq. 3.24)}, \tag{2}$$

where n_p is the size of the search space, and $d_{x_i,x_j}^p, d_{x_i,x_j}^g$ are the phenotypic and genotypic distance, respectively, between two distinct individuals x_i, x_j. The term $\frac{2}{n_p(n_p-1)}$ is equal to $\frac{1}{\binom{n_p}{2}}$, the proportion of each distinct pair of individuals. This definition "describes how well the phenotypic distance structure is preserved when mapping Φ_p on Φ_g", where Φ_p is the phenotypic search space and Φ_g is the genotypic search space (ibid. p. 84).

As observed by *Galván-López et al. (2011)*, high values of d_m and d_c actually denote low locality while low values denote high locality. To avoid confusion, we will refer to low metric values as strong locality and high metric values as weak locality.

Similar in spirit, *Gottlieb & Raidl (1999)* also defined a pair of locality metrics for mutation and crossover operators called *mutation innovation* and *crossover innovation*. They additionally defined *crossover loss*, which measures the number of phenotypic properties that are lost by crossover. These metrics are probabilistic and empirical in nature, so they are harder to reason about analytically. But they have been demonstrated in practice to predict GEA performance on the multidimensional knapsack problem (*Pereira et al., 2006*; *Raidl & Gottlieb, 2005*).

In a different study, *Chiam, Goh & Tan (2006)* defined the concept of *preservation*, which "measure[s] the similarities between the genotype and phenotype search space."

Their study uses Hamming distance between genotypes and L2 norms between phenotypes to define analogous metrics to Rothlauf's (called *proximity preservation* and *remoteness preservation*). Unlike Rothlauf's metrics, their metrics look in both directions of the genotype-phenotype mapping. The authors demonstrated with examples (as we prove formally in the next section) that SB and BRG have the same genotype-to-phenotype locality, but not phenotype-to-genotype locality. They also predicted, based on this similarity, that crossover-based GEAs would perform about the same with both SB and BRG encodings, which appears to contradict some past experimental results, as we discuss in 'Genetic algorithms'.

A different approach to approximating locality was given by *Mathias & Whitley (1994b)*, and independently, by *Weicker (2010)* using the various norms of the phenotype-genotype distance matrix for different representations. These studies did not draw performance predictions from these metrics.

Another interesting aspect of locality of representation is its effect over time. Some studies explored the effects of problems that are themselves dynamic in nature, therefore changing their representation over time (*Mueller-Bady et al., 2017*; *Neumann & Witt, 2015*). One such study benchmarked delta coding on the same five De Jong functions as we have. Their algorithm monitors the population diversity, and once the population becomes more homogeneous, it changes the representation to search different subpartitions of the phenotypic hyperspace. This approach performed well compared to contemporaneous state-of-the-art GEA implementations such as CHC and GENITOR (*Mathias & Whitley, 1994a*). Another example applied a hybrid GEA to the maximum clique problem that changes mutation rates based on the phase of the simulation (*Ouch, Reese & Yampolskiy, 2007*).

For any search function, there are multiple representations that make the optimization problem trivial (*Liepins & Vose, 1990*), so it makes sense to try different representations dynamically. Whitley, Rana, and Heckendorn even computed the expected number of local optima for a random representation (*Whitley, Rana & Heckendorn, 1997*). The first two authors also introduced the concept of *shifting* to dynamically switch from one Gray representation to another to escape local optima (*Rana & Whitley, 1997*). *Barbulescu, Watson & Whitley (2000)* expanded this work by proving a bound on the number of Gray representations available via shifting. They showed that to improve performance, the GEA needs to shift to a dissimilar representation to the current one.

Shifting between representations occurs sequentially in these works, but can also be evaluated in parallel using a modified island model (*Skolicki & De Jong, 2004*). In all these cases, however, the different representations are either standard binary or Gray. These studies have not explored the possibility of changing to a random representation with different locality to evaluate its effect on GEA performance.

Whitley (2000) did compute summary statistics for any random permutation function, but these did not include average locality. To offer a framework of understanding the effect of arbitrary representations on locality, we also compute in 'Expected point locality' the expected point locality of any random representation.

THEORETICAL RESULTS ON POINT LOCALITY

Defining point locality

Recall Rothlauf's definition for locality (Eq. (1)). Since we are for the moment specifically concerned with single-bit mutations, both d_{min}^g and d_{min}^p are equal to 1. And since we limit ourselves to binary-integer representations, all phenotypic distances fall in the range $[0, 2^\ell - 1]$ and all genotypic distances fall in the range $[0, \ell]$. We can use these assumptions to precisely define our own derived metric, *point locality*, using units that we find more intuitive.

We first define a representation $r : \{0,1\}^\ell \rightarrow [0, 2^\ell)$ as a bijection between the set of ℓ-bit bitstrings $\{0,1\}^\ell$ and the discrete integer interval $[0, 2^\ell)$. This ensures that the representation is not redundant –i.e., every integer in the interval $[0, 2^\ell)$ is represented by exactly one ℓ-bit bitstring, and the number of search-space points n_p is exactly 2^ℓ. A representation r can therefore be equivalently described as a permutation $\pi : [0, 2^\ell) \rightarrow [0, 2^\ell)$, where $\pi(i) = j$ if and only if the SB representation of i maps to j under r. Consequently, we can write π as a 2^ℓ-tuple where the ith coordinate (starting at 0) is π (i^{th}). We also use the notation $\hat{s}_i$ to denote the binary string produced from flipping the ith coordinate of the binary string $s \in \{0,1\}^\ell$. Formally, $\hat{s}_i = s \oplus 2^i$, where $\oplus$ denotes bitwise exclusive-or.

It is worth noting two important assumptions about this definition that limit the generalization of our results. First, we only consider nonredundant representations, because they are the most efficient from an information-theoretic point of view. In other words, they represent the most phenotypic values for a given bitstring length ℓ. Second, we map all phenotypic values to the range $[0 : 2^\ell)$. This is primarily a choice of convenience. Nonredundant phenotypes could could come in any integer range of size 2^ℓ, continuous or not. If a representation is nonredundant or noncontinuous, our theoretical results no longer necessarily hold. However, we believe that many practical applications of GEAs do in fact represent integers as we have, perhaps because the SB encoding is supported in hardware by all extant computer systems and is therefore extremely fast to translate in practice.

We can now define the *point locality* p_r for a nonredundant bitstring-to-integer representation r as the average change in phenotypic value for a uniformly random single-bit flip in the genotype. More formally:

Definition 1 *The point locality p_r for a representation r is $p_r \stackrel{\text{def}}{=} \overline{|r(\hat{s}_i) - r(s)|} \; \forall s \in \{0,1\}^\ell, \forall i \in [0, \ell)$. Explicitly,*

$$p_r \stackrel{\text{def}}{=} \frac{\sum_s \sum_i |r(\hat{s}_i) - r(s)|}{2^\ell \cdot \ell}. \tag{3}$$

Note that for any given ℓ, our definition of p_r is a simple linear transformation of Rothlauf's d_m. In our domain, d_m simply sums the phenotypic distances minus one between all distinct genotypic neighbors, while p_r computes the average phenotypic distance between all ordered pairs of genotypic neighbors. Also note that in Rothlauf's d_m, d_{min}^p occurs in the summation $\ell 2^\ell$ times. Coupled with the fact that $d_{min}^p = 1$ in our

domain, we have the relationship:

$$p_r = \frac{2d_m + \ell 2^\ell}{\ell 2^\ell} = \frac{d_m}{\ell 2^{\ell-1}} + 1.$$

Computing tight bounds for point locality

Our first analysis proves lower and upper bounds on p_r, computes p_r for SB and Gray representations, and verifies the existence of representations with both minimum and maximum p_r.

Lower bound

Theorem 1 (Lower bound): $p_r \geq \frac{2^\ell - 1}{\ell}$ *for nonredundant binary-integer representations,*

Proof We reduce the problem of minimizing locality to another problem, that of enumerating nodes on a hypercube while minimizing neighbor distances, for which lower and upper bounds have been established by *Harper (1964)*. We use the term $\Delta_{ij} \overset{\text{def}}{=} |i - j|$ to denote the absolute difference between the numbers assigned to two adjacent vertices i and j on the unit ℓ-cube. The unit ℓ-cube consists of all elements in $\{0, 1\}^\ell$. Two vertices in the ℓ-cube are adjacent if they differ by only one bit (i.e., they have Hamming distance 1). Note that assigning a number n to a vertex i can be thought of as a representation r mapping i to n, or $r(i) = n$. Therefore, we have $\Delta_{\hat{s}_i s} = |r(\hat{s}_i) - r(s)|$ for adjacent vertices $\hat{s}_i$ and s, since a 1-bit difference is equivalent to a single bit-flip mutation. Following *Harper (1964)*, we define the sum $\sum \Delta_{\hat{s}_i s}$ to be the sum of the absolute difference between two adjacent vertices $\hat{s}_i$ and s that runs over all possible pairs of neighboring vertices in the ℓ-cube. Note that

$$2\sum \Delta_{\hat{s}_i s} = \sum_s \sum_i |r(\hat{s}_i) - r(s)|,$$

since the RHS computes $|r(\hat{s}_i) - r(s)|$ twice for every ordered pair.

Harper proved that $\sum \Delta_{\hat{s}_i s} \geq 2^{\ell-1}(2^\ell - 1)$. Therefore,

$$
\begin{aligned}
p_r &= \frac{\sum_s \sum_i |r(\hat{s}_i) - r(s)|}{2^\ell \cdot \ell} \\
&= \frac{2\sum \Delta_{\hat{s}_i s}}{2^\ell \cdot \ell} \\
&\geq \frac{2(2^{\ell-1}(2^\ell - 1))}{2^\ell \cdot \ell} \quad (ibid.) \\
&= \frac{2^\ell - 1}{\ell},
\end{aligned}
$$

proving that $p_r \geq \frac{2^\ell - 1}{\ell}$.

Corollary 2 *Standard binary encoding is optimal, meaning that it has the strongest (lowest) point locality, equal to* $\frac{2^\ell - 1}{\ell}$.

