Locality, Distance Distortion, and Binary Representations of Integers

Franz Rothlauf

Working Paper 11/2003
July 2003

Working Papers in Information Systems 1

University of Mannheim
Department of Information Systems 1
D-68131 Mannheim/Germany
Phone +49 621 1811691, Fax +49 621 1811692
E-Mail: wifo1@uni-mannheim.de
Internet: `http://www.bwl.uni-mannheim.de/wifo1`

Locality, Distance Distortion, and Binary Representations of Integers

Franz Rothlauf
Dept. of Information Systems 1
University of Mannheim
D-68131 Mannheim/Germany
rothlauf@uni-mannheim.de

July 20, 2003

Abstract

This paper investigates how the performance of evolutionary search is influenced by the locality and distance distortion of the used representation. The locality of a representation describes how well neighboring phenotypes correspond to neighboring genotypes. The distance distortion measures whether the distances between individuals are preserved when mapping the phenotypes onto the genotypes. When using mutation as the main search operator, perfect locality ensures that the difficulty of the problem is not changed by the representation. In analogy, representations that preserve the distances between individuals guarantee that the difficulty of the problem remains unchanged for crossover-based search.

Applying these concepts to binary representations of integers reveals that in contrast to the binary encoding the Gray encoding has perfect locality, and therefore guarantees that for mutation-based search, easy problems remain easy. However, because both Gray and binary encoding change the distances between individuals, using these encodings also changes the difficulty of problems for crossover-based search and some easy problems become more difficult to solve by genetic algorithms.

1 Introduction

Using evolutionary algorithms – for example for integer optimization problems – raises the question as to which representation should be used. In the literature only a few theory-based recommendations can be found (for example Whitley (1999)) but a greater and better theoretical understanding on how representations influence the performance of evolutionary search is necessary.

This paper has two goals. Firstly, we want to describe how the performance of evolutionary search is influenced by the locality and distance distortion of the used representation. For our investigation we distinguish between mutation-based and crossover-based search. We illustrate that encodings with perfect locality, that means neighboring phenotypes correspond to neighboring genotypes, do not modify the difficulty of the optimization problem for mutation-based search. In analogy, the difficulty of problems remains unchanged for crossover-based search if representations are used that preserve the distances between corresponding phenotypes and genotypes. Secondly, we focus on binary representations of integers, namely binary and Gray encoding. Gray encoding has perfect locality and preserves problem difficulty for mutation-based search. Therefore, when using mutation as the main search operator easy problems remain easy. In contrast, when using

binary encoding which has low locality, some easy problems become more difficult to solve for mutation-based search.

The remainder of the paper is organized as follows. We start with a short introduction into representations. In subsection 2.2 we describe the locality and distance distortion of an encoding. After a short description of the used optimization problem and the binary and Gray encoding, in section 4 we present empirical results. The paper ends with concluding remarks.

2 Representations for Evolutionary Algorithms

In the following section we discuss the decomposition of a genotype-fitness mapping and describe the locality and distance distortion of a representation.

2.1 Genotype-Phenotype Mappings and Phenotype-Fitness Mappings

When using some kind of representation, every optimization problem can be decomposed into a genotype-phenotype mapping f_g, and a phenotype-fitness mapping f_p (Liepins & Vose, 1990).

We define Φ_g as the genotypic search space where the genetic operators such as recombination or mutation are applied to. An optimization problem on Φ_g could be formulated as follows: The function $f(\boldsymbol{x}) : \Phi_g \to \mathbb{R}$ assigns an element in $\mathbb{R}$ to every element in the genotype space Φ_g. The optimization problem is defined by finding the optimal solution

$$\hat{\boldsymbol{x}} = \max_{\boldsymbol{x} \in \Phi_g} f(\boldsymbol{x}),$$

where $\boldsymbol{x}$ is a vector of decision variables (or alleles), and $f(\boldsymbol{x})$ is the fitness function. The vector $\hat{\boldsymbol{x}}$ is the global maximum.

