UNIVERSITY OF DORTMUND

REIHE COMPUTATIONAL INTELLIGENCE

COLLABORATIVE RESEARCH CENTER 531

Design and Management of Complex Technical Processes and Systems by means of Computational Intelligence Methods

On the Impact of Objective Function Transformations on Evolutionary and Black-Box Algorithms

Tobias Storch

No. CI-193/05

Technical Report ISSN 1433-3325 April 2005

Secretary of the SFB 531 · University of Dortmund · Dept. of Computer Science/XI
44221 Dortmund · Germany

This work is a product of the Collaborative Research Center 531, "Computational Intelligence," at the University of Dortmund and was printed with financial support of the Deutsche Forschungsgemeinschaft.

On the Impact of Objective Function Transformations on Evolutionary and Black-Box Algorithms[*]

Tobias Storch

Department of Computer Science II
University of Dortmund
44221 Dortmund, Germany
e-mail: tobias.storch@uni-dortmund.de

Abstract

Different fitness functions describe different problems. Hence, certain fitness transformations can lead to easier problems although they are still a model of the considered problem. In this paper, the class of neutral transformations for a simple rank-based evolutionary algorithm (EA) is described completely, i.e., the class of functions that transfers easy problems for this EA in easy ones and difficult problems in difficult ones. Moreover, the class of neutral transformations for this population-based EA is equal to the black-box neutral transformations. Hence, it is a proper superset of the corresponding class for an EA based on fitness-proportional selection, but it is a proper subset of the class for random search. Furthermore, the minimal and maximal class of neutral transformations is investigated in detail.

1 Introduction

Evolutionary algorithms (EAs) belong to the broad class of general randomized search heuristics. Their area of application is as huge as their variety and they have been applied successfully in numerous situations. EAs are population-based optimizers. Here, we consider the maximization of objective (fitness) functions $f : S \to \mathbb{Z}$, where S is a discrete search space. In particular often pseudo-Boolean functions are investigated, where $S = \{0, 1\}^n$.

Let us survey arbitrary transformations $g : \mathbb{Z} \to \mathbb{Z}$ applied to an objective function f. If algorithm A optimizes f efficiently, then A will also be efficient on $g \circ f$ for some transformations g. We will specify the definition of efficiency later.

[*]This work was supported by the Deutsche Forschungsgemeinschaft (DFG) as a part of the Collaborative Research Center “Computational Intelligence” (SFB 531).

In such a situation, we call g *neutral* for f with A. But on the other hand some transformations may turn the algorithm to inefficiency in maximization. This separation strongly depends on the specific algorithm and the specific function. We are interested in transformations g which are neutral for all functions f with respect to A and we call g *neutral* for A, in this situation. This separation depends on the specific algorithm only. Such a classification can assist to categorize a problem as being efficiently optimizable more easily and to generalize obtained results on a specific function to classes of functions. The last topic leads directly to considerations concerning robustness aspects of algorithms. If a function h can be deconstructed into a function f, where A is efficient on, and a neutral transformation g for A, then A is also efficient in optimizing $h = g \circ f$. The investigations of such structural properties lead to a better understanding of the considered algorithms and its operators.

The probably best-known types of EAs – all incorporating aspects of natural selection – are genetic algorithms and evolution strategies. For an overview of common EAs and operators in EAs see e.g., [1]. At first, we consider a simple steady-state $(\mu{+}1)$ EA$^{\text{rank}}$ applying uniform selection (which is a *rank*-based selection scheme), standard-bit mutation, and elitism for deletion. We will observe that the class of neutral transformations $\mathcal{T}_{\text{rank}}$ for the $(\mu{+}1)$ EA$^{\text{rank}}$ consists of all *truncated strictly increasing* functions. This even holds for all rank-based algorithms, whose main property is that their behavior does not depend on the specific fitness value. We will give precise definitions later on.