Proof We prove optimal locality by direct computation of the sum of distances for this encoding. SB encoding is also a representation—call it *SB*. We consider p_{SB}:

$$p_{SB} = \frac{\sum_s \sum_i |SB(\hat{s}_i) - SB(s)|}{2^\ell \cdot \ell}.$$

The inner sum $\sum_i |SB(\hat{s}_i) - SB(s)|$ computes the sum of all differences obtained from flipping the ith bit of a given SB string s. Flipping the ith bit elicits an absolute phenotypic difference of 2^i for any s and i, reducing the inner sum to:

$$\sum_i |SB(\hat{s}_i) - SB(s)| = \sum_{i=0}^{\ell-1} 2^i = 2^\ell - 1.$$

Now since there are 2^ℓ elements in $\{0,1\}^\ell$, the outer sum reduces to $\sum_s (2^\ell - 1) = (2^\ell)(2^\ell - 1)$. Combining these lets us compute p_{SB}:

$$\begin{aligned}
p_{SB} &= \frac{\sum_s \sum_i |SB(\hat{s}_i) - SB(s)|}{2^\ell \cdot \ell} \\
&= \frac{2^\ell (2^\ell - 1)}{2^\ell \cdot \ell} \\
&= \frac{2^\ell - 1}{\ell},
\end{aligned}$$

which is the lower bound given by Theorem 1. Thus, SB has optimal point locality. $\square$

Upper bound

Theorem 3 (Upper bound): $p_r \leq 2^{\ell-1}$ *for nonredundant binary-integer representations,*

Proof We rely on another result from *Harper (1964)* in which he proved that $\sum \Delta_{\hat{s}_i s} \leq \ell 2^{2(\ell-1)}$. We have

$$\begin{aligned}
p_r &= \frac{\sum_s \sum_i |r(\hat{s}_i) - r(s)|}{2^\ell \cdot \ell} \\
&= \frac{2 \sum \Delta_{\hat{s}_i s}}{2^\ell \cdot \ell} \\
&\leq \frac{2(\ell 2^{2(\ell-1)})}{2^\ell \cdot \ell} \quad (ibid.) \\
&= \frac{2^{2(\ell-1)}}{2^{\ell-1}} \\
&= 2^{\ell-1}.
\end{aligned}$$

Thus $p_r \leq 2^{\ell-1}$. $\square$

Claim 4 *There exists a nonredundant binary-integer representation r with upper bound point locality $p_r = 2^{\ell-1}$.*

Proof Here, we reduce the problem of constructing a representation r with upper bound locality $p_r = 2^{\ell-1}$ to that of assigning integers in $[0, 2^\ell)$ to vertices in the ℓ-cube, such that $\sum \Delta_{\hat{s}_i s}$ is maximized. *Harper (1964)* constructed an algorithm assigning numbers to vertices to maximize $\sum \Delta_{\hat{s}_i s}$, shown in Algorithm 1. Maximizing $\sum \Delta_{\hat{s}_i s}$ is equivalent to maximizing p_r, so such a representation exists.

Algorithm 1: Harper's algorithm to assign integers on the ℓ-cube to maximize $\sum \Delta_{\hat{s}_i,s}$, the sum of neighbor differences. The function assign(v, n) assigns the number n to ℓ-cube vertex v. $|v|_1$ denotes the number of '1' bits in vertex v.

---

1   $V \leftarrow \{0,1\}^\ell$;

2   $s \leftarrow$ randomly selected node from V;

3   assign$(s, 0)$;

4   parity $\leftarrow |s|_1 \bmod 2$;

5   $L_1 \leftarrow$ randomly shuffled $[1, ..., 2^{\ell-1} - 1]$;

6   $L_2 \leftarrow$ randomly shuffled $[2^{\ell-1}, ..., 2^\ell - 1]$;

7   $i \leftarrow 0$;

8   $j \leftarrow 0$;

9   **for** $v \in V \setminus \{s\}$ **do**

10   **if** $|v|_1 \bmod 2 = parity$ **then**

11   assign$(v, L_1[i])$;

12        $i \leftarrow i + 1$;

13   **else**

14   assign$(v, L_2[j])$;

15        $j \leftarrow j + 1$;

Computing point locality for Gray encodings

Gray encodings are encodings where two phenotypic neighbors are also genotypic neighbors. In our domain, this means that any two phenotypes that are separated by one unit are represented by two genotypes that are separated by one bit flip. For a given length ℓ, there are multiple Gray encodings, but the most common is BRG. Here, we show that BRG has equivalent point locality to SB, and that not all Gray encodings share the same locality value.

Point locality of BRG

Claim 5 *Binary Reflected Gray (BRG) encoding is also optimal.*

This claim can be derived from Harper's construction algorithm for optimal representations, which can be shown to also construct BRG. Instead, we show an original proof by induction.

Proof Let *BRG* notate the representation for Binary Reflected Gray , so

$$p_{BRG} = \frac{\sum_s \sum_i |BRG(\hat{s}_i) - BRG(s)|}{2^\ell \cdot \ell}.$$

We prove this claim by breaking it up into two lemmas and proving them separately. The first lemma computes the sum of differences from mutating the leftmost (most-significant) bit in BRG. We use this result in the second lemma to compute the sum of differences from mutating all bits in BRG.

Lemma 6 $\sum_{s \in \{0,1\}^\ell} |BRG(\hat{s}_{\ell-1}) - BRG(s)| = 2^{2\ell-1}$.

In other words, the sum of the differences obtained by flipping the leftmost bit over all ℓ-bit bitstrings in BRG encoding is $2^{2\ell-1}$.

Proof Consider the recursive nature of BRG codes (*Rowe et al., 2004*). Let L_ℓ be the ordered list of ℓ-bit BRG codes where $L_\ell[i]$ is the bitstring that maps to i. Note that 2^ℓ is the length of L_ℓ and [ ] denotes list indexing. The left half of L_ℓ contains $L_{\ell-1}$ prefixed with 0 and the right half of L_ℓ contains $L_{\ell-1}$ in reverse order prefixed with 1. Flipping the leftmost bit of $L_\ell[i]$ will yield $L_\ell[2^\ell - 1 - i]$. Thus we have

$$\sum_{s\in\{0,1\}^\ell} |BRG(\hat{s}_{\ell-1}) - BRG(s)| = \sum_{i=0}^{2^\ell-1} |BRG(L_\ell[2^\ell - 1 - i]) - BRG(L_\ell[i])|$$

$$= \sum_{i=0}^{2^\ell-1} |2^\ell - 1 - i - i|$$

$$= \sum_{i=0}^{2^\ell-1} |2^\ell - (2i+1)|.$$

Note that for a given $i \in [0, 2^\ell - 1]$,

$$|2^\ell - (2i+1)| = \begin{cases} 2^\ell - (2i+1) & i < 2^\ell/2 \\ (2i+1) - 2^\ell & i \geq 2^\ell/2, \end{cases}$$

which lets us split the sum to

$$= \sum_{i=0}^{2^{\ell-1}-1} (2^\ell - (2i+1)) + \sum_{i=2^{\ell-1}}^{2^\ell-1} ((2i+1) - 2^\ell)$$

$$= \sum_{i=0}^{2^{\ell-1}-1} 2^\ell - \sum_{i=0}^{2^{\ell-1}-1} (2i+1) + \sum_{i=2^{\ell-1}}^{2^\ell-1} (2i+1) - \sum_{i=2^{\ell-1}}^{2^\ell-1} 2^\ell$$

$$= (2^{\ell-1})(2^\ell) - \sum_{i=0}^{2^{\ell-1}-1} (2i+1) + \sum_{i=2^{\ell-1}}^{2^\ell-1} (2i+1) - (2^{\ell-1})(2^\ell)$$

$$= \sum_{i=2^{\ell-1}}^{2^\ell-1} (2i+1) - \sum_{i=0}^{2^{\ell-1}-1} (2i+1).$$

Splitting the left sum and using the facts that the sum of the first n odd numbers is n^2,

$$= \left(\sum_{i=0}^{2^\ell-1} (2i+1) - \sum_{i=0}^{2^{\ell-1}-1} (2i+1) \right) - (2^{\ell-1})^2$$

$$= (2^\ell)^2 - (2^{\ell-1})^2 - (2^{\ell-1})^2$$

$$= 2^{2\ell} - 2 \cdot 2^{2\ell-2}$$

$$= 2^{2\ell} - 2^{2\ell-1}$$

$$= 2^{2\ell}(1 - \frac{1}{2})$$

$$= 2^{2\ell-1},$$

Thus $\sum_{s\in\{0,1\}^\ell}|BRG(\hat{s}_{\ell-1})-BRG(s)|=2^{2\ell-1}$.

Lemma 7 $\sum_s\sum_i|BRG(\hat{s}_i)-BRG(s)|=2^{2\ell}-2^\ell$.

Proof We proceed with induction on ℓ. For the base case ($\ell=1$), the set $\{0,1\}$ contains two BRG codes, which correspond to the integers 0 and 1, respectively. Thus $\sum_{s\in\{0,1\}}\sum_{i=0}^{1-1}|BRG(\hat{s}_i)-BRG(s)|=1+1=2=2^{2\cdot1}-2^1$.

For the inductive hypothesis (I.H.), assume $\sum_s\sum_i|BRG(\hat{s}_i)-BRG(s)|=2^{2\ell}-2^\ell$ for some $\ell\in\mathbb{N}$. We must now show that $\sum_{s\in\{0,1\}^{\ell+1}}\sum_{i=0}^{\ell}|BRG(\hat{s}_i)-BRG(s)|=2^{2(\ell+1)}-2^{(\ell+1)}$. Note that in the inductive step, we are working with strings of length $\ell+1$.

$$\sum_{s\in\{0,1\}^{\ell+1}}\sum_{i=0}^{\ell}|BRG(\hat{s}_i)-BRG(s)|$$

$$=\sum_{s\in\{0,1\}^{\ell+1}}\left(|BRG(\hat{s}_\ell)-BRG(s)|+\sum_{i=0}^{\ell-1}|BRG(\hat{s}_i)-BRG(s)|\right)$$

$$=\sum_{s\in\{0,1\}^{\ell+1}}\sum_{i=0}^{\ell-1}|BRG(\hat{s}_i)-BRG(s)|+\sum_{s\in\{0,1\}^{\ell+1}}|BRG(\hat{s}_\ell)-BRG(s)|.$$

By I.H. and the fact that there are two copies of the ℓ-bit BRG code in the $(\ell+1)$-bit BRG code,

$$=2\cdot(2^{2\ell}-2^\ell)+\sum_{s\in\{0,1\}^{\ell+1}}|BRG(\hat{s}_\ell)-BRG(s)|.$$

By Lemma 6,

$$=2\cdot(2^{2\ell}-2^\ell)+2^{2(\ell+1)-1}$$
$$=2^{2\ell+1}-2^{\ell+1}+2^{2\ell+1}$$
$$=2\cdot2^{2\ell+1}-2^{\ell+1}$$
$$=2^{2(\ell+1)}-2^{(\ell+1)}.$$

$\square$

Now we can prove Corollary 5 Considering p_{BRG}:

$$p_{BRG}=\frac{\sum_s\sum_i|BRG(\hat{s}_i)-BRG(s)|}{2^\ell\cdot\ell}$$
$$=\frac{2^{2\ell}-2^\ell}{2^\ell\cdot\ell}\quad\text{Lemma 7}$$
$$=\frac{2^\ell(2^\ell-1)}{2^\ell\cdot\ell}$$
$$=\frac{2^\ell-1}{\ell},$$

which is the lower bound given by Theorem 1. Therefore, BRG has optimal point locality.