When using a representation we have to introduce phenotypes and genotypes. Thus, the fitness function f can be decomposed into two parts. The first maps the genotypic space Φ_g to the phenotypic space Φ_p, and the second maps Φ_p to the fitness space $\mathbb{R}$:

$$f_g(\boldsymbol{x}_g) : \Phi_g \to \Phi_p,$$
$$f_p(\boldsymbol{x}_p) : \Phi_p \to \mathbb{R},$$

where $f = f_p \circ f_g = f_p(f_g(\boldsymbol{x}_g))$. The genotype-phenotype mapping f_g is the used representation. f_p represents the fitness function and assigns a fitness value $f_p(\boldsymbol{x}_p)$ to every individual $\boldsymbol{x}_p \in \Phi_p$. The difficulty of an optimization problem f_p with regard to a specific optimization method is determined by the specific structure and complexity of f_p. Using a representation f_g results in the problem $f = f_p(f_g(\boldsymbol{x}_g))$ which can have a different difficulty.

We want to focus in the following on binary representations of integers. Therefore, $\Phi_g = \{0,1\}^l$ and $\Phi_p = \{0, 1, \ldots, 2^l - 1\}$, where l is the length of a genotype.

2.2 Properties of Representations

When using representations the distances between individuals can be changed when mapping the phenotypes on the genotypes. In the following we introduce the locality and distance distortion of an encoding. These properties of encodings describe how well representations preserve the distances between corresponding phenotypes and genotypes.

To define distances between individuals it is necessary to introduce distance metrics on Φ_g and Φ_p. Although there are other metrics possible, we use for $\Phi_g = \{0,1\}^l$ the Hamming metric. Then,

the Hamming distance between two genotypes $\boldsymbol{x}_g$ and $\boldsymbol{y}_g$ is defined as $d^g_{\boldsymbol{x}_g,\boldsymbol{y}_g} = \sum_{i=0}^{l-1} |x_{g,i} - y_{g,i}|$, where $x_{g,i}$ denotes the ith bit of the genotype $\boldsymbol{x}_g$. On the phenotypic space Φ_p the distance between two phenotypes x_p and y_p should be defined as $d^p_{x_p,y_p} = |x_p - y_p|$.

2.2.1 Locality

The locality of a representation f_g describes how well neighboring phenotypes correspond to neighboring genotypes. Two individuals $\boldsymbol{x}, \boldsymbol{y} \in \Phi$ are neighbors if $d_{\boldsymbol{x},\boldsymbol{y}} = d_{min}$, where d_{min} is the minimal distance between two individuals in Φ. Using binary or integer search spaces Φ_g and Φ_p results in $d_{min} = 1$. In general, the locality d_m of a representation can be defined as

$$d_m = \sum_{d^p_{\boldsymbol{x}_i,\boldsymbol{x}_j} = d^p_{min}} |d^g_{\boldsymbol{x}_i,\boldsymbol{x}_j} - d^g_{min}|,$$

where $d^p_{\boldsymbol{x}_i,\boldsymbol{x}_j}$ is the phenotypic distance between the phenotypes $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, $d^g_{\boldsymbol{x}_i,\boldsymbol{x}_j}$ is the genotypic distance between the corresponding genotypes, and d^p_{min}, respective d^g_{min} is the minimum distance between two (neighboring) phenotypes, respectively genotypes. We see that d_m only considers neighboring phenotypes ($d^p_{\boldsymbol{x}_i,\boldsymbol{x}_j} = d^p_{min}$).

For $d_m = 0$ an encoding has perfect locality and all phenotypic neighbors correspond to genotypic neighbors. Then, in general, a small change of a genotype results in a small change of the corresponding phenotype. The situation is different if $d_m \neq 0$. Then, small changes in a genotype can result in large changes in the phenotype. This means applying a mutation operator to an individual can result in a phenotype that has nothing in common with its parent. Therefore, when using low locality encodings systematic local search is no longer possible, but local search based on small mutation steps becomes random search.

As a result, the locality of an encoding affects the difficulty of problems. If we assume that an encoding $f_g(x_g)$ has perfect locality ($d_m = 0$), then all neighboring phenotypes correspond to neighboring genotypes. Therefore, mutation-based search approaches using a representation f_g show the same performance on the optimization problem $f = f_p(f_g(\boldsymbol{x}_g))$ as on the original optimization problem $f_p(\boldsymbol{x}_p)$. A mutation step has the same effect on genotypes and phenotypes. This means for mutation-based search that encodings with perfect locality ensure that the difficulty of the optimization problem $f_p(\boldsymbol{x}_p)$ is the same as the difficulty of $f = f_p(f_g(\boldsymbol{x}_g))$. Using a representation f_g where $d_m = 0$ ensures that problems f_p which are easy remain easy and problems f_p which are difficult remain difficult for mutation-based search.