Definition 1. Truncated Strictly Increasing (t.s.i.)
A function $g : \mathbb{Z} \to \mathbb{Z}$ is called truncated strictly increasing *if there is $z(g) \in \mathbb{Z} \cup \{-\infty, \infty\}$ such that $g(v) < g(v+1)$ for all $v < z(g)$ and $g(v) = g(v+1)$ for all $v \geq z(g)$.*

A t.s.i. function is strictly increasing up to some value and then constant – including the strictly increasing and the constant functions as well. In order to prove rigorously that the class $\mathcal{T}_{\text{rank}}$ cannot be extended for the $(\mu{+}1)$ EA$^{\text{rank}}$ we will present for each not t.s.i. transformation g a class of pseudo-Boolean functions $\mathcal{F}$, where the $(\mu{+}1)$ EA$^{\text{rank}}$ is efficient on every $f \in \mathcal{F}$. But, only on an exponentially small fraction of $g \circ \mathcal{F} := \{g \circ f \mid f \in \mathcal{F}\}$ it is not totally inefficient. Moreover, even every algorithm is inefficient on this class of functions. We will specify how every algorithm can be investigated in the following paragraph. Furthermore, for the $(\mu{+}1)$ EA$^{\text{prop}}$ which is similar to the $(\mu{+}1)$ EA$^{\text{rank}}$ but comes up with fitness-*prop*ortional selection we will prove that $\mathcal{T}_{\text{prop}} \subset \mathcal{T}_{\text{rank}}$ holds, where $\mathcal{T}_{\text{prop}}$ is the class of neutral transformations for the $(\mu{+}1)$ EA$^{\text{prop}}$. This will be shown by presenting a particular pair of a function and a t.s.i. transformation with the proposed properties. We remark that the $(\mu{+}1)$ EA$^{\text{rank}}$ and the $(\mu{+}1)$ EA$^{\text{prop}}$ are quite successful on a wide range of problems even with $\mu = 1$ (see e.g., [2] or [9]). A canonical extension of the class of rank-based algorithms are the not fitness-based algorithms, whose main property is that their behavior does not depend on the fitness value at all. We will make this more precise later, too. Hence, we will prove that for every not fitness-based algorithm the class of neutral transformations $\mathcal{T}_{\text{not fitness}}$ contains all *increasing* functions, i.e., $\mathcal{T}_{\text{rank}} \subset \mathcal{T}_{\text{not fitness}}$.

Definition 2. Increasing
A function $g : \mathbb{Z} \to \mathbb{Z}$ is called increasing *if for every v holds $g(v) \leq g(v+1)$.*

Moreover, for the well-known random search algorithm $\mathrm{A}^{\text{random}}$ that is not fitness-based we will identify the increasing functions as the only neutral transformations. Furthermore, the question arises which transformations are neutral for every algorithm. We will show that this class $\mathcal{T}_{\text{ids}}$ consists of all *truncated identities.*

Definition 3. Truncated Identity (t.i.)
A function $g : \mathbb{Z} \to \mathbb{Z}$ is called truncated identity *if there is $z(g) \in \mathbb{Z} \cup \{-\infty, \infty\}$ such that $g(v) = v$ for all $v < z(g)$ and $z(g) \leq g(v) = g(v+1)$ for all $v \geq z(g)$.*

A t.i. is the identity up to some value and then constant – including the identity and the constant functions as well. For a particular algorithm A^{ids} we will prove that the class of neutral transformations consists of the t.i. only. Furthermore, for an algorithm A^{all} we will demonstrate the other extreme. The class of neutral transformations for this algorithm $\mathcal{T}_{\text{all}}$ consists of all transformations.

One characteristic of all EAs and even of all general search heuristics is that they gather information about the problem instance – the problem itself is known in advance – by querying one search point after the other to a so-called *black-box.* At time t the next query x_t is determined by a so-called *black-box algorithm* with knowledge about the whole history. The history consists of the pairs of previously queried elements and their function values $(x_1, v_1), \ldots, (x_{t-1}, v_{t-1})$. If all black-box algorithms are investigated, then we obtain general lower bounds on the complexity of a problem. In contrast to considerations of a single EA it is not meaningful to investigate the black-box complexity for a single function f. In such a scenario there exists always an efficient black-box algorithm. It just queries an optimal element of f within the first step. Therefore, classes of functions are considered, where the particular function f is randomly chosen from. This theoretical framework was introduced by Droste, Jansen, and Wegener [3]. The evaluation of the black-box typically requires most resources during an optimization step. Therefore, we neglect all additional calculations performed by a black-box algorithm or an EA in particular and identify the number of steps as performance measure only.

