A Tunable Model for Multi-Objective, Epistatic, Rugged, and Neutral Fitness Landscapes

Thomas Weise, Stefan Niemczyk, Hendrik Skubch, Roland Reichle, Kurt Geihs
University of Kassel
Wilhelmshöher Allee 73
34121 Kassel, Germany
{weise,niemczyk,skubch,reichle,geihs}@vs.uni-kassel.de

ABSTRACT

The fitness landscape of a problem is the relation between the solution candidates and their reproduction probability. In order to understand optimization problems, it is essential to also understand the features of fitness landscapes and their interaction. In this paper we introduce a model problem that allows us to investigate many characteristics of fitness landscapes. Specifically noise, affinity for overfitting, neutrality, epistasis, multi-objectivity, and ruggedness can be independently added, removed, and fine-tuned. With this model, we contribute a useful tool for assessing optimization algorithms and parameter settings.

Categories and Subject Descriptors

F.2.1 [**Analysis of Algorithms and Problem Complexity**]: Numerical Algorithms and Problems; G.1.6 [**Numerical Analysis**]: Optimization; H.1.1 [**Models and Principles**]: Systems and Information Theory; I.2.0 [**Artificial Intelligence**]: General

General Terms

Experimentation, Measurement, Theory

Keywords

Genetic Algorithm, Fitness Landscape, Multi-Objective, Epistasis, Ruggedness, Neutrality, Model, Benchmark

1. INTRODUCTION

Many real-world problems can be solved very efficiently by probabilistic optimization methods like evolutionary algorithms. However, some frequently occurring characteristics cause difficulties for optimization techniques. Some of the most important features that influence the problem hardness for probabilistic optimizers are the problem size, affinity for overfitting and oversimplification, neutrality, epistasis, multi-objectivity, ruggedness, and deceptiveness. Often, the influence of these features on the progress of optimization and their interactions with each other are unclear and complicated in real-world applications. Hence, our goal is to find a common approach for studying them and to define simple model problems where they become tangible. We are confident that the analysis of these features will lead to the development of more robust and more efficient optimization methods.

The main contribution of this paper is a new model problem that exhibits all of the aforementioned features in a controllable manner. Each of them is introduced as a distinct filter component in the problem which can separately be activated, deactivated, and fine-tuned. The model problem is comprehensive, yet simple. It allows for extensive experiments being conducted in a small timeframe, making it an ideal tool to assess different optimization algorithms or parameter settings. Additionally, it is also well suited for theoretical analysis because of its simplicity.

The paper is organized as follows. In Section 2, we will present several features which are well known to influence the quality and efficiency of optimization techniques. Some inspiring related work is outlined in Section 3. In Section 4, we propose our new model problem. We substantiate the considerations that were used for defining our model with some first experimental results provided in Section 5. We finally conclude and give pointers to future work in Section 6.

2. FITNESS LANDSCAPES

In biology, a fitness landscape is the visualization of the relationship of the genotypes to their corresponding reproduction probability [44, 18, 13, 5]. In global optimization algorithms, it displays the relation of the reproduction operations, the solution candidates, and their fitness or objective values [23, 21, 9]. In genetic algorithms, we distinguish between genotypes and phenotypes. The genotypes, the elements of the search space $\mathbb{G}$, are bit strings of fixed or variable length [15]. They are translated to phenotypes in the problem space $\mathbb{X}$ with a genotype-phenotype mapping (GPM) as illustrated in Figure 1. The problem space can virtually be anything, from the real numbers, the possible combinations of different accessories for a car, to, as in case of the example, the points in a two-dimensional plane. $m \geq 1$ objective functions represent the criteria subject to optimization. They rate the qualities of the features of the phenotypes and, by doing so, map them to points in the objective space (usually $\mathbb{R}^m$). Especially if multiple criteria are to be optimized ($m > 1$), the objective values are mapped to fitness values in $\mathbb{R}^+$. The genetic algorithm uses this infor-

GECCO'08, July 12–16, 2008, Atlanta, Georgia, USA.

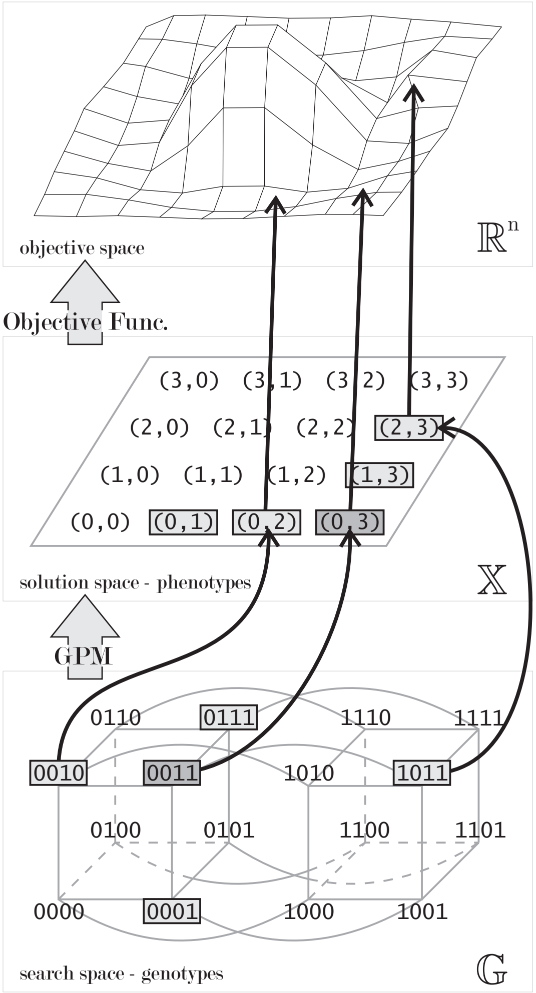

Figure 1: The relation of search space, problem space, and objective space.

mation to determine which solution candidates are worthy of further exploration. New points in the search space are generated by applying *reproduction operators* on the genotypes. In general, this flow is the same for all optimization algorithms although they differ in the way they conduct the search [35].