2.2.2 Distance Distortion

When using crossover-based search, the locality concerning small changes d_m must be extended towards locality concerning small and large changes. The distance distortion d_c describes how well the phenotypic distance structure is preserved by an encoding f_g when mapping Φ_p on Φ_g:

$$d_c = \frac{2}{n_p(n_p - 1)} \sum_{i=1}^{n_p} \sum_{j=i+1}^{n_p} |d^p_{\boldsymbol{x}_i,\boldsymbol{x}_j} - d^g_{\boldsymbol{x}_i,\boldsymbol{x}_j}|,$$

where $n_p = |\Phi_g| = |\Phi_p|$, and $d_{g,min} = d_{p,min}$.

For $d_c = 0$ the distances between the phenotypes are the same as the distances between the corresponding genotypes. Then, genotypes and phenotype have the same distance structure. Perfect locality ($d_m = 0$) is a necessary condition for an encoding f_g to preserve the distances between the individuals ($d_c = 0$).

To allow standard crossover operators (for example one-point or uniform crossover) to work well, the representation f_g must preserve the distances between the individuals ($d_c = 0$). In general, crossover operators should create offspring that inherit the properties of their parents. This means in terms of distances that when using crossover the distances between an offspring $\boldsymbol{x}_o$ and its parents $\boldsymbol{x}_{p1}$ and $\boldsymbol{x}_{p2}$ should be smaller than the distances between both parents ($d_{\boldsymbol{x}_o,\boldsymbol{x}_{p1}}, d_{\boldsymbol{x}_o,\boldsymbol{x}_{p2}} \leq d_{\boldsymbol{x}_{p1},\boldsymbol{x}_{p2}}$). Otherwise, the crossover operator would create offspring which have nothing in common with their parents and a genetic algorithm (GA) using crossover would become random search.

When using representations that do not change the distances between the individuals ($d_c = 0$), standard crossover operators always create offspring that are similar to their parents and the distance between offspring and parent is smaller than the distance between both parents. However, when using representations where $d_c \neq 0$ the phenotypic distance between offspring and parent can be larger than the phenotypic distance between both parents even if the crossover operator works fine ($d^g_{\boldsymbol{x}_o,\boldsymbol{x}_{p1}}, d^g_{\boldsymbol{x}_o,\boldsymbol{x}_{p2}} \leq d^g_{\boldsymbol{x}_{p1},\boldsymbol{x}_{p2}}$). As a result, using a representation f_g, where $d_c \neq 0$ changes the difficulty of the optimization problem f_p. The preservation of distances ($d_c = 0$) is necessary to ensure that the used encoding f_g does not change the difficulty of a problem for crossover-based search approaches.

Our concept of distance distortion d_c is similar to the concept of causality which was introduced by Sendhoff et al. (1997) as a measurement of problem complexity. In contrast to their work, we do not investigate what makes a problem difficult, but we focus on how representations modify the difficulty of a given problem. We do not want to measure problem difficulty, but just decide whether a representation changes distances or not ($d_c = 0$ or $d_c \neq 0$).

3 Binary Representations of Integers

In the following we define a class of integer optimization problems we use for our investigations. In subsection 3.2 and 3.3 we briefly characterize the binary and the Gray encoding with respect to their locality and distance distortion.

3.1 Integer Optimization Problems

All our integer test problems f_p should be easy and they should be defined on the phenotypes $x_p \in \{0, 1, \ldots 2^l - 1\}$ independently of the used representation.

$$f_p(x_p) = x_{max} - |x_p - a|, \tag{1}$$

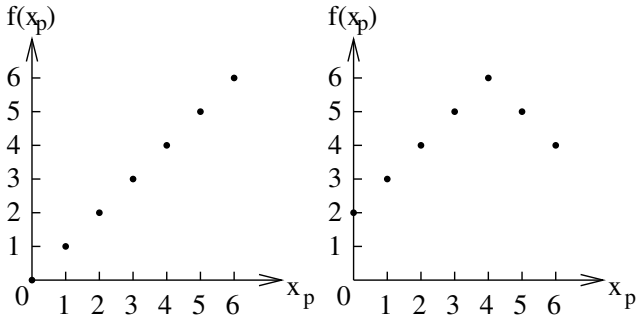
where $l \in \mathbb{N}$ and $a \in \{0, 1, \ldots, x_{max}\}$. For $a = x_{max}$ the problem becomes the standard Bin-Int problem (compare Figure 1(a)).