Let us estimate the efficiency of a randomized algorithm now. Therefore, let T_{A,f_n} denote a random variable which describes the number of function evaluations until the algorithm A first queries an optimal search point of the pseudo-Boolean function $f_n : \{0,1\}^n \to \mathbb{Z}$, $n \in \mathbb{N}$. If the expectation of T_{A,f_n} for a sequence of functions $f = (f_1, \ldots, f_n, \ldots)$ is upper bounded by a polynomial in n we call A *efficient* on f and *highly inefficient* if it is at least exponentially lower bounded. Moreover, we call A *totally inefficient* on f if the probability that an optimum has been queried is exponential small even after an exponential number of steps. In this case, a polynomially bounded number of (parallel) (independent) multistarts of A is still totally inefficient.

Whenever we consider transformations g we also have to distinguish whether the black-box algorithm has *access* to g or not. Access means that the algorithm is able to evaluate $g(v)$ for all $v \in \mathbb{Z}$. Again, we will first identify the neutral transformations $\mathcal{T}_{\text{black}}$ for all black-box algorithms with access to the specific transformation. I.e., if for a class of functions $\mathcal{F}$ an efficient black-box algorithm exists, then there is also an efficient black-box algorithm *with access* to $g \in \mathcal{T}_{\text{black}}$ for $g \circ \mathcal{F}$. Surprisingly, it holds $\mathcal{T}_{\text{black}} = \mathcal{T}_{\text{rank}}$, where the $(\mu{+}1)$ EA$^{\text{rank}}$ *cannot access* the transformation. Hence, we remark that even $\mathcal{T}_{\text{black}} \subset \mathcal{T}_{\text{not fitness}}$ is no contradiction to the observation that the class of objective functions with an efficient black-box algorithm is larger than the corresponding class for the not fitness-based algorithms. Finally, we will demonstrate that for black-box algorithms their access to the transformation can be essential.

Let us summarize. We will especially show the hierarchy:

$$\mathcal{T}_{\text{ids}} \subseteq \mathcal{T}_{\text{prop}} \subset \mathcal{T}_{\text{rank}} = \mathcal{T}_{\text{black}} \subset \mathcal{T}_{\text{not fitness}} \subset \mathcal{T}_{\text{all}}$$

The classes $\mathcal{T}_{\text{ids}}$ and $\mathcal{T}_{\text{all}}$ even represent the two extreme cases of the minimal and maximal class of transformations. With exception of $\mathcal{T}_{\text{prop}}$ the classes of neutral transformations are specified completely and example algorithms are presented and analyzed for all investigated classes.

The investigations concerning black-box algorithms help us to define rank-based and not fitness-based algorithms now. For history $(x_1, v_1), \ldots, (x_{t-1}, v_{t-1})$ let $\text{rank}_t(v_i) := |\{v_j \mid v_j \leq v_i, 1 \leq j \leq t-1\}|$ denote the *rank* of v_i at step t. Let $p_t^{(0)}(x)$ and $p_t^{(1)}(x)$ denote the probability that the considered algorithm creates the element x in step t with the histories $(x_1, v_1^{(0)}), \ldots, (x_{t-1}, v_{t-1}^{(0)})$ and $(x_1, v_1^{(1)}), \ldots, (x_{t-1}, v_{t-1}^{(1)})$ respectively. The two histories consist of equal sequences of search points but not necessarily of equal sequences of objective function values.

Definition 4. Rank-Based and **Not Fitness-Based**
If $p_t^{(0)}(x) = p_t^{(1)}(x)$ *holds for all* $t \geq 1$ *and for all* x*, where for all* $1 \leq i \leq t-1$ *it is* $\text{rank}_t(v_i^{(0)}) = \text{rank}_t(v_i^{(1)})$*, then the algorithm is called* rank-based.
If $p_t^{(0)}(x) = p_t^{(1)}(x)$ *holds for all* $t \geq 1$ *and for all* x*, then the algorithm is called* not fitness-based.

E.g., the $(\mu{+}1)$ EA$^{\text{rank}}$ is a rank-based algorithm and fitness-based, while the $(\mu{+}1)$ EA$^{\text{prop}}$ is fitness-based only.

We remark that the investigations made here are similar for other discrete search spaces, for minimization, and for objective functions with the decision space $\mathbb{R}$.

Throughout the paper let 0^i and 1^i, $i \geq 0$, denote the bitstrings which consist of i zeros and ones respectively. Moreover, let $|x|$ denote the L_1-norm of $x = x_1 \cdots x_n \in \{0,1\}^n$, i.e., $|x| := \sum_{i=1}^n x_i$, and let $\text{num}(x) := \sum_{i=1}^n x_i 2^{i-1} \in \{0, \ldots, 2^n - 1\}$.