The shape of the fitness landscape has a major impact on the quality of the solutions found by the optimization process. In this section, we discuss some of the basic features of fitness landscapes and how they influence each other.

2.1 Ruggedness and Causality

It is a general rule for genotype design that it should exhibit causality [29, 28]. The principle of strong causality (locality) states that small changes in an object lead to small changes in its behavior [32, 33]. In rugged fitness landscapes, this is not the case: small changes in an individual's genotype often cause large changes in its fitness. This hinders an optimization algorithm in finding and climbing a gradient in objective space. A region of the fitness landscape is *deceptive* if performing a gradient descend does not lead towards a solution but instead away from it. Ruggedness and deceptiveness are closely related. Matter of fact, in the model proposed in this work, there is a smooth transition between the two phenomena.

2.2 Epistasis

In biology, *epistasis* is defined as a form of interaction between different genes. It was coined by Bateson [2] in order to describe how one gene can suppress the phenotypical expression of another gene. According to Lush [22, 1], the interaction between genes is epistatic if the effect on the fitness from altering one gene depends on the allelic[1] state of other genes.

Epistasis in evolutionary algorithms means that a change in one property of a solution candidate, induced by a reproduction operator, also leads to a change in some of its other properties [4, 27]. We speak of minimal epistasis when every gene is independent of every other gene and of maximal epistasis if every genes is dependent on every other gene [31, 26].

Epistasis violates the locality principle previously discussed, since a modification in a genotype will alter multiple properties of a phenotype, probably leading to ruggedness in the fitness landscape.

2.3 Neutrality

We call the application of a reproduction operator to a solution candidate *neutral* if it yields no relevant change in objective space. Redundancy in the genome (multiple genotypes that translate to the same phenotype) always leads to neutrality. Neutrality and redundancy exist both in natural as well as in artificial evolution [40, 39]. Neutrality may have positive [38, 37] as well as negative [34, 36] effects on the optimization process.

2.4 Overfitting and Oversimplification

Overfitting and oversimplification are very common phenomena in all applications where the objective functions are evaluated using sample data as is the case in many applications of Genetic Programming like function fitting and symbolic regression, for instance. Both, overfitting and oversimplification, lead to solutions that work correctly with the data samples used during training but fail to deliver acceptable results for inputs not occurring in the training set.

If training sets only represent a fraction of the input space, the resulting incomplete coverage may fail to represent some of the dependencies and characteristics of the data, leading to oversimplified solutions. Noise in the training data can lead to wrong assumptions. The optimizer may for example try to find a "model" for the noise [25] and thus overfits the sample data. It should be noted that an overfitted individual can even have better a fitness than the correct solution itself when evaluated using the training data [25].

2.5 Multi-Objectivity

Many optimization problems are multi-objective, i. e., involve multiple, possible conflicting criteria [6, 3, 10]. In Genetic Programming, for instance, we want to evolve algorithms which are not only correct but also small and resource friendly.

---

[1] A gene in the context of evolutionary algorithms is a part of a genotype which encodes a distinguishable part of a phenotype and an allele is its value.

3. MODELING AND BENCHMARKING

The computational effort to solve optimization problems depends on their fitness landscapes. It is thus beneficial to understand the features previously discussed. We can obtain such knowledge by artificially creating problems that exhibit some of them in a tunable manner. Using these problems, we can run practical experiments as well as derive hardness measures mathematically.

3.1 Kauffman's NK Fitness Landscapes

In the late 1980s, Kauffman defined the *NK fitness landscape* [19, 17, 20], a family of fitness functions with tunable epistasis, in an effort to investigate the links between epistasis and ruggedness.

The phenotypes of this problem are bit strings of the length N ($\mathbb{X} = \{0,1\}^N$). In terms of the NK landscape, only one single objective function is used: $F_{N,K} : \{0,1\}^N \mapsto \mathbb{R}^+$ and each bit contributes one value to this objective. The "fitness" f_i of a bit $x[i]$ is determined by its value and the values of K other bits $x[i_1], x[i_2], \dots, x[i_K]$ with $i_{1\dots K} \in [0, N-1] \setminus \{i\}$, called its *neighbors*.

$$F_{N,K}(x) = \frac{1}{N} \sum_{i=0}^{N-1} f_i(x[i], x[i_1], x[i_2], \dots, x[i_K]) \qquad (1)$$

Whenever the value of a bit changes, the fitness values of all the bits to whose neighbor set it belongs will change uncorrelated to their previous state. If $K = 0$, there is no epistasis at all. For $K = N - 1$ the epistasis is maximized and the fitness contribution of each gene depends on all other genes.

We can consider the f_i as single objective functions that are combined to a fitness value $F_{N,K}$ by averaging. Then, NK Fitness landscapes can lead to another well known aspect of multi-objective optimization: conflicting criteria. An improvement in one objective may very well lead to degeneration in another one.