The difficulty of this class of optimization problems is independent of the parameter a. a only changes the location of the optimal solution in the search space and evolutionary search algorithms should show the same performance for different values of a. Two examples for the integer one-max problem are given in Figure 1.

3.2 Binary Encoding

When using the *binary encoding*, each integer value $x_p \in \Phi_p = \{1, 2, \ldots, x_{max}\}$ is represented by a binary string $\boldsymbol{x}_g$ of length $l = \lceil \log_2(x_{max}) \rceil$. The genotype-phenotype mapping f_g is defined as $x_p = f_g(\boldsymbol{x}_g) = \sum_{i=0}^{l-1} 2^i x_{g,i}$, with $x_{g,i}$ denoting the ith bit of $\boldsymbol{x}_g$.

This encoding has problems associated with the *Hamming cliff* (Schaffer, Caruana, Eshelman, & Das, 1989). The Hamming cliff describes the effect that some neighboring phenotypes are



(a) $a = 6$ (b) $a = 4$

Figure 1: Two examples for the easy integer problem

represented by genotypes with much larger distances. Examples are $x_p = 7$ and $y_p = 8$ ($d^p_{x_p,y_p} = 1$). The corresponding genotypes are $\boldsymbol{x}_g = 0111$ and $\boldsymbol{y}_g = 1000$, where $d^g_{\boldsymbol{x}_g,\boldsymbol{y}_g} = 4$.

As a result of the Hamming cliff the binary encoding has partially low locality and $d_m \neq 0$. Therefore, the encoding also does not preserve the distances between the individuals ($d_c \neq 0$). An example for the use of the binary encoding is given in Table 1.

3.3 Gray Encoding

To overcome problems with the Hamming cliff when using the binary encoding, the *Gray encoding* was developed (Gray, 1953). For the Gray encoding all phenotypic neighbors correspond to genotypic neighbors.

The Gray encoded bitstring itself can be constructed in two steps. At first, the phenotype is encoded using the binary encoding, and subsequently the binary encoded string can be converted into the corresponding Gray encoded string. The binary string $\boldsymbol{x} \in \{0,1\}^l = \{x_1, x_2, \ldots, x_l\}$ is converted to the corresponding Gray code $\boldsymbol{y} \in \{0,1\}^l = \{y_1, y_2, \ldots, y_l\}$ by the mapping $\gamma : \mathbb{B}^l \to \mathbb{B}^l$:

$$y_i = \begin{cases} x_i & \text{if } i = 1, \\ x_{i-1} \oplus x_i & \text{otherwise,} \end{cases}$$

where $\oplus$ denotes addition modulo 2.

The Gray encoding overcomes the problems with the Hamming cliff. Every two neighboring phenotypes ($d_{\boldsymbol{x}_p,\boldsymbol{y}_p} = 1$) are encoded by neighboring genotypes ($d_{x_g,y_g} = 1$). Therefore, the locality of the Gray encoding is perfect ($d_m = 0$). The high locality gives Gray encoding an advantage in comparison to binary encoding for mutation-based search. This effect has already been noticed by other work (Whitley, Rana, & Heckendorn, 1997; Whitley, 1999) which found that Gray encoding introduces less local optima in comparison to binary encoding. However, the encoding changes the distances between the individuals ($d_c \neq 0$). For example, the distance between the phenotypes $x_p = 0$ and $y_p = 3$ is $d_{x_p,y_p} = 3$, whereas the genotypes $\boldsymbol{x}_g = 000$ and $\boldsymbol{y}_g = 010$ have only distance $d^g_{\boldsymbol{x}_g,\boldsymbol{y}_g} = 1$. Table 1 gives an example for the use of the Gray encoding.

x_p	0	1	2	3	4	5	6	7
binary	000	001	010	011	100	101	110	111
Gray	000	001	011	010	110	111	101	100

Table 1: An example for using binary and Gray encodings

4 Experiments

This section addresses two issues. Firstly, the section should illustrate how Gray and binary encoding influence the performance of mutation-based and crossover-based genetic search. The presented results confirm previous theoretical results (Whitley, Rana, & Heckendorn, 1997; Whitley, 1999) which theoretically showed that mutation-based search performs best using Gray encoding. However, when using crossover as the main search operator their results can not be used any more. Dependent on the considered problem sometimes Gray and sometimes binary encoding show better performance.