$\square$

Note that this equivalence in point locality between SB and BRG has already been demonstrated empirically for small values of ℓ (*Chiam, Goh & Tan, 2006*), but our proof holds for all values of ℓ.

Suboptimal Gray encodings

We now shift to generating a Gray encoding with suboptimal point locality, which requires the following lemma:

Lemma 8 *For $\ell \geq 3$, the ℓ-cube Q_ℓ always contains a Hamiltonian path starting with the sequence of nodes $0^{\ell-3}000$, $0^{\ell-3}001$, $0^{\ell-3}011$, $0^{\ell-3}111$, where the notation 0^x denotes a length- x bitstring of all 0s.*

Proof Recall that the ℓ-cube contains 2^ℓ vertices labeled as binary strings in $\{0,1\}^\ell$, where vertex i is connected to vertex j if and only if i and j have Hamming distance 1. We proceed by induction on ℓ.

For the base case ($\ell = 3$), the path $000, 001, 011, 111, 101, 100, 110, 010$ is Hamiltonian. For the inductive step, assume that Q_ℓ has a Hamiltonian path starting with $0^{\ell-3}000$, $0^{\ell-3}001$, $0^{\ell-3}011$, $0^{\ell-3}111$. Consider $Q_{\ell+1}$. By the inductive step, we can trace a path starting with the sequence $0^{\ell+1}, 0^{\ell-2}001, 0^{\ell-2}011, 0^{\ell-2}111$ to some node $0v$ ($v \in \{0,1\}^\ell$) such that the first bit never flips to 1. From there, we hop to $1v$, then trace out the remainder of the path until all nodes have been visited, as nodes of the form $1x$ for all $x \in \{0,1\}^\ell$ constitute another copy of Q_ℓ. The resulting path is Hamiltonian, since every node has been visited exactly once.

Claim 9 *There exists a Gray encoding g with suboptimal point locality $p_g > \frac{2^\ell-1}{\ell}$ for any $\ell \geq 3$.*

Proof A bitstring $b \in \{0,1\}^\ell$ has ℓ neighbors that are all Hamming distance one away. We can thus reduce the problem of constructing a Gray code to that of constructing a Hamiltonian path on the ℓ-cube. Recall that in Theorem 1 we mapped the problem of minimizing locality to that of minimizing $\sum \Delta_{\hat{s}_i s}$. In the same paper, *Harper (1964)* formulates an algorithm that provably generates all representations that minimize $\sum \Delta_{\hat{s}_i s}$, which we describe in Algorithm 2:

Our goal is to construct a Hamiltonian path that violates this algorithm. This path in turn will determine a Gray code that has suboptimal point locality, because Harper's algorithm generates all representations with optimal point locality.

Our modified algorithm starts by assigning 0 to vertex $0^{\ell-3}000$. We then assign 1 to $0^{\ell-3}001$ and assign 2 to $0^{\ell-3}011$. Algorithm 2 would force us to assign 3 to $0^{\ell-3}010$ if we wanted to produce an optimal Gray code. Instead, we assign 3 to $0^{\ell-3}111$, which violates the algorithm. The remainder of the path can be traversed arbitrarily such that it is Hamiltonian, due to Lemma 8. This construction generates a Gray code g with $p_g > \frac{2^\ell-1}{\ell}$ for $\ell \geq 3$. $\square$

Algorithm 2: Harper's algorithm to assign integers on the hypercube $\{0,1\}^{\ell}$ to minimize $\sum \Delta_{\hat{s}_i,s}$, the sum of neighbor differences. The function assign(v,n) assigns the number n to hypercube vertex v.

---

1   $V \leftarrow \{0,1\}^{\ell}$;

2   $s \leftarrow$ randomly selected node from V;

3   assign$(s,0)$;

4   $i \leftarrow 1$;

5   **while** $i \leq 2^{\ell} - 1$ **do**

6 | $v \leftarrow$ unassigned node in V with highest number of already assigned neighbors;
      `/* If there are multiple nodes with this property, choose one at`
      `   random                                                      */`

7 | assign(v,i);

8 | $i \leftarrow i+1$;
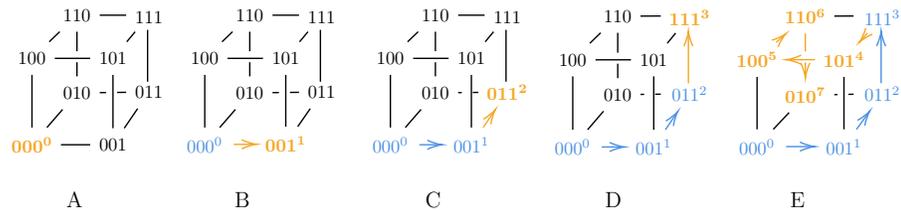
Figure 1   **A visualization of our modified algorithm from Claim 9 used to generate g.** In step D, we assign 3 to 111 instead of 010, violating Algorithm 2 and thus ensuring g is a suboptimal Gray code.

Example   *Consider a 3-bit Gray code representation g, given by the permutation $\pi = [0,1,3,7,5,4,6,2]$:*

$000 \mapsto 0$

$001 \mapsto 1$

$011 \mapsto 2$

$111 \mapsto 3$

$101 \mapsto 4$

$100 \mapsto 5$

$110 \mapsto 6$

$010 \mapsto 7.$

Then g is a Gray code generated by 9, and $p_g \approx 3.667$ which is greater than the lower bound $\frac{2^3-1}{3} \approx 2.333$ from Theorem 1. A visualization of how our modified algorithm from 9 generates g appears in Fig. 1. Note that in step D, we assign 3 to 111, which has fewer neighbors than 010 that are already labeled.

Also note that Algorithm 2 can generate both SB and BRG codes, as illustrated in Figs. 2 and 3.
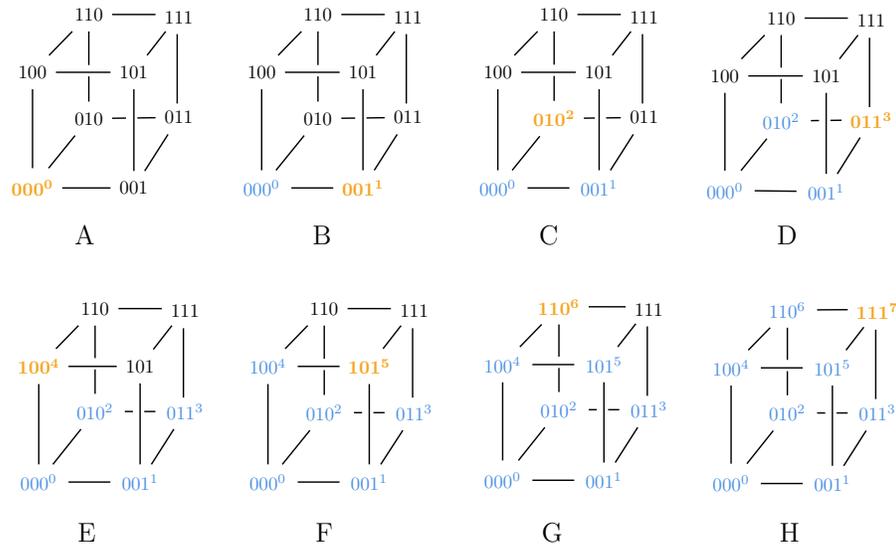
Figure 2 Algorithm 2 generating SB for $\ell = 3$.
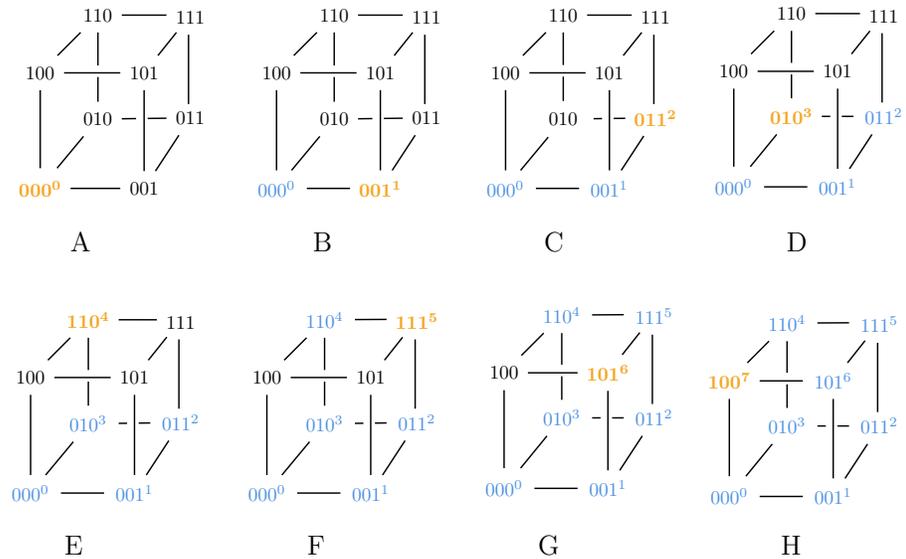
Figure 3 Algorithm 3 generating BRG for $\ell = 3$.

Expected point locality

Our final proof on point locality answers the question: "what is the expected point locality of an arbitrary representation?"

Theorem 10 (Expected Point Locality): $\mathbb{E}[p_r] = \frac{2^\ell + 1}{3}$ *for nonredundant binary-integer representations.*

Proof We prove this by direct computation, using a few identities, starting with the definition of p_r:

$$p_r \stackrel{\text{def}}{=} \frac{\sum_s \sum_i |r(\hat{s}_i) - r(s)|}{\ell 2^\ell}$$

$$\implies \mathbb{E}[p_r] = \frac{1}{\ell 2^\ell} \mathbb{E}\left[\sum_s \sum_i |r(\hat{s}_i) - r(s)|\right]. \tag{4}$$

Let $L = \{0, \ldots, 2^\ell - 1\}$. Computing the expectation in Eq. (4) is equivalent to assigning numbers in L to vertices of the ℓ-cube uniformly at random, and then summing the absolute differences between every adjacent pair. Let k denote the value assigned to vertex s, and let π_k denote a size ℓ random subset of $L \setminus \{k\}$. Due to linearity of expectation, the sum of absolute differences for each vertex s can be examined independently even though they are dependent variables. Then, s has ℓ neighbors where the ith neighbor is assigned the ith element in π_k, which we denote $\pi_k(i)$. This gives us the following relation (letting $n = 2^\ell$ for simplicity):

$$\mathbb{E}\left[\sum_s \sum_i |r(\hat{s}_i) - r(s)|\right] = \mathbb{E}\left[\sum_{k=0}^{n-1} \sum_{i=1}^{\ell} |\pi_k(i) - k|\right]. \tag{5}$$