The properties of the NK landscapes have intensely been studied in the past, most notably by Kauffman [18], Weinberger [41], and Fontana et al. [11].

3.2 The Royal Road

The Royal Road Functions [16, 24, 12] are a set of special fitness landscapes for genetic algorithms with fixed-length bit string genomes. In genetic algorithms, schemas are blueprints of binary strings that may contain *don't care*-symbols (*) at different loci [15]. The Royal Road Functions define a set S of such schemas $s_1, s_2, \dots, s_n$ and an objective function, subject to maximization, as

$$f(x) = \sum_{\forall s \in S} c(s)\sigma(s, x) \qquad (2)$$

where x is the solution candidate, $c(s)$ is a value assigned to the schema s (usually its order), and $\sigma(s, x)$ is one if x is an instance of s and zero otherwise. Listing 1 outlines an example for the Royal Roads.

```
s1 = 11******; c(s1) = 2
s2 = **11****; c(s2) = 2
s3 = ****11**; c(s3) = 2
s4 = ******11; c(s4) = 2
s5 = 1111****; c(s5) = 4
s6 = ****1111; c(s6) = 4
s7 = 11111111; c(s7) = 8
```

Listing 1: An example Royal Road Function.

The original Royal Road problems can be defined for binary string genomes of any given length n, as long as n is fixed. A Royal Road benchmark for variable-length genomes has been defined by Defoin Platel et al. [7]. In [8], the same authors combine their previous work on the Royal Road with Kauffman's NK landscapes and introduced the Epistatic Road.

An analogue of the Royal Road for Genetic Programming has been specified by Punch et al. [30]. This *Royal Tree* problem specifies a series of functions A, B, C, ... with increasing arity, i. e., A has one argument, B has two arguments, C has three, and so on. Additionally, a set of terminal nodes x, y, z is defined. A perfect tree has a certain predefined depth. Its root is a A node, which has B nodes as children which, in turn, have C nodes attached to them and so on.

4. MODEL DEFINITION

The goal of our research presented in this paper was to define a model problem with tunable ruggedness, epistasis, neutrality, multi-objectivity, overfitting, and oversimplification features. The distinct layers of this problem, each introducing one of these aspects independently, are outlined using an example in Figure 2 and are specified in the following sections.

The basic problem is to find a binary string $x^\star$ of a predefined length n consisting of alternating zeros and ones. The tuning parameter for the problem size is $n \in \mathbb{N}$.

$$x^\star = 0101010101010\dots01, \; |x^\star| = n \qquad (3)$$

4.1 Overfitting and Oversimplification

Searching this optimal string could be done by comparing each genotype g with $x^\star$. Therefore we would use the Hamming distance [14], which defines the difference between two binary strings a and b of equal length as:

$$h(a, b) = |\{\forall i : a[i] \neq b[i] \land 0 \leq i < |a|\}| \qquad (4)$$

Instead of doing this directly, we test the solution candidate against t data samples. These samples are modified versions of the perfect string $x^\star$.

As outlined in Section 2.4, we can distinguish between overfitting and oversimplification. The latter is caused by incompleteness of the tests and the former can originate from noise in the test cases. Both forms can be expressed in terms of our model by the objective function $f_{\varepsilon,o,t}$ (based on a slightly extended version of the Hamming distance h^*) which is subject to minimization.

$$h_*(a, b) = |\{\forall i : a[i] \neq b[i] \land b[i] \neq * \land 0 \leq i < |a|\}| \qquad (5)$$

$$f_{\varepsilon,o,t}(x) = \sum_{i=1}^{t} h_*(x, test_i), \quad f_{\varepsilon,o,t} \in [0, (n-o)t] \qquad (6)$$

In the case of oversimplification, the perfect solution $x^\star$ will always reach a perfect score in all tests. There may be incorrect solutions reaching this value in some cases too, because some of the facets of the problem are hidden. We

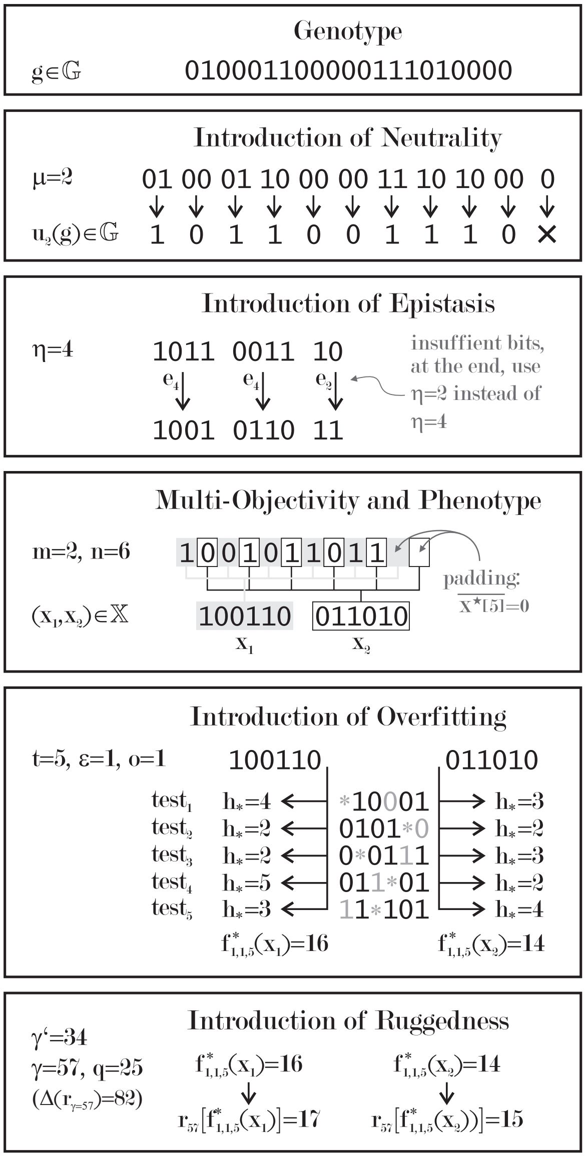

Figure 2: An example for the fitness landscape model.

take this into consideration by placing o *don't care* symbols ($*$) uniformly distributed into the test cases. The values of the solution candidates at their loci have no influence on the fitness.