Secondly, we want to explain the different results for binary and Gray encoding by using the concepts of locality and distance distortion. We have illustrated in section 3 that perfect locality ($d_m = 0$) ensures that the difficulty of a problem remains unchanged for mutation-based search. Therefore, mutation-based search approaches using the Gray encoding show the same performance on problems of the same difficulty. In contrast, if the locality is low ($d_m \neq 0$) or the distances between individuals are changed ($d_c \neq 0$) the difficulty of the problem is changed by f_g. Then, some easy problems become more difficult and evolutionary algorithms can have more problems in solving them.

The section starts with mutation-based search and continues in subsection 4.2 with crossover-based search.

4.1 Mutation-Based Search Using Simulated Annealing

We investigate how the locality of an encoding influences the performance of mutation-based search approaches. We assume in our investigations that the integer problem defined in equation 1 is easy for mutation-based search independent of the position of the optimal solution a. As a result, representations with perfect locality ($d_m = 0$) should show the same performance independently of a. In contrast, representations with low locality ($d_m \neq 0$) will change the difficulty of some easy problems and make these problems more difficult to solve for mutation-based search approaches.

We want to use simulated annealing (SA) as a representative for mutation-based search because it only uses mutation, and can in contrast to, for example a (1+1) evolution strategy, solve difficult multi-modal problems more easily. Simulated annealing can be modeled as a GA with population size 1 and Boltzmann selection (Mahfoud & Goldberg, 1995). In each generation a genotypic offspring $\boldsymbol{x}_o$ is created by applying mutation to the parent $\boldsymbol{x}_p$. Therefore, if we use bit-flipping-mutation, $\boldsymbol{x}_o$ always has genotypic distance 1 to its parent $\boldsymbol{x}_p$. If $\boldsymbol{x}_o$ has higher fitness than $\boldsymbol{x}_p$ it replaces $\boldsymbol{x}_p$. If it has lower fitness it replaces $\boldsymbol{x}_p$ with probability $P(T) = \exp\left(-\frac{f(\boldsymbol{x}_o)-f(\boldsymbol{x}_p)}{T}\right)$. With lowering the temperature T, the probability of accepting worse solutions decreases. For further information the reader is referred to other work (van Laarhoven & Aarts, 1988).

For our first investigation we concatenate 10 integer problems, where $x_p \in \{0, \dots 31\}$. When using Gray or binary encoding, each of the 10 phenotypic integers $x_p \in \{0, \dots, 31\}$ corresponds to 5 bits in the genotype ($l = 5$). Therefore, the overall length of a genotype is $l_{\boldsymbol{x}_g} = 50$. The fitness of an individual is calculated as the sum over the fitness of the 10 sub-problems. The fitness of one sub-problem is calculated according to equation 1.

Figure 2 presents results using simulated annealing for two instances ($a = 15$ and $a = 31$) of the integer problem defined in equation 1. We show the number of correctly solved sub-problems over the number of fitness evaluations. The start temperature $T_{start} = 50$ is reduced in every step by the factor 0.995. Therefore, $T_{t+1} = 0.995 * T_t$. Mutation is defined to randomly change one bit in the genotype. We performed 100 runs and each run was stopped after 2000 mutation steps. The results show that mutation-based search approaches using Gray encoding always solve all 10

sub-problems. In contrast, for $a = 15$ mutation-based search using binary encoding gets stuck in local optima because the optimal solution lies in areas with low locality.