Since every number in $L \setminus \{k\}$ has equal probability of appearing as the ith element of π_k, $\mathbb{E}[\pi_k(i)]$ does not depend on i at all. Therefore, once we apply linearity of expectation on the right side of Eq. (5), we can replace $\pi_k(i)$ with a discrete uniform random variable sampled from $L \setminus \{k\}$. Denote this random variable as X^k. We then have:

$$= \ell \sum_{k=0}^{n-1} \mathbb{E}[|X^k - k|]$$

$$= \ell \sum_{k=0}^{n-1} \sum_{x \in L \setminus \{k\}} |x - k| \cdot \mathrm{P}(X^k = x)$$

$$= \frac{\ell}{n-1} \sum_{k=0}^{n-1} \sum_{x \in L \setminus \{k\}} |x - k|$$

$$= \frac{\ell}{n-1} \sum_{k=0}^{n-1} \left(S_{1,k} + S_{1,n-k-1}\right),$$

where we let $S_{i,j} = \sum_{m=i}^{j} m$. Continuing with the fact that $S_{1,m} = \frac{m(m+1)}{2}$:

$$= \frac{\ell}{2(n-1)} \sum_{k=0}^{n-1} ((n-k-1)(n-k) + k(k+1))$$

$$= \frac{\ell}{2(n-1)} \left(\sum_{k=0}^{n-1} (n-k)^2 - \sum_{k=0}^{n-1} (n-k) + \sum_{k=0}^{n-1} k^2 + \sum_{k=0}^{n-1} k\right).$$

Let $Q_{i,j} = \sum_{m=i}^{j} m^2$. Then

$$= \frac{\ell}{2(n-1)} \left(Q_{1,n} - S_{1,n} + Q_{1,n-1} + S_{1,n-1} \right)$$

$$= \frac{\ell}{2(n-1)} \left(2Q_{1,n-1} + n^2 - n \right).$$

Using the fact that $Q_{1,m} = \frac{m(m+1)(2m+1)}{6}$ and simplifying:

$$= \frac{\ell}{2(n-1)} \cdot \frac{2n(n-1)(n+1)}{3}$$

$$= \frac{\ell 2^{\ell}(2^{\ell}+1)}{3}.$$

Substituting back into Eq. (4) yields:

$$\mathbb{E}[p_r] = \frac{1}{\ell 2^{\ell}} \cdot \frac{\ell 2^{\ell}(2^{\ell}+1)}{3}$$

$$= \frac{2^{\ell}+1}{3}.$$

$\square$

We have verified this value experimentally for $1 \le \ell \le 8$. It is interesting to note that the expected asymptotic value differs from the worst-case (maximum) locality by only a constant factor, and grows further apart with ℓ from the optimal locality.

Having explored the properties of point locality for single-bit mutations, we now turn our attention to general locality and distance distortion for any variation operator.

THEORETICAL RESULTS ON GENERAL LOCALITY

General locality is a measurement of the average perturbation of a phenotype for any arbitrary change in genotype, such as the one created by a crossover operator. Again, we refine Rothlauf's definition for our representation domain. As it turns out, this metric proves to carry no useful information about representations in this domain, because the phenotypic range grows much faster with ℓ than the genotypic range.

Defining general locality

Again we start with Rothlauf's definition, called distance distortion (Eq. (2)), and develop an equivalent definition, called *general locality*, that is tailored to our domain. We define the general locality g_r for a representation r as the mean difference between phenotypic and genotypic distances between each unique pair of individuals. More formally:

Definition 2 *The general locality for a representation* r *is*

$$g_r \stackrel{\text{def}}{=} \frac{1}{\binom{2^{\ell}}{2}} \sum_{(s_1,s_2) \in S_{\ell}} |d^p(s_1,s_2) - d^g(s_1,s_2)|, \tag{6}$$

where S_{ℓ} *is the set of all unordered pairs from* $\{0,1\}^{\ell}$ *(so* $|S_{\ell}| = \binom{2^{\ell}}{2}$*),* $d^p(s_1,s_2)$ *is the phenotypic distance* $|r(s_1) - r(s_2)|$*, and* $d^g(s_1,s_2)$ *is the genotypic (Hamming) distance*

between s_1 and s_2. Note that our definition of general locality mirrors Chiam's remoteness preservation and is also equivalent to Rothlauf's d_c metric, since $\sum_{i=0}^{n_p}\sum_{j=i+1}^{n_p}$ is identical to $\sum_{(s_1,s_2)\in S_\ell}$.

We begin our analysis of general locality by proving a lower bound on its value, and continue by proving the asymptotic equivalence of all nonredundant binary integer representations under this metric.

Lower bound for general locality

We compute the tight lower bound on general locality for our domain since it gives an order-of-magnitude estimate for its value, as well as a framework to compute its actual asymptotic value.

Theorem 11 (Lower bound) $g_r \geq \frac{1}{\binom{2^\ell}{2}}(\frac{1}{6}(2^\ell-1)(2^\ell)(2^\ell+1)-\ell 2^{2(\ell-1)})$ *for nonredundant binary-integer representations.*

Proof By definition, $g_r = \frac{1}{\binom{2^\ell}{2}}\sum_{(s_1,s_2)\in S_\ell}|d^p(s_1,s_2)-d^g(s_1,s_2)|$. By the triangle inequality, we have

$$g_r \geq \frac{1}{\binom{2^\ell}{2}}\sum_{(s_1,s_2)\in S_\ell}\left(d^p(s_1,s_2)-d^g(s_1,s_2)\right)$$

$$= \frac{1}{\binom{2^\ell}{2}}\left(\sum_{(s_1,s_2)\in S_\ell}d^p(s_1,s_2)-\sum_{(s_1,s_2)\in S_\ell}d^g(s_1,s_2)\right)$$

$$= \frac{1}{\binom{2^\ell}{2}}(P-G),$$

where we let $P=\sum_{(s_1,s_2)\in S_\ell}d^p(s_1,s_2)$ and $G=\sum_{(s_1,s_2)\in S_\ell}d^g(s_1,s_2)$. Since P only deals with phenotypes in $\mathbb{N}$, it is equivalent to

$$P = \sum_{i=0}^{2^\ell-1}\sum_{j=i+1}^{2^\ell-1}(j-i).$$

Let the outer sum fix i. The inner sum computes the sum of numbers from 1 to $2^\ell-1-i$. This reduces P to

$$= \sum_{i=1}^{n}i+\sum_{i=1}^{n-1}i+\cdots+\sum_{i=1}^{1}i.$$

where we let $n = 2^\ell - 1$ for simplicity. Using the facts that the sum of the first m natural numbers is $\frac{1}{2}m(m+1)$ and the sum of the first m squares is $\frac{1}{6}m(m+1)(2m+1)$, we have

$$= \frac{1}{2}\sum_{i=1}^{n} i(i+1)$$

$$= \frac{1}{2}\left(\sum_{i=1}^{n} i^2 + \sum_{i=1}^{n} i\right)$$

$$= \frac{1}{2}\left(\frac{1}{6}n(n+1)(2n+1) + \frac{1}{2}n(n+1)\right)$$

$$= \frac{1}{6}n(n+1)(n+2).$$

Substituting $n = 2^\ell - 1$ back into the equation yields:

$$= \frac{1}{6}(2^\ell - 1)(2^\ell)(2^\ell + 1).$$

Now, since G is the sum of the Hamming distances between all unique pairs of bitstrings, it is equivalent to

$$G = \frac{1}{2}2^\ell \sum_{i=1}^{\ell} i\binom{\ell}{i},$$

because for each of the 2^ℓ bitstrings, a bitstring has $\binom{\ell}{i}$ other bitstrings with Hamming distance i (choose i of the ℓ bits to be flipped). We divide by two because we count each pair twice. Simplifying G gives us

$$= 2^{\ell-1}\sum_{i=1}^{\ell} i \cdot \frac{\ell}{i}\binom{\ell-1}{i-1}$$

$$= \ell 2^{\ell-1}\sum_{i=1}^{\ell}\binom{\ell-1}{i-1}$$

$$= \ell 2^{\ell-1}2^{\ell-1}$$

$$= \ell 2^{2(\ell-1)}.$$

Substituting P and G back into g_r yields

$$g_r \geq \frac{1}{\binom{2^\ell}{2}}(P - G)$$

$$= \frac{1}{\binom{2^\ell}{2}}\left(\frac{1}{6}(2^\ell - 1)(2^\ell)(2^\ell + 1) - \ell 2^{2(\ell-1)}\right).$$

□

Asymptotic value of general locality

Here, we prove that the asymptotic value of general locality in our domain is invariant of the actual representation.

Theorem 12 $g_r \sim \frac{1}{\binom{2^\ell}{2}}(\frac{1}{6}(2^\ell - 1)(2^\ell)(2^\ell + 1) - \ell 2^{2(\ell-1)})$ *for any nonredundant binary-integer representation r on ℓ bits. That is, as ℓ grows, the value of g_r is independent of the actual representation.*

Proof The key intuition behind this proof is that for nonredundant binary-integer representations, as ℓ grows, the phenotypic distances grow at an asymptotically greater rate than the genotypic distances. We can separate the phenotypic distances from the genotypic distances by partitioning S_ℓ into two sets $S_\ell = S_\ell^p \sqcup S_\ell^g$, where $\sqcup$ denotes disjoint union:

$$S_\ell^p = \{(s_1, s_2) \in S_\ell \,|\, d^p(s_1, s_2) > d^g(s_1, s_2)\},$$
$$S_\ell^g = \{(s_1, s_2) \in S_\ell \,|\, d^p(s_1, s_2) \leq d^g(s_1, s_2)\}.$$

In other words, S_ℓ^p contains all the pairs in S_ℓ where the two bitstrings have greater phenotypic (Euclidean) distance than genotypic (Hamming) distance, and S_ℓ^g contains all pairs where the two bitstrings have greater or equal genotypic distance than phenotypic distance. We can rewrite g_r as (letting $C = 1/\binom{2^\ell}{2}$)

$$g_r = \frac{1}{\binom{2^\ell}{2}} \sum_{(s_1, s_2) \in S_\ell} |d^p(s_1, s_2) - d^g(s_1, s_2)|$$
$$= C(P(\ell) + G(\ell)),$$

where we let

$$P(\ell) = \sum_{(s_1, s_2) \in S_\ell^p} (d^p(s_1, s_2) - d^g(s_1, s_2)), \tag{7}$$

$$G(\ell) = \sum_{(s_1, s_2) \in S_\ell^g} (d^g(s_1, s_2) - d^p(s_1, s_2)). \tag{8}$$