When overfitting is enabled, the perfect solution will not reach the optimal score in any test case because of the noise present. Incorrect solutions may score better in some tests and even outperform the real solution if the noise level is high. Noise is introduced in the test cases by toggling ε of the remaining $n - o$ bits, again following a uniform distribution. An optimization algorithm can find a correct solution only if there are more training samples with correctly defined values for each locus than with wrong or don't care values.

4.2 Neutrality

We can create a well-defined amount of neutrality during the genotype-phenotype mapping by applying a transformation u_μ that shortens the solution candidates by a factor μ. The i^{th} bit in $u_\mu(g)$ is defined as 0 if and only if the majority of the μ bits starting at locus $i * \mu$ in g is also 0, and as 1 otherwise. The default value 1 set in draw situations has (in average) no effect on the fitness since the target solution $x^\star$ is defined as a sequence of alternating zeros and ones. If the length of a genotype g is not a multiple of μ, the remaining $|g| \bmod \mu$ bits are ignored. The tunable parameter for the neutrality in our model is μ. If μ is set to 1, no additional neutrality is modeled.

4.3 Epistasis

Epistasis in general means that a slight change in one gene of a genotype influences some other genes. We can introduce epistasis in our model as part of the genotype mapping and apply it after the neutrality transformation. We therefore define a bijective function e_η that translates a binary string z of length η to a binary string $e_\eta(z)$ of the same length. Assume we have two binary strings z_1 and z_2 which differ only in one single locus, i. e., their Hamming distance is one. e_η introduces epistasis by exhibiting the following property:

$$h(z_1, z_2) = 1 \Rightarrow h(e_\eta(z_1), e_\eta(z_2)) \geq \eta - 1 \; {\scriptstyle \forall z_1, z_2 \in \{0,1\}^\eta} \quad (7)$$

The meaning of Equation 7 is that a change of one bit in a genotype g leads to the change of at least $\eta - 1$ bits in the corresponding mapping $e_\eta(g)$. This, as well as the demand for bijectivity, is provided if we define e_η as follows:

$$e_\eta(z) = \begin{cases} e_\eta(z)[i] = \bigotimes\limits_{\substack{\forall j: 0 \leq j < \eta \wedge \\ j \neq (i-1) \bmod \eta}} z[j] & \forall z: 0 \leq z < 2^{\eta-1} \\ \overline{e_\eta(z - 2^{\eta-1})} & \textit{otherwise} \end{cases} \quad (8)$$

In other words, for all strings $z \in \{0,1\}^\eta$ which have the most significant bit (MSB) not set, the e_η transformation is performed bitwise. The i^{th} bit in $e_\eta(z)$ equals the exclusive or combination of all but one bit in z. Hence, each bit in z influences the value of $\eta - 1$ bits in $e_\eta(z)$. For all z with 1 in the MSB, $e_\eta(z)$ is simply set to the negated e_η transformation of z with the MSB cleared (the value of the MSB is $2^{\eta-1}$). This division in e is needed in order to ensure its bijectiveness. This and the compliance with Equation 7 can be shown with a rather lengthy proof omitted here.

In order to introduce this model of epistasis in genotypes of arbitrary length, we divide them into blocks of the length η and transform each of them separately with e_η. If the length of a given genotype g is no multiple of η, the remaining $|g| \bmod \eta$ bits at the end will be transformed with the function $e_{|g| \bmod \eta}$ instead of e_η, as outlined in Figure 2.

It may be an interesting fact that the e_η transformations are a special case of the NK landscape discussed in Section 3.1 with $N = \eta$ and $K \approx \eta - 2$.

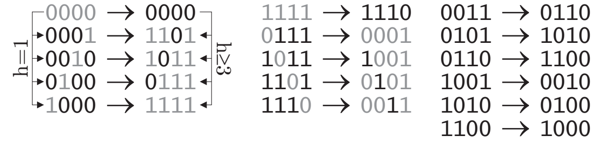

Figure 3: An example for the epistasis mapping $z \rightarrow e_4(z)$.

The tunable parameter η for the epistasis ranges from 2 to $n * m$, the product of the basic problem length n and the number of objectives m (see next section). If it is set to a value smaller than 3, no additional epistasis is introduced. Figure 3 outlines the mapping for $\eta = 4$.

4.4 Multi-Objectivity

A multi-objective problem with m criteria can easily be created by interleaving m instances of the benchmark problem with each other and introducing separate objective functions for each of them. Instead of just dividing the genotype g in m blocks, each standing for one objective, we scatter the objectives as illustrated in Figure 2. The bits for the first objective comprise $x_1 = (g_{[0]}, g_{[m]}, g_{[2m]}, \dots)$, those used by the second objective $x_2 = (g_{[1]}, g_{[m+1]}, g_{[2m+1]}, \dots)$. Notice that no bit in g is used by more than one objective. Superfluous bits (beyond index $nm - 1$) are ignored. If g is too short, the missing bits in the phenotypes are replaced with the complement from $x^{\star}$, i.e., if one objective misses the last bit (index $n - 1$), it is padded by $\overline{x^{\star}{}_{[n-1]}}$ which will worsen the objective by 1 on average.