To generalize our investigation and determine exactly how the performance of mutation-based search depends on the structure of the integer optimization problem in Figure 3 we illustrate how SA performance depends on the value of the optimal solution a. We show results for $l = 3$ (left) and $l = 5$ (right). We only change a and use the same parameter settings as before. The plots show that using Gray encoding allows SA to reliably find the optimal solution independently of the location of the best solution. Using binary encoding often results in lower SA performance and the SA gets stuck in local optima.

We can explain the performance differences by using the concept of locality. We assume that our integer problems are easy for mutation-based search independently of the value of the optimal solution a. High locality encodings like Gray encoding do not change the difficulty of the problems and all problems remain easy independently of a. Low locality encodings like binary encoding, however, change the difficulty of some easy problems and make them more difficult. As a result, SA has larger problems finding optimal solutions and the performance of mutation-based search decreases.

4.2 Crossover-Based Search Using Genetic Algorithms

In this subsection we investigate how the performance of crossover-based search depends on the used Gray or binary encoding.

We have seen in section 3 that both, Gray and binary encoding, do not preserve the distances between the individuals. Therefore, offspring produced by standard crossover mechanism could have nothing in common with their parents. For example, if we use binary encoding and uniform crossover we can get from the parents $x_p = 4$ ($x_g = 100$) and $y_p = 3$ ($y_g = 011$) the offspring $z_p = 7$ ($z_g = 111$). The offspring phenotypically has nothing in common with its parents and the distances between the offspring and its parents are much larger than the distances between both parents. Therefore, both encodings change the difficulty of the easy integer problem. How exactly, we will illustrate in the following.

As before we concatenate 10 integer problems, where $x_p \in \{0, \dots 31\}$ and the genotypic length of a sub-problem is $l = 5$. For our investigation we use a selectorecombinative standard GA (Goldberg, 1989) using only uniform crossover and no mutation. The population size is set to $n = 20$ and we use tournament selection without replacement of size 2. We performed 100 runs, and each run was stopped after the population was fully converged.

In Figure 4 we show the number of correctly solved sub-problems over the number of generations for $a = 31$ (left) and $a = 15$ (right). GAs using binary encoding outperform Gray encoding for $a = 31$. For $a = 15$, GAs using Gray encoding perform significantly better than Gray encoding.

As before, we want to generalize our investigation and show in Figure 5 how the average number of correctly solved sub-problems at the end of the run depends on the value of the optimal solution a. We show results for $l = 3$ (left) and $l = 5$ (right). It can be seen that GAs using binary encoding perform better than Gray encoding if a is either small (the optimal solution consists mostly of only 0s) or large (the optimal solution consists mostly of only 1s). Otherwise, GAs using Gray encoding perform better.

When using crossover-based search the difficulty of the original optimization problem f_p only remains unchanged if the genotype-phenotype mapping f_g does not change the distances between the individuals ($d_c = 0$). However, both encodings, Gray and binary, change the distances between corresponding genotypes and phenotypes and therefore, change the difficulty of the optimization problem. As a result, for both encodings GA performance strongly varies for different a, although the difficulty of f_p remains constant and is independent of a.

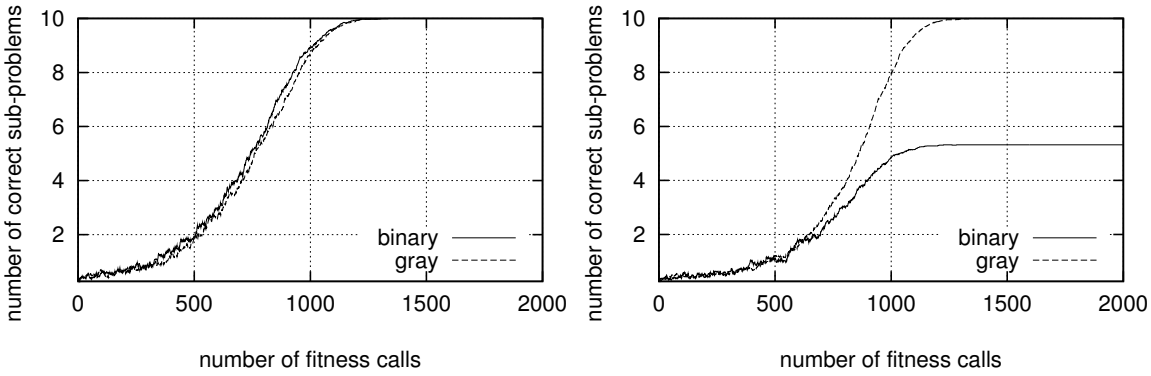


Figure 2: We use SA and show the number of correctly solved sub-problems over the number of fitness calls for $a = 31$ (left) and $a = 15$ (right).