Note that $|S_\ell^g| \leq \frac{1}{2}(2\ell - 1)2^\ell$ since each of the 2^ℓ bitstrings can have at most $2\ell - 1$ bitstrings for which their genotypic distance is greater or equal to their phenotypic distance. This is because for an integer $i \geq \ell$, the integers between $i - \ell$ and $i + \ell$ can be represented by bitstrings with genotypic distances greater than phenotypic distances $j \leq \ell$, where we subtract 1 due to the fact that each bitstring only has one other bitstring with Hamming distance ℓ. We then divide by two since we count each pair twice. Thus $|S_\ell^g| = \mathcal{O}(\ell 2^\ell)$. We can now say $|S_\ell^p| = |S_\ell| - |S_\ell^g| \geq \binom{2^\ell}{2} - \frac{1}{2}(2\ell - 1)2^\ell = \frac{2^\ell(2^\ell - 2\ell)}{2}$, so $|S_\ell^p| = \Omega(2^{2\ell})$. Consider

$$\lim_{\ell \to \infty} \frac{|S_\ell|}{|S_\ell^p|} = \lim_{\ell \to \infty} \frac{|S_\ell^p| + |S_\ell^g|}{|S_\ell^p|}$$
$$= \lim_{\ell \to \infty} \left(1 + \frac{|S_\ell^g|}{|S_\ell^p|}\right)$$
$$= 1 + 0 = 1,$$

since $|S_\ell^p| = \Omega(2^{2\ell})$ grows faster than $|S_\ell^g| = \mathcal{O}(\ell 2^\ell)$. Thus $|S_\ell^p|$ dominates $|S_\ell^g|$, and $|S_\ell| \sim |S_\ell^p|$. We can now perform a similar analysis for $P(\ell)$ and $P(\ell) + G(\ell)$. Note that $G(\ell) \leq |S_\ell^g|(\ell-1) \leq \frac{1}{2}\ell^2 2^\ell - \frac{1}{2}\ell 2^\ell$ since any pair in S_ℓ^g can have a maximum $d^g - d^p$ of $\ell - 1$. Thus $G(\ell) = \mathcal{O}(\ell^2 2^\ell)$. Also note that $P(\ell) \geq |S_\ell^p| \geq \frac{2^\ell(2^\ell - 2\ell)}{2}$ since any pair in S_ℓ^p can have a minimum $d^p - d^g$ of 1. Thus $P(\ell) = \Omega(2^{2\ell})$. Consider

$$\lim_{\ell \to \infty} \frac{P(\ell) + G(\ell)}{P(\ell)} = \lim_{\ell \to \infty} 1 + \frac{G(\ell)}{P(\ell)}$$
$$= 1 + 0 = 1,$$

since $P(\ell) = \Omega(2^{2\ell})$ grows faster than $G(\ell) = \mathcal{O}(\ell^2 2^\ell)$, and so $P(\ell) + G(\ell) \sim P(\ell)$. Now we can make a statement about g_r. We have

$$\frac{1}{C}g_r = P(\ell) + G(\ell)$$
$$\lim_{\ell \to \infty} \frac{g_r}{CP(\ell)} = \lim_{\ell \to \infty} \frac{P(\ell) + G(\ell)}{P(\ell)}$$
$$\lim_{\ell \to \infty} \frac{g_r}{CP(\ell)} = 1.$$

Thus $\frac{g_r}{C} \sim P(\ell)$. Since $|S_\ell^p| \sim |S_\ell|$ and $CP(\ell) \sim g_r$, we replace S_ℓ^p with S_ℓ in Eq. (7) to obtain (recall $C = 1/\binom{2^\ell}{2}$)

$$g_r \sim \frac{1}{\binom{2^\ell}{2}} \sum_{(s_1,s_2) \in S_\ell} \left(d^p(s_1, s_2) - d^g(s_1, s_2) \right)$$
$$= \frac{1}{\binom{2^\ell}{2}} \left(\frac{1}{6}(2^\ell - 1)(2^\ell)(2^\ell + 1) - \ell 2^{2(\ell-1)} \right),$$

which was found in the proof of Theorem 11. Thus g_r and $\frac{1}{\binom{2^\ell}{2}}(\frac{1}{6}(2^\ell - 1)(2^\ell)(2^\ell + 1) - \ell 2^{2(\ell-1)})$ are asymptotically equal for any representation r.

Theorem 12 states that all representations have asymptotically equal general locality. Since SB and BRG are well-defined for any number of bits, it follows that their general localities are asymptotically equal as well. In fact, on just their 11-bit representations (with only $n_p = 2,048$ possible genotypes, where a brute-force search likely outperforms many GEAs), we calculate $g_{SB} = 677.497$ and $g_{BRG} = 677.502$, only a 0.0007% difference. The implication is therefore that general locality, and by extension, the original definition of distance distortion, are not useful as a description or prediction of any representation in this domain. In a domain where phenotypic and genotypic distances grow with ℓ at similar rates, such as with unary representation, these definitions may have more power.

EXPERIMENTAL RESULTS

As the previous section proved, Gray encoding exhibits no general or point locality advantage over standard binary. On the other hand, several studies found a performance

advantage for Gray under various GEAs (*Whitley, Rana & Heckendorn, 1997*; *Whitley & Rana, 1997*; *Whitley, 1999*; *Mathias & Whitley, 1994b*). As a first step towards understanding why stronger locality does not always lead to a GEA performance advantage, this section expands on past experimental results and analyzes the factors that lead to better performance. Our experiments progress from the simple to the complex, to allow a tractable analysis of the effect of point locality, as well as an evaluation of both localities in richer and more realistic scenarios. All of our source code, representations, choices of parameters, and Markov models can be found in the supplementary material and at https://github.com/shastrihm/GAMO-R.

Simulated annealing

One very simple GEA is single-organism simulated annealing (SA), which we apply to the gen-one-max problem from Sec. 5.4 of Rothlauf's book (*Rothlauf, 2006*).[3] We start with SA, rather than the somewhat simpler (1+1) EA algorithm that we explore in the next section, because we can replicate Rothlauf's results exactly, and then expand upon them, giving us a solid basis for comparison.

To recap Rothlauf's experiment, it defines an ℓ-bit genotype x that is translated to an integer phenotype $x^p \in [0 : 2^\ell - 1]$ by a given representation. The fitness of the phenotype includes a deceptive trap, and is evaluated against a target a with the function

$$f_p(x^p) = x_{max} - |x^p - a|, \quad \text{(ibid. Eq. 5.2)}$$

where x_{max} is defined as the maximum phenotypic value, $2^\ell - 1$, and is attained only at the global maximum a. The genotype is iteratively mutated with a random single-bit flip, and the offspring replaces the parent if it improves its fitness, or at a probability determined by a Boltzmann process. This probability decreases both with the fitness difference between the parent and offspring, and with a global temperature parameter that decreases every iteration.

As in the original study, we set $\ell = 5$, the initial temperature to 50, and the cooling factor to 0.995, and experiment with different representations and a values. Each experiment is run for a few thousand generations (mutations) until it converges on a solution. Finally, we repeat each experiment concurrently and independently for thousands of different random starting genotypes and record for each generation the percentage of experiments (genotypes) that are at the global optimum.[4]

Rothlauf compared the GEA's performance across two representations, SB and BRG, using either $a = 31$ (where both representations perform equally) and $a = 15$, where SB often gets trapped on the local maximum phenotype 16 and thus significantly underperforms BRG. Our code produced the same results, as shown in Fig. 4.

In the original study, Rothlauf summarizes the worse performance of SB as follows: "The binary encoding has problems associated with the Hamming cliff and has low locality. Due its low locality, small changes of the genotype do not always result in small changes of the corresponding phenotype" (p. 136). However, as the previous section showed, BRG has no locality advantage over SB by either metric. In particular, the locality metric that captures the mean degree of variation across single-bit mutations, namely point locality, is

[3]In Rothlauf's book, this problem is called generalized-one-max or *gen-one-max*, but is not in fact related to the linear ONEMAX problem (*Droste, Jansen & Wegener, 2002*). Nevertheless, we keep this nomenclature to remain consistent with the work we are comparing against.

[4]The book concatenates ten 5-bit genotypes to one bitstring and sums up their individual fitness values. To simplify the analysis, we use a single 5-bit organism at a time. Our code allows an exact reproduction of Rothlauf's result if concatenation is selected.
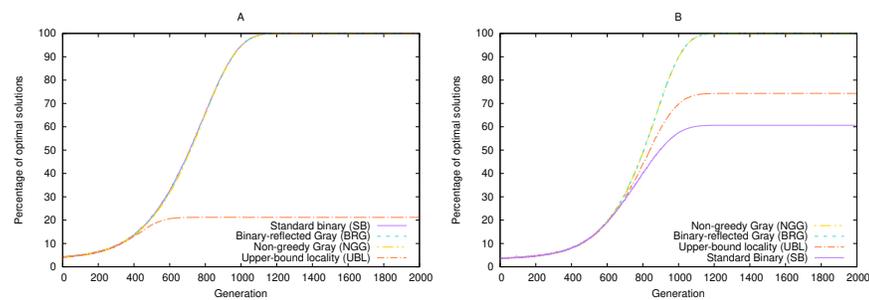
Figure 4 **Performance of four representations under simulated annealing for a 5-bit gen-one-max problem averaged over 100,000 random seeds with (A) $a = 31$ and (B) $a = 15$.** Higher results are better.

equal for both: $p_{SB} = p_{BRG} = \frac{2^\ell - 1}{\ell} = 6.2$. Weak locality cannot therefore be an explanation for performance differences in this case.

The other explanation about the Hamming cliff requires some unpacking. A Hamming cliff, which we loosely define as a genotype whose immediate genotypic neighbors are not also phenotypic neighbors, can represent a local phenotype maximum (*Rowe & Hidović, 2004*). That is because any genotype other than the global maximum, whose immediate genotypic neighbors (i.e., one bit-flip away) are not phenotypic neighbors, will have nonlocal jumps in values of the fitness function. These jumps could therefore all be farther away from the global maximum's fitness. Such genotypes are disfavored by the selection process to mutate, because any mutation lowers the fitness value. With a certain combination of temperature, cooling factor, and fitness parameters, the Boltzmann selection operator can develop a nonzero probability of getting stuck in the suboptimal genotype, leading to underperformance of the GEA. This is exactly what happens in the example of $a = 15$, where BRG has only one maximum at 15, but SB also has a suboptimal local maximum at 16, where no single-bit mutation improves fitness. Over time, SB has ≈ 0.4 probability of getting stuck in this local optimum, leading to $\approx 60\%$ optimal solutions over all random seeds.

Gray encoding, on the other hand, is guaranteed not to have more than one optimum, because every non-optimal phenotype has at least one genotypic neighbor—one bit flip away—that is also a phenotypic neighbor—one integer away—in the right direction. Whenever such a mutation happens, it is selected by the Boltzmann process, converging eventually to the single optimum. This property is invariant of the point locality value. To demonstrate this, we synthesized a Gray representation based on Claim 9 and termed it non-greedy Gray (NGG).[5] By construction, for $\ell = 5$, $p_{NGG} = 8.35 > p_{SB}$. Nevertheless, NGG performs perfectly for both target values, as Fig. 4 shows.