Because of the interleaving, the objectives will begin to conflict if epistasis ($\eta > 2$) is applied, similar to NK landscapes. Changing one bit in the genotype will change the outcome of at most $\min\{\eta, m\}$ objectives. Some of them may improve while others may worsen.

A non-functional objective function minimizing the length of the genotypes is added if variable-length genomes are used during the evolution. If fixed-length genomes are used, they can be designed in a way that the blocks for the single objectives have always the right length.

4.5 Ruggedness

In an optimization problem, there can be at least two (possibly interacting) sources of ruggedness of the fitness landscape. The first one, epistasis, has already been modeled and discussed. The other source concerns the objective functions themselves, the nature of the problem. We will introduce this type of ruggedness *a posteriori* as a permutation $r : [0, q] \mapsto [0, q]$ of the objective values (where q is a convenient abbreviation for the maximum possible objective value $(n - o)t$).

Before we do that, let us shortly outline what makes a function *rugged*. Ruggedness is obviously the opposite of smoothness and causality. In a smooth objective function, the objective values of the solution candidates neighboring in problem space are also neighboring. In our original problem with $o = 0$, $\varepsilon = 0$, and $t = 1$ for instance, two individuals differing in one bit will also differ by one in their objective values. We can write down the list of objective values the solution candidates will take on when they are stepwise improved from the worst to the best possible configuration as $(q, q-1, .., 2, 1, 0)$. If we exchange two of the values in this list, we will create some artificial ruggedness. A measure for the ruggedness of such a permutation r is $\Delta(r)$:

$$\Delta(r) = \sum_{i=0}^{q-1} |r_i - r_{i+1}| \tag{9}$$

The original sequence of objective values has the minimum value $\Delta_{\min} = q$ and the maximum possible value is $\Delta_{\max} = \frac{q(q+1)}{2}$. There exists at least one permutation for each Δ value in $\Delta_{\min}..\Delta_{\max}$.

We can hence define the permutation r_γ which is applied after the objective values are computed and which has the following features:

1. It is bijective (since it is a permutation).
2. It must preserve the optimal value, i. e., $r_\gamma[0] = 0$.
3. $\Delta(r_\gamma) = \Delta_{\min} + \gamma$.

With $\gamma \in [0, \Delta_{\max} - \Delta_{\min}]$, we can fine-tune the ruggedness. For $\gamma = 0$, no ruggedness is introduced. For a given q, we can compute the permutations r_γ with the procedure *buildRPermutation*(γ, q) defined in Algorithm 1.

Algorithm 1: $r_\gamma = buildRPermutation(\gamma, q)$

```
Input: $\gamma$ the $\gamma$ value
Input: $q$ the maximum objective value
Result: $r_\gamma$ the permutation $r_\gamma$
permutate($\gamma, r, q, start$)
begin
    if $\gamma > 0$ then
        if $\gamma \leq (q-1)$ then
            $permutate(\gamma - 1, r, q, start)$
            $exchangeElementsAtIndex(r, q, q - \gamma)$
        else
            $i \longleftarrow \lfloor \frac{start+1}{2} \rfloor$
            if $(start \bmod 2) = 0$ then
                $i \longleftarrow q + 1 - i$
                $d \longleftarrow -1$
            else $d \longleftarrow 1$
            $j \longleftarrow start$
            while $j \leq q$ do
                $r[j] \longleftarrow i$
                $i \longleftarrow i + d$
                $j \longleftarrow j + 1$
            $permutate(\gamma - q + start, r, q, start + 1)$
end
begin
    $r \longleftarrow (0, 1, 2, .., q-1, q)$
    return $permutate(\gamma, r, q, 1)$
end
```

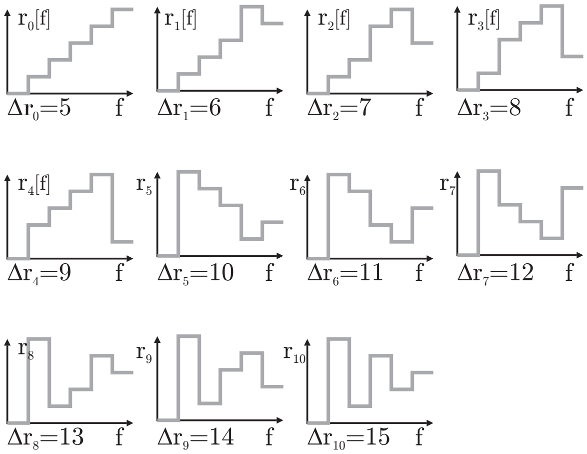

Figure 4: An example for r_γ with $\gamma = 0..10$ and $q = 5$.

Figure 4 outlines all ruggedness permutations r_γ for an objective function which can range from 0 to $q = 5$. As can be seen, the permutations scramble the objective function more and more with rising γ and reduce its gradient information.