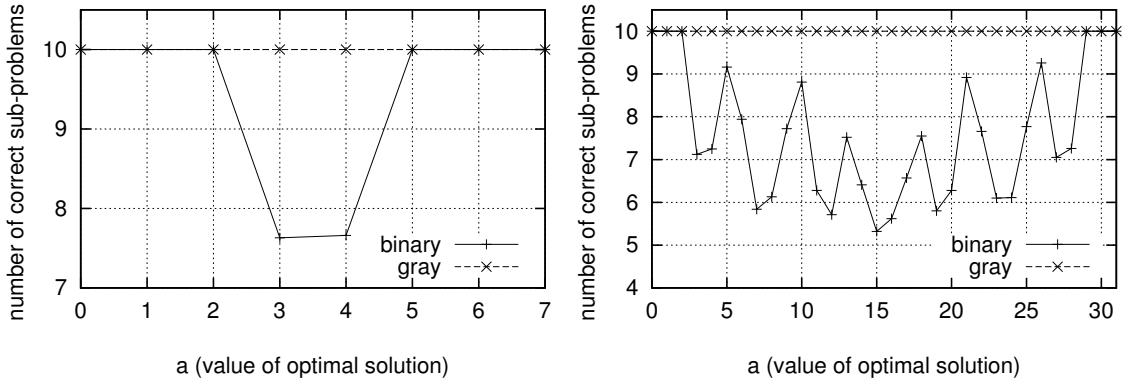


Figure 3: We use SA and show the number of correctly solved sub-problems at the end of a run over the location of the optimal solution a for $l = 3$ (left) and $l = 5$ (right).

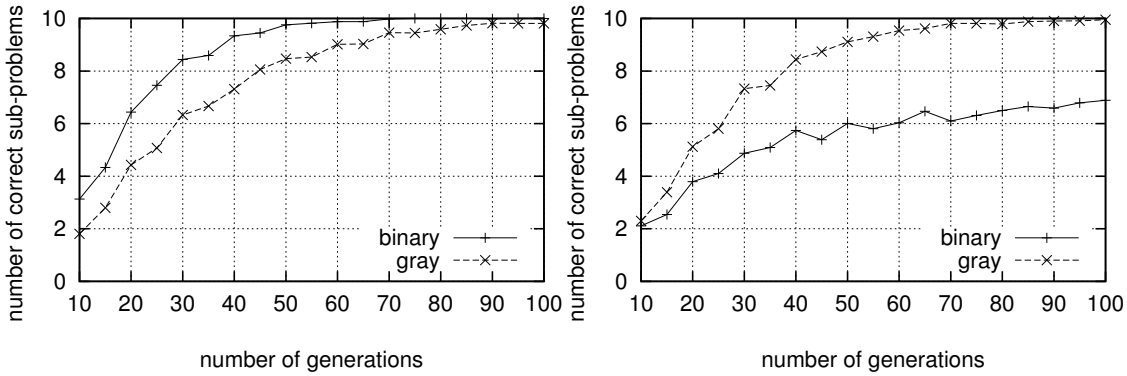


Figure 4: We use GA and show the number of correctly solved sub-problems over the number of generations for $a = 31$ (left) and $a = 15$ (right).