To take this point to the extreme, we synthesized a worst-case representation using the construction in Claim 4 and termed it upper-bound locality (UBL).[6] By construction, $p_{UBL} = 2^{5-1} = 16 > p_{SB}$. And although UBL performs poorly for $a = 31$ and suboptimally for $a = 15$, it still outperforms SB in the latter case.

The two point-locality extremes, SB and UBL, outperform each other for different target values. Even among representations with the same locality, performance is uneven.

[5] NGG = [0, 1, 19, 2, 31, 28, 20, 3, 23, 26, 24, 25, 22, 27, 21, 4, 13, 14, 18, 15, 30, 29, 17, 16, 12, 9, 11, 10, 7, 8, 6, 5].

[6] UBL = [24, 1, 4, 19, 15, 16, 21, 13, 9, 26, 18, 0, 23, 12, 6, 22, 3, 28, 20, 14, 30, 7, 5, 27, 29, 10, 8, 31, 2, 17, 25, 11].

We must therefore conclude again that locality does not determine performance for this particular problem. The second explanation, the existence of Hamming cliffs, is also insufficient to predict performance (*Chakraborty & Janikow, 2003*).

Gray coding is sometimes described as higher-performing than SB because it can produce fewer local maxima for many useful problems (*Whitley, Rana & Heckendorn, 1997*). But the local maxima count on its own is not a strong predictor of GEA performance for this problem, because the probability of getting stuck in any specific maximum depends on its fitness distance from its genotypic neighbors. For example, UBL has four Hamming cliffs (local maxima) with $a = 15$, compared to SB's two, and yet UBL still outperforms SB. In another example with $a = 29$, SB has three local maxima, and yet it still converges to the global optimum every single time.

In summary, neither locality nor local maxima count can predict performance for the gen-one-max problem under Boltzmann selection. This holds true for many other functions as well (*Chakraborty & Janikow, 2003*). A more nuanced explanation of the GEA performance arises from two observations: that the linear fitness function penalizes local maxima that are farther from the global maximum (in phenotypic distance), and that the Boltzmann selection process gives preference to mutations that minimize phenotypic distance.

The interaction between these two factors can be modeled with a simple Markov chain, which fully predicts the probability of converging to the local maximum in the absence of a cooling factor. Adding the cooling factor complicates the model, but it can still be computed with a dynamic probability function. The main constraint this GEA imposes on some representations is the inability to break free from some local maxima. In the next section we turn to a different GEA, so that we may evaluate the effect of point locality with different mutation and selection operators that can always jump out of a local maximum.

Evolutionary algorithms

Our next experiment introduces two changes to the previous GEA. First, we use simple elitist selection instead of Boltzmann selection. An offspring only replaces a parent if it improves its fitness. This change would make local maxima impassable for single-bit mutation, so we change the mutation operator as well. Mutations can now flip any bit independently with probability m, so that any genotype can be mutated to any other genotype at some positive probability that depends on m and the Hamming distance between them. Therefore any genotype could in principle be mutated to the global maximum in a single step. From a locality point of view, invoking multiple bit flips per generation has the same effect as invoking a single random bit flip for multiple generations, so a representation with strong point locality under a single bit flip should also exhibit a smaller mean change in phenotype value in this experiment.

This modified GEA is denoted as (1+1) EA (*Droste, Jansen & Wegener, 2002*), has been studied extensively, and may be considered as a degenerate kind of simulated annealing. *Mühlenbein (1992)* investigated this GEA in the context of the simpler ONEMAX problem by developing an analytical model for the expected number of generations to reach the optimum solution. Using a simpler fitness function that merely counts the number of '1'

Table 1 Mean generations to reach optimum in a 5-bit (1+1) EA gen-one-max evaluation with a mutation rate of 0.2, averaged across 100,000 runs of up to 1,000 generations each. Runs that did not reach the optimal solution were excluded from the mean. Lower results are better.

Optimum (a)	SB	BRG	NGG	UBL
31	17.5	23.8	28.3	184.8
15	103.6	21.3	23.3	124.8

bits in the genotype, his model predicts the mean number of generations to converge on the optimum when the mutation rate is $m = \frac{1}{\ell}$. His prediction holds for any representation since the fitness function ignores the phenotype. Indeed, when we ran the same simulation on various ℓ values larger than 5, we get similar empirical convergence times as Mühlenheim's experimental results, irrespective of the representation.

Reintroducing Rothlauf's gen-one-max fitness function adds back a dependency on the representation, as Table 1 shows. Modeling this performance with a Markov chain confirms the empirical results and the differences between the representations. This is an example where SB actually outperforms both Gray encodings, at least for $a = 31$. In this case, note that a mutation improves fitness if and only if it flips a bit to '1'. Improving mutations, which are the only ones that can change the genotype, are independent of each other and of any sequence, so the order of '0' to '1' bit flips does not matter. This leads to swift convergence to the optimum. Conversely, in all other representations, some sequences of fitness improvement require first flipping a specific bit one way, then later the other way. It imposes an ordering or interdependency of mutations that prolongs convergence. It is especially bad for UBL, because many fitness improvements require more than one bit flip at the same time, which occurs less frequently.

When we switch to $a = 15$, we introduce the same handicap to SB, because now it has to flip every single bit at once when trying to escape the phenotype 16, at a low probability of m^ℓ. Despite the dismal performance, it is worth noting that SB still performs better under (1+1) EA than under simulated annealing (SA): in 2,000 generations, it reaches the optimal solution in 65.9% of the runs, compared to SA's 60.6%, and if allowed to evolve for more generations, all runs will eventually find the optimum.

It is also worth noting that NGG slightly underperforms BRG for both target values, which could be a related to its weaker locality. But overall, locality is a poor explanation for (1+1) EA performance in this experiment, because SB outperforms BRG with $a = 31$ and underperforms it with $a = 15$, despite both representations having the exact same point locality.

Genetic algorithms

[7] Tournament selection yielded substantively identical results.

For our last experiment, we add three more layers of complexity. First, we add a population element with fitness-proportionate selection (roulette-wheel).[7] Second, we add a recombination operator, specifically single-point crossover. And finally, we expand our evaluation functions to De Jong's test suite, which can be more challenging to solve with a GEA than gen-one-max (*De Jong, 1975*). These five functions take real numbers as

Table 2 GA online performance for four representations averaged over 3,000 runs. *Caruana & Schaffer (1988)* results averaged over five runs in parentheses. Lower values denote better performance.

Function	Description	Dimensions	Total bits	Optimum	SB	BRG	NGG	UBL
f1	Parabola	3	30	0	2.49 (1.66)	2.03 (1.43)	2.03	6.70
f2	Rosenbrock's Saddle	2	24	0	32.56 (25.16)	22.31 (13.18)	22.35	199.48
f3	Step function	5	50	−30	−26.48 (−27.78)	−25.10 (−27.16)	−25.06	−17.86
f4	Quadratic with noise	30	240	0	41.46 (24.28)	32.32 (21.77)	32.28	65.83
f5	Shekel's foxholes	2	34	≈1	56.71 (30.78)	35.86 (20.68)	35.84	149.92

inputs, but De Jong and subsequent studies used fixed-point numbers as inputs, which lie in the same binary-to-integer scope as the rest of this paper.

To compare different representations under a genetic algorithm (GA), we attempted to replicate the experiments of *Caruana & Schaffer (1988)*, which in turn derive their GA parameters from *Grefenstette (1986)*. We measured the online performance under SB, BRG, NGG, and UBL representations, where online performance is defined as the average fitness of all function evaluations to the current point. Their results and ours are shown in Table 2.

Unfortunately, we could not uncover the full details or source code of their implementation, and had to construct our own experiment from scratch, likely with different parameter or implementation choices. These differences could help explain why our results are not identical to the original study's. Another explanation could be that the original study averaged each experiment over only five trials, which we found too noisy. Instead, we averaged performance over 3,000 trials (90,000 fitness evaluations) for each experiment to increase statistical robustness.

While our results differ in quantity, they agree in quality. Caruana and Schaffer found BRG to perform similar to or better than SB on the minimization of all five test functions, as have we and others (*Hinterding, Gielewski & Peachey, 1995*). While Caruana and Schaffer recognize that an encoding is as likely to outperform another encoding on an arbitrary search, they believe that "many common functions which have ordered domains have local correlations between domain and range. A GA using Gray coding will often perform better on this class of functions than one using binary coding because the Gray coding preserves this structure better than binary code." The authors suggest that it is the introduction of Hamming cliffs that biases SB away from preserving the domain structure. The fact that the weaker-locality NGG performs so closely to BRG supports this explanation.

Their paper does not clarify whether and how the crossover mechanism interacts with the representation. For example, there is evidence that Gray encoding interferes with standard crossover operators because it disrupts common schemata (*Weicker, 2010*), and that if crossover is the only variation operator, SB can actually outperform BRG on gen-one-max (*Rothlauf, 2002*).

What *is* clear from our experiment is that these behaviors are not captured by the locality metrics. BRG outperforms SB despite having the same point locality, and is equal in performance to NGG, with the weaker locality. Additionally, all four representations

share the same values for the distance distortion, general locality, or remoteness preservation metrics.

To summarize all three experiments, none of our results shows that strong locality leads necessarily to better GEA performance. The next section suggests some reasons for this counterintuitive result.

DISCUSSION

Locality and GEA performance

Why should strong locality improve GEA performance? We can list two reasons. First, strong locality helps preserve building blocks (schemata) across variation operators, because it supports the linkage between genotypic and phenotypic building blocks (*Vose, 1991*). Second, specifically for mutation operators, strong locality enables localized, hill-climbing, or gradient search, by effecting small changes to the genotype, such as single-bit mutation. With strong locality, small genotypic changes lead to small phenotypic changes, and because many practical fitness functions are locally continuous, they also lead to small fitness changes. In Rothlauf's words (*Rothlauf, 2003*):

> ...low-locality representations randomize the search process and make problems that are easy for mutation-based search more difficult and difficult problems more easy. Although low-locality representations increase the performance of local search on difficult, deceptive problems this is not relevant for real-world problems as we assume that most problems in the real-world are easy for mutation-based search.

Despite these reasons, we found no strong association between various locality measures and GEA performance in our work and others'. One possible explanation is that the common test functions we evaluated are better-suited for Gray encoding, perhaps because of their sensitivity to Hamming cliffs (*Caruana & Schaffer, 1988*; *Whitley, 1999*). Another explanation is that in the domain we study here, nonredundant binary-integer mappings, our locality and distance metrics are skewed: phenotypic distances grow exponentially with ℓ, while genotypic (Hamming) distances grow linearly. For example, if we instead use the redundant unary bitstring-to-integer representation, for example, then phenotypic distances also grow linearly with genotypic distances, but the GEA would typically underperform nonredundant representations (*Rothlauf, 2006*). As another example, in the separate domain of program trees, some definitions of locality have been found to correlate well with GEA performance (*Galván-López et al., 2011*).