5. FIRST EXPERIMENTAL RESULTS

In this section, we will provide a selection of the first experimental results which have been obtained with our model. For the tests, we have used a standard genetic algorithm with population size 1000, single-point crossover, single-bit mutation, and a variable-length bit string genome with a maximum string length of 8000 bits. In each test, we applied a non-functional objective minimizing the length of the strings. We suggest using these settings as default setup for all experiments involving our model in order to keep the results comparable. Furthermore, we have used tournament selection with tournament size 5 and Pareto ranking for fitness assignment. For each experimental setting, at least 50 runs have been performed.

5.1 Basic Complexity

In the experiments, we distinguish between *success* and *perfection*. Success means finding individuals x of optimal *functional* fitness, i. e., $f_{\varepsilon,o,t}(x) = 0$. Multiple such *successful* strings may exist, since superfluous bits at the end of genotypes do not influence their functional objective. The perfect string $x^\star$ has no such useless bits, it is the shortest possible solution with $f_{\varepsilon,o,t} = 0$ and, hence, also optimal in the non-functional length criterion. We will refer to the number of generations needed to find a successful individual as *success generations*.

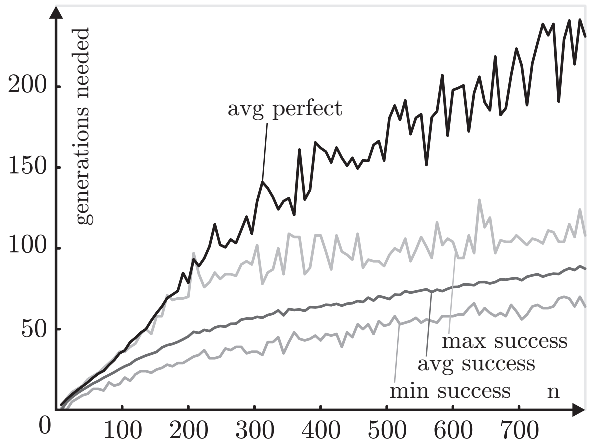

Figure 5: The basic problem hardness.

In Figure 5, we have computed the minimum, average, and maximum number of the success generations for values of n ranging from 8 to 800. As illustrated, the problem hardness increases smoothly with rising string length n. Trimming down the solution strings to the perfect length becomes more and more complicated with growing n. This is likely because the fraction at the end of the strings where the trimming is to be performed will shrinks in comparison with its overall length.

5.2 Ruggedness

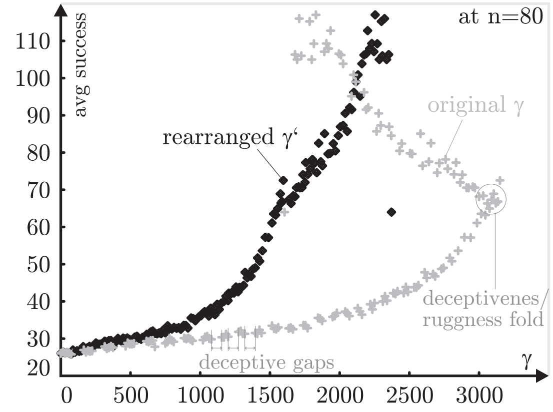

Figure 6: Experimental results for the ruggedness.

In Figure 6, we plotted the average success generations with $n = 80$ and different ruggedness settings γ. Interestingly, the gray original curve behaves very strange. It is divided into alternating solvable and unsolvable[2] problems. The unsolvable ranges of γ correspond to gaps in the curve. With rising γ, the solvable problems require more and more generations until they are solved. After a certain γ threshold value, the unsolvable sections become solvable. From there on, they become simpler with rising γ. At some point, the two parts of the curve meet.

Algorithm 2: $\gamma = translate(\gamma', q)$

```
begin
    $l \longleftarrow \frac{q(q-1)}{2}$
    $i \longleftarrow \left\lfloor \frac{q}{2} \right\rfloor * \left\lfloor \frac{q+1}{2} \right\rfloor$
    if $\gamma \leq i$ then
        $j \longleftarrow \left\lfloor \frac{q+2}{2} - \sqrt{\frac{q^2}{4} + 1 - \gamma} \right\rfloor$
        $k \longleftarrow \gamma - j\,(q+2) + j^2 + q$
        return $k + 2\left(j\,(q+2) - j^2 - q\right) - j$
    else
        $j \longleftarrow \left\lfloor \frac{(q \bmod 2)+1}{2} + \sqrt{\frac{1-(q \bmod 2)}{4} + \gamma - 1 - i} \right\rfloor$
        $k \longleftarrow \gamma - (j - (q \bmod 2))\,(j-1) - 1 - i$
        return $l - k - 2j^2 + j - (q \bmod 2)\,(-2j+1)$
end
```

The reason for this behavior is rooted in the way that we construct the ruggedness mapping r and illustrates the close relation between ruggedness and deceptiveness. Algorithm 1 alternates between creating groups of mappings that are mainly rugged and such that are mainly deceptive. In Figure 4, for instance, from $\gamma = 5$ to $\gamma = 7$, the permutations also exhibit a high degree of deceptiveness while before and after that range they are just rugged. The black curve in Figure 6 depicts the results of rearranging the γ-values with Algorithm 2. This algorithm maps deceptive gaps to higher γ-values and ensures continuity of the resulting curve. Now

---

[2] We call a problem unsolvable if it has not been solved within 1000 generations.

the mappings feature a stepwise transition from normal to rugged to deceptive from the left to the right.[3]

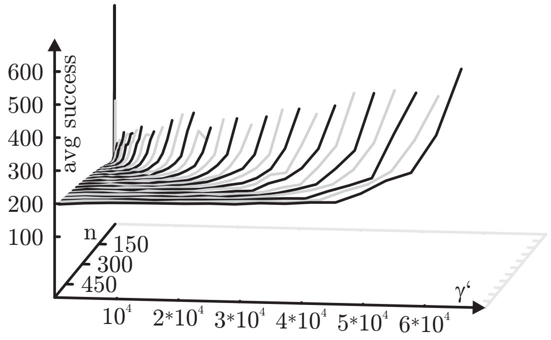

Figure 7: Experiments with the rearranged ruggedness.