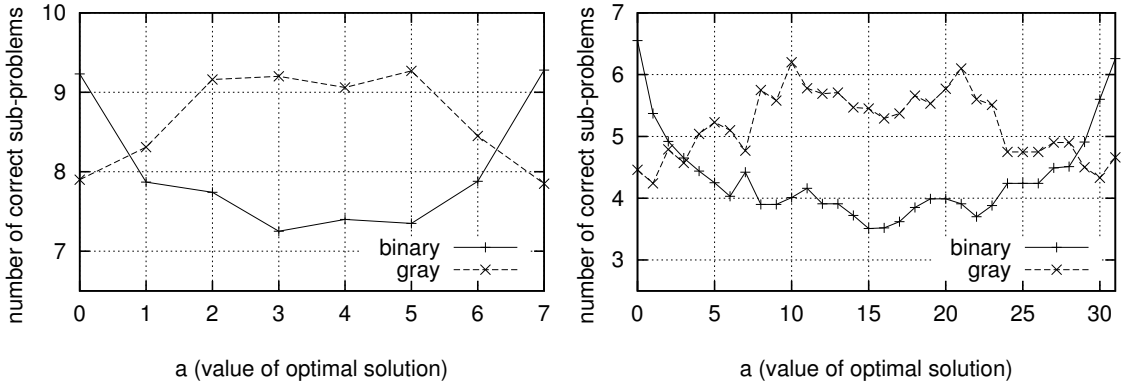


Figure 5: We use GA and show the number of correctly solved sub-problems at the end of a run over the location of the optimal solution a for $l = 3$ (left) and $l = 5$ (right).

5 Conclusions

This paper discusses the locality and distance distortion of representations and examines the performance of different types of binary representations of integers.

We illustrate that the performance of mutation-based, as well as crossover-based search methods, strongly depends on the used representations f_g. The representation f_g can change the difficulty of the optimization problem and can make problems easier, but also harder to solve. We identify two important properties of representations. The locality d_m of a representation f_g describes how well neighboring phenotypes correspond to neighboring genotypes. The distance distortion d_c determines if the distances between the individuals are preserved. We illustrate that the difficulty of a problem remains unchanged for mutation-based search if the locality of the used representation is perfect ($d_m = 0$). In analogy, the difficulty of a problem regarding crossover-based search stays the same if the distances between the individuals are preserved by the representation.

When using these concepts for binary representations of integers we see that Gray encoding has perfect locality and therefore preserves the difficulty of optimization problems. As a result, easy problems remain easy. In contrast, the binary encoding has low locality and some easy problems become more difficult. When using crossover-based search both encodings change the distances between the individuals. Therefore, not all easy problems remain easy but some of them become more difficult to solve.

We have seen that representations that preserve the distance structure between individuals do not change the difficulty of a problem. If we would be able to describe in a more detailed way how representations that do not preserve the distance structure modify problem difficulty, we could use representations to make problems easier and use our search methods more efficiently.

References

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning.* Reading, MA: Addison-Wesley.

Gray, F. (1953, March). Pulse code communications. U.S. Patent 2632058.

Liepins, G. E., & Vose, M. D. (1990). Representational issues in genetic optimization. *Journal of Experimental and Theoretical Artificial Intelligence*, *2*, 101–115.

Mahfoud, S. W., & Goldberg, D. E. (1995). Parallel recombinative simulated annealing: A genetic algorithm. In *Parallel Computing*, Volume 21 (pp. 1–28). Amsterdam, The Netherlands: Elsevier Science.

Schaffer, J. D., Caruana, R. A., Eshelman, L. J., & Das, R. (1989). A study of control parameters affecting online performance of genetic algorithms for function optimization. In Schaffer, J. D. (Ed.), *Proceedings of the Third International Conference on Genetic Algorithms* (pp. 51–60). San Mateo, CA: Morgan Kaufmann.

Sendhoff, B., Kreutz, M., & von Seelen, W. (1997). A condition for the genotype-phenotype mapping: Causality. In Bäck, T. (Ed.), *Proceedings of the Seventh International Conference on Genetic Algorithms* (pp. 73–80). San Francisco: Morgan Kaufmann.

van Laarhoven, P. J. M., & Aarts, E. H. L. (1988). *Simulated Annealing: Theory and Applications.* Dordrecht, The Netherlands: Kluwer.

Whitley, D. (1999). A free lunch proof for gray versus binary encodings. In Banzhaf, W., Daida, J., Eiben, A. E., Garzon, M. H., Honavar, V., Jakiela, M., & Smith, R. E. (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference: Volume 1* (pp. 726–733). San Francisco, CA: Morgan Kaufmann Publishers.

Whitley, D., Rana, S., & Heckendorn, R. (1997). Representation issues in neighborhood search and evolutionary algorithms. In *Genetic Algorithms and Evolution Strategy in Engineering and Computer Science* (Chapter 3, pp. 39–58). West Sussex, England: John Wiley & Sons Ltd.