Regardless, even if in our domain locality does not affect eventual performance, we think it has a role in predicting GEA performance over time, especially in regards to the mutation operator. The role of the mutation operator in GEAs is to introduce genotypic and phenotypic variation (*Hinterding, Gielewski & Peachey, 1995*). In other words, mutations serve to diversify the population so that more potential solutions are evaluated. Locality, as defined here, quantifies the average impact of single mutations, with strong locality leading to smaller overall phenotypical variation. Whether or not strong locality improves GEA performance depends on the effect of diversity on performance, which itself varies over time.

Phenotypic diversity over time

In the process of converging towards an optimal solution, GEAs combine and balance two different search strategies, exploration and exploitation (*Črepinšek, Liu & Mernik, 2013*; *Weicker, 2010*). In the former, high variance or diversity is desired so that many subspaces of the fitness landscape are explored. When the GEA identifies a promising subspace, large variations in phenotype are more likely to be disruptive rather than helpful to fitness, so small perturbations are usually preferred (*Rowe et al., 2004*; *Mathias & Whitley, 1994b*). If the only variance-introducing operator in a GEA is single-bit mutation, then we should expect strong-locality representations to underperform weak-locality representations in the early phase of the search, and outperform in the latter phase. Because of this changing relationship over time, several studies explored dynamically changing the mutation rate and other GEA parameters (*Doerr & Doerr, 2019*).

This difference matters most if our GEA is meaningfully constrained by computational resources (*Jansen, 2020*). The GEA's task then is to quickly explore the solution space for a "good enough" solution, rather than an eventual convergence towards an optimum. If a GEA is configured such that the expected fitness is relatively high after a few generations, and before convergence, then it has a higher probability of finding a decent solution with fewer resources, even if it is slower to converge later. This distinction is similar to the one between online performance and offline performance of the GEA (*Grefenstette, 1986*).

As an example, consider a comparison of performance over time between a strongest-possible locality representation and a weakest-possible locality representation in the the simulated annealing experiment ('Simulated annealing'). We picked two arbitrary target *a* values for SB and UBL such that both representations have the same number of local maxima, four. Because we are interested in the likelihood of getting a "good enough" solution in limited time, rather than an optimal solution, we measure the expected (mean) fitness per generation across 100,000 runs, instead of the percentage of perfect solutions.

Figure 5 shows that for long-enough runs (starting from about 400 generations), both representations perform about the same, with a small eventual advantage to SB. But in the earlier generations, during the exploration phase, SB significantly underperforms UBL. In other words, this example shows that if we must stop our simulation at an early point before convergence, the representation with the weaker locality is more likely to produce a solution of higher fitness. But after that threshold has been passed, the stronger-locality representation may be more suitable for fine-grained exploitation to locate a higher-fitness solution. On the other hand, sometimes we are specifically interested in local or hill-climbing search (*Mitchell, Holland & Forrest, 1994*), in which case Gray encoding may be preferable to SB, even with the same locality, because it always offers a genotypic neighbor that can move the phenotype towards the maximum (*Rowe et al., 2004*).

When we repeat the experiment with (1+1) EA, we see the same effect, where UBL starts out stronger but is then quickly surpassed by SB. In this case, both representations end up with the same eventual performance. Even the GA experiments show the same locality effect (Fig. 6), where UBL always starts out stronger than the other representations, even if only for a very short time.
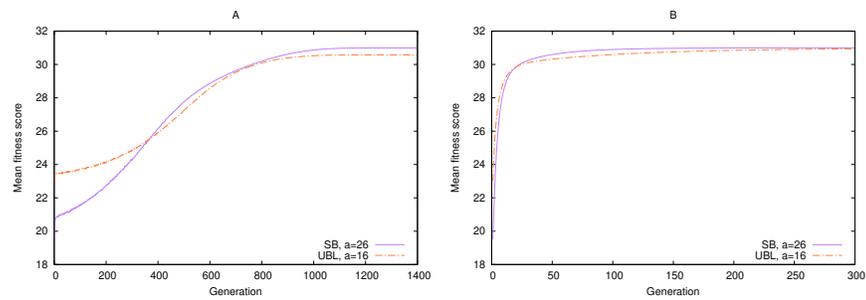
Figure 5 **Convergence speed for 5-bit gen-one-max problems with four local maxima each and either (A) Simulated Annealing or (B) (1 + 1) EA.** Fitness is averaged over 100,000 trials.
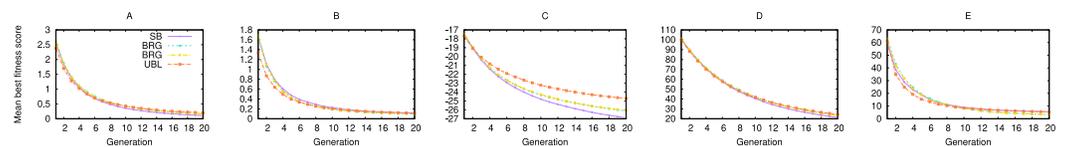
Figure 6 **Performance of four representations for the first 20 generations of a genetic algorithm.** Each data point averages fitness values from the best solution in that generation across 3,000 trials. Functions shown are (A) f1; (B) f2; (C) f3; (D) f4; and (D) f5.

This change in the role of mutations over time is also why some GEAs vary the mutation rate or operator over the course of the search (*Eiben & Schippers, 1998*; *Gómez, Dasgupta & González, 2003*; *Hansen & Ostermeier, 2001*; *Zhao, Gao & Hu, 2007*). An alternative approach would start the search with a weak-locality representation, and switch to a strong-locality representation as we move from the exploration phase to the exploitation phase.

Some consider the recombination operator as the main exploration tool, with mutation as the main exploitation tool (*Caruana & Schaffer, 1988*), while others disagree (*Hinterding, Gielewski & Peachey, 1995*). A recent result showed that crossover and mutation outperform any mutation-only algorithm for exploitation, at least for the hill-climbing function OneMax (*Corus & Oliveto, 2020*). Steady-state GEAs, unlike the generational GEAs considered here, allow for extremely low selective pressure independent from the mutation rate, which lowers the sensitivity to mutation operators and enables more effective exploration (*Corus & Oliveto, 2017*; *Corus et al., 2021*).

We think a more nuanced discussion needs to include the locality of the representation under the chosen mutation operator. Our results suggest that strong-locality representations are more suitable for an exploitation-oriented mutation operator, since they limit the perturbation of the phenotype, and vice-versa.

Changing representation randomly during the GEA execution may not be a practical tool to improve performance, because the expected number of local optima is too high (*Whitley, Rana & Heckendorn, 1997*). As we have shown in 'Expected point locality', the expected locality is also too high, closer to the pessimal locality than to the optimal locality,

on average. But if our goal for the exploration phase of the GEA is to generate significant genotypic diversity and coverage of the search subspaces, then shifting to and between random representations may help this goal. To the best of our knowledge, no one has tried this approach, and it remains as promising future work for analysis.

CONCLUSION AND FUTURE WORK

Various properties of GEA representations have been studied to help explain GEA performance, such as the number of induced local optima or the existence of Hamming cliffs. Among these properties, locality is particularly interesting to analyze, because it is independent of the fitness landscape and can be computed precisely, which is not always the case for other properties.

But if strong locality is also a strong factor in good GEA performance, as has been expressed in the literature, then we do not always have the right metrics to measure it. Our own point and global locality metrics, based on the works of Rothlauf and others, show no clear relationship to offline GEA performance in the domain of nonredundant binary-integer representations. It is quite likely that the result that locality does not predict GEA performance extends to more domains. Conversely, in some other domains, such as redundant representations or those with linear scaling of the phenotype, this negative finding may not hold. We plan to explore locality metrics specific to redundant, nonbinary, or noninteger representations as well. This result also does not always hold for resource-constrained GEAs, since for short-enough executions, weak-locality representations do appear to outperform strong-locality representations in our experiments, as they emphasize exploration over exploitation.

Both the d_m and d_c locality metrics (and consequently, p_r and g_r) estimate the locality of a representation by using the sum of phenotypic distances, which grows exponentially in our domain. Another path for future research might be to use other statistics as estimators of locality that grow more slowly with the bitstring length, such as the minimum phenotypic distance, maximum, or standard deviation. In addition, both Rothlauf's locality metric and our own equivalent point locality focus on the single-bit mutation operator. Other locality metrics could look at different operators, and perhaps combine them with a crossover operator to expand on distance distortion.

Of the two most common binary-integer encodings, Gray has often been shown to outperform standard binary on many GEAs and test functions. There are various context-dependent explanations for this advantage, such as the Hamming cliff, differing numbers of local optima, and the properties of the typical test functions. What our study has shown, both analytically and empirically, is that the locality of the representations cannot reliably be one of these explanations. A complete characterization and understanding of the performance difference between binary and Gray remains an interesting research question, and a different locality metric may be able to shed more light.

ACKNOWLEDGEMENTS

We wish to thank Prof. Dr. Franz Rothlauf for fruitful discussions at the beginning of this project.

Funding

The authors received no funding for this work.

Competing Interests

The authors declare there are no competing interests.

Author Contributions

- Hrishee Shastri performed the experiments, analyzed the data, performed the computation work, prepared figures and/or tables, authored or reviewed drafts of the paper, and approved the final draft.
- Eitan Frachtenberg conceived and designed the experiments, performed the experiments, analyzed the data, performed the computation work, prepared figures and/or tables, authored or reviewed drafts of the paper, and approved the final draft.

Data Availability

The complete code for all of the experimental results is available as Supplemental Material and at GitHub: https://github.com/shastrihm/GAMO-R.

Supplemental Information

Supplemental information for this article can be found online at http://dx.doi.org/10.7717/peerj-cs.561#supplemental-information.

REFERENCES

Bäck T, Fogel DB, Michalewicz Z. 2018. *Evolutionary computation 1: basic algorithms and operators.* Boca Raton: CRC Press.

Barbulescu L, Watson J-P, Whitley LD. 2000. Dynamic representations and escaping local optima: improving genetic algorithms and local search. In: *The proceedings of the 17th National Conference on Artificial Intelligence and 12th conference on innovative applications of artificial intelligence.* Palo Alto: AAAI Press, 879–884.

Bitner JR, Ehrlich G, Reingold EM. 1976. Efficient generation of the binary reflected Gray code and its applications. *Communications of the ACM* **19(9)**:517–521.

Caruana RA, Schaffer JD. 1988. Representation and hidden bias: Ggray vs. binary coding for genetic algorithms. In: *Machine learning proceedings 1988.* Amsterdam: Elsevier, 153–161.

Chakraborty UK, Janikow CZ. 2003. An analysis of Gray versus binary encoding in genetic search. *Information Sciences* **156(3–4)**:253–269.

Chiam S, Goh C, Tan K. 2006. Issues of binary representation in evolutionary algorithms. In: *Proceedings of the IEEE conference on cybernetics and intelligent systems*. Piscataway: IEEE, 1–8.