Figure 7 sketches the average success generations for the rearranged ruggedness problem for multiple values of n and γ'. Depending on the basic problem size n, the problem hardness increases steeply with rising values of γ'.

5.3 Epistasis

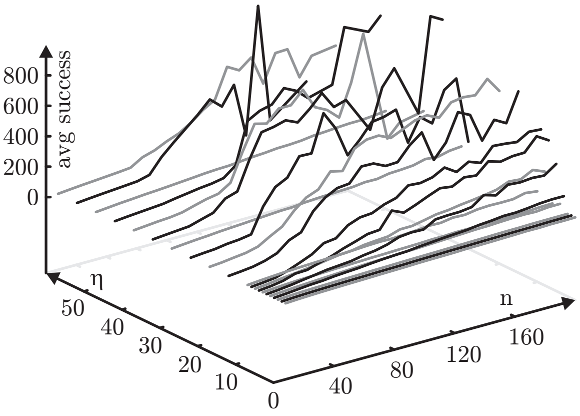

Figure 8: Experimental results for different epistasis values.

Figure 8 illustrates the relation between problem size n, the epistasis factor η, and the average success generations. Although rising epistasis makes the problems harder, the complexity does not rise as smoothly as in the previous experiments. The cause for this is likely the presence of crossover – if mutation was allowed only, the impact of epistasis would most likely be more intense. Another interesting fact is that experimental settings with odd values of η tend to be much more complex than those with even ones. We are currently investigating the reason for this phenomenon.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented a model problem providing tunable overfitting and oversimplification affinity, epistasis, neutrality, ruggedness, and multi-objectivity. Up until now, only models incorporating a subset of these features of the fitness landscape were available. Additionally, the effects of the parameter settings of these models on the features were often not obvious and only indirectly tangible.

In our model on the other hand, we can not only study all the mentioned features but also have parameters to tune them in a simple and plain manner. Of course, some of the features of the fitness landscape interact with each other as we have mentioned before (see Section 2.2 and Section 3.1). Our model, however, comes very close to separating them and allows deactivating certain aspects as far as possible for some experiments.

One part of our future work is to learn more about the impact of the model settings on the optimization process. We will therefore perform many more experiments. This will provide us with more empiric data on how the features of the fitness landscape influence the success probability of optimization. We also aim at establishing our model as a benchmark that can help to evaluate optimization algorithms in different situations in an unbiased manner.

Our other work [43, 42] focuses on an area of Genetic Programming which is very prone to high epistasis. We hope that the settings for evolutionary algorithms that perform well with the epistasis in our model will also prove to be useful for GP. For finding such settings, this benchmark has the advantage that experiments run very fast with it, while GP is rather time consuming if it involves complex simulations.

7. REFERENCES

[1] L. Altenberg. Nk fitness landscapes. In *Handbook of Evolutionary Computation*, chapter B2.7.2. Oxford University Press, 1996.

[2] W. Bateson. *Mendel's Principles of Heredity*. Cambridge University Press, 1909.

[3] C. A. Ceollo Coello. An updated survey of evolutionary multiobjective optimization techniques: State of the art and future trends. In *Congress on Evolutionary Computation (CEC)*, pages 3–13. IEEE Press, 1999.

[4] Y. Davidor. Epistasis variance: A viewpoint on GA-hardness. pages 23–35. In [31].

[5] R. Dawkins. *Climbing Mount Improbable*. Penguin Books, 1996.

[6] K. Deb. *Multi-Objective Optimization Using Evolutionary Algorithms*. Wiley Interscience Series in Systems and Optimization. Wiley & Sons, 2001.

[7] M. Defoin Platel, M. Clergue, and P. Collard. Maximum homologous crossover for linear genetic programming. In *Genetic Programming: 6th European Conf.*, volume 2610 of *LNCS*, pages 29–48. Springer, 2003.

[8] M. Defoin Platel, S. Vérel, M. Clergue, and P. Collard. From royal road to epistatic road for variable length evolution algorithm. In *Evolution Artificielle, 6th Int. Conf.*, volume 2936 of *LNCS*, pages 3–14. Springer, 2003.

[9] A. E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing*. Natural Computing Series. Springer, 2003.

[10] C. M. Fonseca and P. J. Fleming. Multiobjective optimization and multiple constraint handling with evolutionary algorithms – part i: A unified

---

[3] This is a deviation from our original idea, but this idea did not consider deceptiveness.

formulation. *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, 28(1):26–37, 1998.

[11] W. Fontana, P. F. Stadler, E. G. Bornberg-Bauer, T. Griesmacher, I. L. Hofacker, M. Tacker, P. Tarazona, E. D. Weinberger, and P. Schuster. Rna folding and combinatory landscapes. *Physical Review E*, 47(3):2083–2099, 1993.

[12] S. Forrest and M. Mitchell. Relative building-block fitness and the building-block hypothesis. In Whitley, editor, *Foundations of Genetic Algorithms 2*, pages 109–126. Morgan Kaufmann, 1992.

[13] S. Gavrilets. *Fitness Landscapes and the Origin of Species*. Number MPB-41 in Monographs in Population Biology. Princeton University Press, 2004.

[14] R. W. Hamming. Error-detecting and error-correcting codes. *Bell System Technical Journal*, 29(2):147–169, 1950.

[15] J. H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, 1975.

[16] T. Jones. A description of holland's royal road function. *Evolutionary Computation*, 2:411–417, 1994.

[17] S. A. Kauffman. Adaptation on rugged fitness landscapes. In *Lectures in the Sciences of Complexity: Complex Systems Summer School*, volume I of *Santa Fe Institute Studies in the Sciences of Complexity*, pages 527–618. Addison Wesley, 1988.

[18] S. A. Kauffman. *The Origins of Order: Self-Organization and Selection in Evolution*. Oxford University Press, 1993.

[19] S. A. Kauffman and S. Levin. Towards a general theory of adaptive walks on rugged landscapes. *Journal of Theoretical Biology*, 128(1):11–45, 1987.

[20] S. A. Kauffman and E. D. Weinberger. The nk model of rugged fitness landscapes and its application to maturation of the immune response. *Journal of Theoretical Biology*, 141:211–245, 1989.

[21] W. B. Langdon and R. Poli. *Foundations of Genetic Programming*. Springer, Berlin, 2002.

[22] J. L. Lush. Progeny test and individual performance as indicators of an animal's breeding value. *Journal of Dairy Science*, 18(1):1–19, 1935.

[23] M. Mitchell. *An Introduction to Genetic Algorithms*. Complex Adaptive Systems. MIT Press, 1998.

[24] M. Mitchell, S. Forrest, and J. H. Holland. The royal road for genetic algorithms: Fitness landscapes and GA performance. In Varela and Bourgine, editors, *Towards a Practice of Autonomous Systems: 1st European Conf. on Artificial Life*, pages 245–254. The MIT Press, 1992.

[25] D. J. Montana. Strongly typed genetic programming. *Evolutionary Computation*, 3(2):199–230, 1995.

[26] B. Naudts, D. Suys, and A. Verschoren. Generalized royal road functions and their epistasis. *Computers and Artificial Intelligence*, 19(4), 2000.

[27] B. Naudts and A. Verschoren. Epistasis on finite and infinite spaces. In *8th Int. Conf. on Systems Research, Informatics and Cybernetics*, pages 19–23, 1996.

[28] C. C. Palmer. *An approach to a problem in network design using genetic algorithms*. PhD thesis, Polytechnic University, New York, NY, 1994.

[29] C. C. Palmer and A. Kershenbaum. Representing trees in genetic algorithms. In *1st IEEE Conf. on Evolutionary Computation (CEC)*, volume 1, pages 379–384. IEEE Press, 1994.

[30] W. F. Punch, D. Zongker, and E. D. Goodman. The royal tree problem, a benchmark for single and multiple population genetic programming. In Angeline and Kinnear, Jr, editors, *Advances in Genetic Programming 2*, pages 299–316. MIT Press, 1996.

[31] Rawlins, editor. *1st Workshop on Foundations of Genetic Algorithms (FOGA)*. Morgan Kaufmann, 1990.

[32] I. Rechenberg. *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Frommann-Holzboog Verlag, 1973.

[33] J. P. Rosca and D. H. Ballard. Causality in genetic programming. In *6th Int. Conf. on Genetic Algorithms (ICGA95)*, pages 256–263. Morgan Kaufmann, 1995.

[34] F. Rothlauf. *Representations for Genetic and Evolutionary Algorithms*. Physica-Verlag, 2002.

[35] S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, second edition, 2002.

[36] M. Shackleton, R. Shipman, and M. Ebner. An investigation of redundant genotype-phenotype mappings and their role in evolutionary search. In *Congress on Evolutionary Computation (CEC)*, pages 493–500. IEEE Press, 2000.

[37] R. Shipman. Genetic redundancy: Desirable or problematic for evolutionary adaptation? In *4^{th} Int. Conf. on Artificial Neural Nets and Genetic Algorithms*, pages 1–11. Springer, 1999.

[38] M. Toussaint and C. Igel. Neutrality: A necessity for self-adaptation. In *IEEE Congress on Evolutionary Computation (CEC 2002)*, pages 1354–1359. IEEE Press, 2002.

[39] A. Wagner. *Robustness and Evolvability in Living Systems*. Princeton Studies in Complexity. Princeton University Press, 2005.

[40] A. Wagner. Robustness, evolvability, and neutrality. *FEBS Lett*, 579(8):1772–1778, 2005.

[41] E. D. Weinberger. Local properties of kauffman's nk model, a tuneably rugged energy landscape. *Physical Review A*, 44:6399–6413, 1991.

[42] T. Weise and K. Geihs. Dgpf – an adaptable framework for distributed multi-objective search algorithms applied to the genetic programming of sensor networks. In Filipič and Šilc, editors, *2nd Int. Conf. on Bioinspired Optimization Methods and their Application*, pages 157–166. Jožef Stefan Institute, 2006.

[43] T. Weise, M. Zapf, and K. Geihs. Rule-based genetic programming. In *2nd Int. Conf. on Bio-Inspired Models of Network, Information, and Computing Systems*. IEEE Press, 2007.

[44] S. Wright. The roles of mutation, inbreeding, crossbreeding and selection in evolution. In *6th Annual Congress of Genetics*, volume 1, pages 356–366, 1932.