Corus D, Lissovoi A, Oliveto PS, Witt C. 2021. On steady-state evolutionary algorithms and selective pressure: why inverse rank-based allocation of reproductive trials is best. ArXiv preprint. arXiv:2103.10394.

Corus D, Oliveto PS. 2017. Standard steady state genetic algorithms can hillclimb faster than mutation-only evolutionary algorithms. *IEEE Transactions on Evolutionary Computation* **22(5)**:720–732.

Corus D, Oliveto PS. 2020. On the benefits of populations for the exploitation speed of standard steady-state genetic algorithms. *Algorithmica* **82(12)**:3676–3706.

Črepinšek M, Liu S-H, Mernik M. 2013. Exploration and exploitation in evolutionary algorithms: a survey. *ACM Computing Surveys (CSUR)* **45(3)**:35.

De Jong KA. 1975. An analysis of the behavior of a class of genetic adaptive systems. PhD thesis, University of Michigan, Ann Arbor, MI. *Available at deepblue.lib.umich.edu/handle/2027.42/4507*.

Doerr B, Doerr C. 2019. Theory of parameter control for discrete black-box optimization: provable performance gains through dynamic parameter choices. In: *Theory of evolutionary computation*. Cham: Springer International Publishing, 271–321 DOI 10.1007/978-3-030-29414-4_6.

Doerr B, Klein C, Storch T. 2007. Faster evolutionary algorithms by superior graph representation. In: *Proceedings of the IEEE symposium on foundations of computational intelligence*. Piscataway: IEEE, 245–250 DOI 10.1109/FOCI.2007.372176.

Droste S, Jansen T, Wegener I. 2002. On the analysis of the $(1+1)$ evolutionary algorithm. *Theoretical Computer Science* **276(1–2)**:51–81.

Eiben AE, Schippers CA. 1998. On evolutionary exploration and exploitation. *Fundamenta Informaticae* **35(1–4)**:35–50.

Galván-López E, McDermott J, O'Neill M, Brabazon A. 2010. Towards an understanding of locality in genetic programming. In: *Proceedings of the 12th annual conference on genetic and evolutionary computation*. ACM, 901–908.

Galván-López E, McDermott J, ONeill M, Brabazon A. 2011. Defining locality as a problem difficulty measure in genetic programming. *Genetic Programming and Evolvable Machines* **12(4)**:365–401.

Galván-López E, O'Neill M. 2009. On the effects of locality in a permutation problem: the Sudoku puzzle. In: *Proceedings of the IEEE symposium on computational intelligence and games*. Piscataway: IEEE, 80–87.

Goldberg DE, Deb K, Clark JH. 1992. Genetic algorithms, noise, and the sizing of populations. *Complex Systems* **6**:333–333.

Gómez J, Dasgupta D, González F. 2003. Using adaptive operators in genetic search. In: *Genetic and evolutionary computation conference*. Cham: Springer, 1580–1581.

Gottlieb J, Julstrom BA, Raidl GR, Rothlauf F. 2001. PrüFer numbers: a poor representation of spanning trees for evolutionary search. In: *Proceedings of the 3rd annual*

conference on genetic and evolutionary computation, GECCO'01. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc, 343–350.

Gottlieb J, Raidl GR. 1999. Characterizing locality in decoder-based EAs for the multidimensional knapsack problem. In: *European conference on artificial evolution.* Springer, 38–52.

Grefenstette JJ. 1986. Optimization of control parameters for genetic algorithms. *IEEE Transactions on Systems, Man amd Cybernetics* **16**:122–128. *Available at longwood.cs. ucf.edu/~hyu/gref1986.mod.doc* DOI 10.1109/TSMC.1986.289288.

Hansen N, Ostermeier A. 2001. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation* **9(2)**:159–195.

Harper LH. 1964. Optimal Assignments of Numbers to Vertices. *Journal of the Society for Industrial and Applied Mathematics* **12(1)**:131–135.

Hinterding R, Gielewski H, Peachey TC. 1995. The Nature of Mutation in Genetic Algorithms. In: *Proceedings of the 6th international conference on genetic algorithms.* 65–72.

Hoai NX, McKay RI, Essam D. 2006. Representation and structural difficulty in genetic programming. *IEEE Transactions on Evolutionary Computation* **10(2)**:157–166.

Jansen T. 2020. Analysing stochastic search heuristics operating on a fixed budget. In: *Theory of evolutionary computation: recent developments in discrete optimization.* Cham: Springer International Publishing, 249–270 DOI 10.1007/978-3-030-29414-4_5.

Jansen T, Oliveto PS, Zarges C. 2013. Approximating vertex cover using edge-based representations. In: *Proceedings of the twelfth workshop on foundations of genetic algorithms XII, FOGA XII '13.* New York, NY, USA: Association for Computing Machinery, 87–96 DOI 10.1145/2460239.2460248.

Jones T, Forrest S. 1995. Fitness distance correlation as a measure of problem difficulty. In: *Proceedings of the 6th international conference on genetic algorithms.* 184–192.

Liepins GE, Vose MD. 1990. Representational issues in genetic optimization. *Journal of Experimental & Theoretical Artificial Intelligence* **2(2)**:101–115.

Manderick B. 1991. The genetic algorithm and the structure of fitness landscape. In: *Proceedings of the 4th international conference on genetic algorithms.* 143–150.

Mathias KE, Whitley LD. 1994a. Changing representations during search: a comparative study of delta coding. *Evolutionary Computation* **2(3)**:249–278.

Mathias KE, Whitley LD. 1994b. Transforming the search space with gray coding. In: *Proceedings of the first ieee conference on evolutionary computation.* Piscataway: IEEE, 513–518.

Mitchell M, Holland JH, Forrest S. 1994. When will a genetic algorithm outperform hill climbing. In: *Advances in neural information processing systems.* 51–58.

Mitchell-Olds T, Willis JH, Goldstein DB. 2007. Which evolutionary processes influence natural genetic variation for phenotypic traits? *Nature Reviews Genetics* **8(11)**:845.

Mueller-Bady R, Kappes M, Medina-Bulo I, Palomo-Lozano F. 2017. Optimization of monitoring in dynamic communication networks using a hybrid evolutionary

algorithm. In: *Proceedings of the genetic and evolutionary computation conference, GECCO'17*. New York, NY, USA: Association for Computing Machinery, 1200–1207 DOI 10.1145/3071178.3071255.

Mühlenbein H. 1992. How genetic algorithms really work: I. mutation and hillclimbing. In: *Proceedings of the 2nd international conference on parallel problem solving from nature*. Amsterdam: Elsevier.

Neumann F, Witt C. 2015. On the runtime of randomized local search and simple evolutionary algorithms for dynamic makespan scheduling. In: *Proceedings of the 24th international conference on artificial intelligence, IJCAI'15*. Burlington: AAAI Press, 3742–3748.

Ouch R, Reese K, Yampolskiy R. 2007. Hybrid genetic algorithm for the maximum clique problem combining sharing and migration. *MAICS* **1348**:79–84.

Pereira FB, Marques JM, Leitão T, Tavares J. 2006. Analysis of locality in hybrid evolutionary cluster optimization. In: *Proceedings of the IEEE international conference on evolutionary computation*. Piscataway: IEEE, 2285–2292.

Raidl GR, Gottlieb J. 2005. Empirical analysis of locality, heritability and heuristic bias in evolutionary algorithms: a case study for the multidimensional knapsack problem. *Evolutionary Computation* **13(4)**:441–475.

Rana SB, Whitley LD. 1997. Bit representations with a twist. In: *Proceedings of the 7th international conference on genetic algorithms (ICGA)*. East Lansing, MI: Morgan Kaufmann, 188–195.

Ronald S. 1997. Robust encodings in genetic algorithms: a survey of encoding issues. In: *Proceedings of IEEE international conference on evolutionary computation*. Piscataway: IEEE, 43–48.

Rothlauf F. 2002. Binary representations of integers and the performance of selectore-combinative genetic algorithms. In: *Proceedings of the international conference on parallel problem solving from nature*. Berlin, Heidelberg: Springer, 99–108.

Rothlauf F. 2003. On the locality of representations. In: *Proceedings of the international conference on genetic and evolutionary computation*. 1608–1609.

Rothlauf F. 2006. *Representations for genetic and evolutionary algorithms*. 2nd edition. Heidelberg: Springer.

Rothlauf F, Goldberg D. 1999. Tree network design with genetic algorithms–an investigation in the locality of the Pruefernumber encoding. In: *Late breaking papers at the genetic and evolutionary computation conference*. 238–244.

Rothlauf F, Oetzel M. 2006. On the locality of grammatical evolution. In: *European conference on genetic programming*. Berlin, Heidelberg: Springer, 320–330.

Rowe J, Whitley D, Barbulescu L, Watson J-P. 2004. Properties of Gray and binary representations. *Evolutionary computation* **12(1)**:47–76.

Rowe JE, Hidović D. 2004. An evolution strategy using a continuous version of the gray-code neighbourhood distribution. In: *Genetic and evolutionary computation conference*. Berlin, Heidelberg: Springer, 725–736.

Shastri H, Frachtenberg E. 2020. Locality bounds for nonredundant binary-integer representations. In: *2020 IEEE symposium on foundations of computational intelligence (FOCI)*. Piscataway: IEEE.

Skolicki Z, De Jong K. 2004. Improving evolutionary algorithms with multi-representation island models. In: *International conference on parallel problem solving from nature*. Springer, 420–429.

Vose MD. 1991. Generalizing the notion of schema in genetic algorithms. *Artificial Intelligence* **50(3)**:385–396.

Weicker K. 2010. A binary encoding supporting both mutation and recombination. In: *Proceedings of the international conference on parallel problem solving from nature*. Berlin, Heidelberg: Springer, 134–143.

Whitley D. 1999. A free lunch proof for gray versus binary encodings. In: *Proceedings of the 1st annual conference on genetic and evolutionary computation*. Burlington: Morgan Kaufmann Publishers Inc, 726–733.

Whitley D. 2000. Functions as permutations: regarding no free lunch, walsh analysis and summary statistics. In: *International conference on parallel problem solving from nature*. Paris, France: Springer, 169–178.

Whitley D, Rana S, Heckendorn R. 1997. Representation issues in neighborhood search and evolutionary algorithms. In: Quagliarelli D, Periaux J, Poloni C, Winter G, eds. *Genetic Algorithms and Evolution Strategy in Engineering and Computer Science*. 39–57.

Whitley D, Rana SB. 1997. Representation, search and genetic algorithms. In: *Proceedings of the 14th national conference on artificial intelligence and 9th conference on innovative applications of artificial intelligence*. 497–502.

Wolpert DH, Macready WG. 1997. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation* **1(1)**:67–82.

Zhao X, Gao X-S, Hu Z-C. 2007. Evolutionary programming based on non-uniform mutation. *Applied Mathematics and Computation* **192(1)**:1–11.