The paper is structured as follows. We begin with the proposed results for rank-based EAs including not fitness-based EAs and the $(\mu{+}1)$ EA$^{\text{rank}}$ in particular in Section 2. The results concerning the (μ+1) EA$^{\text{prop}}$ are considered in Section 3. Afterwards, we investigate the A$^{\text{ids}}$ and the A$^{\text{all}}$ in Section 4. We continue with the considerations of black-box algorithms in Section 5 before we finish with some conclusions in Section 6.

2 Rank-Based EAs

Beside uniform selection also linear-ranking, tournament selection, and truncation selection are well-known rank-based selection schemes. In Section 2.1 we will discuss the results concerning t.s.i. transformations on rank-based EAs in general and in Section 2.2 we will define the (μ+1) EA$^{\text{rank}}$ in detail and we will present the results concerning not t.s.i. transformations and the (μ+1) EA$^{\text{rank}}$. Moreover, in Section 2.3 the not fitness-based algorithms including A$^{\text{random}}$ are investigated in detail. The extensions concerning the black-box algorithms will be proven in Section 5.

2.1 T.S.I. Transformations

Let us first investigate objective functions $f : S \to \mathbb{Z}$, where S is a discrete search space. In the following let the A$^{\text{rank}}$ be an arbitrary rank-based algorithm operating on S.

Theorem 5. *Let $p_{t,f}$ be the probability that the* A$^{\text{rank}}$ *needs more than $t \geq 1$ steps to optimize $f : S \to \mathbb{Z}$. If $g : \mathbb{Z} \to \mathbb{Z}$ is t.s.i., then the* A$^{\text{rank}}$ *needs with a probability of $p_{t,g\circ f} \leq p_{t,f}$ more than t steps to optimize $g \circ f$.*

Proof. Let $S_f \subset S$ and $S_{g\circ f} \subset S$ contain all non-optimal elements of S with respect to f and $g \circ f$ respectively. Since each optimal element of f is also one of $g \circ f$ for sure it holds $S_f \supseteq S_{g\circ f}$. At step $t \geq 1$ the A$^{\text{rank}}$ on f has queried the sequence (s_t) of search points $x_1, \ldots, x_t$ with probability $p_{(s_t),f}$. Hence, $p_{t,f}$ is the sum of the $p_{(s_t),f}$ for all sequences (s_t) of length t which do not contain an optimal element, i.e., $x_i \in S_f$ for all $1 \leq i \leq t$. Each such sequence (s_t) is either optimal for the A$^{\text{rank}}$ on $g \circ f$, i.e., $g \circ f(x_i) \in S - S_{g\circ f} \supseteq S - S_f$ for some $1 \leq i \leq t$, or the A$^{\text{rank}}$ on $g \circ f$ queries (s_t) with probability $p_{(s_t),f}$, too. The last argument holds since the A$^{\text{rank}}$ is rank-based and therefore, in this situation and by definition, the algorithm has generated with equal probabilities the search point x_j on the histories $(x_1, f(x_1)), \ldots, (x_{j-1}, f(x_{j-1}))$ and $(x_1, g \circ f(x_1)), \ldots, (x_{j-1}, g \circ f(x_{j-1}))$. This holds for every $1 \leq j \leq t$. We remember that for $x_i \in S_{g\circ f}$, $1 \leq i \leq j-1$, it holds $\text{rank}_j(f(x_i)) = \text{rank}_j(g \circ f(x_i))$. $\square$

The converse is typically incorrect. Therefore, consider an arbitrary function f, where the investigated algorithm is inefficient on. (E.g., every general search heuristic is inefficient on PLATEAU, see Lemma 23 in Section 5.) But the function $g \circ f$, where the t.s.i. transformation g is an arbitrary constant function (e.g., $g(v) = 0$ for all v) will be optimized by every algorithm within the first query since there exist optimal elements only. We remark that this also holds for black-box algorithms.

2.2 Not T.S.I. Transformations

For a simple and typical rank-based EA, the (μ+1) EA$^{\text{rank}}$, we will prove rigorously that the class of neutral transformations consists of all t.s.i. functions. The lower bound was shown in the previous section. For the upper bound it is sufficient to present for each not t.s.i. transformation a class of functions, where the EA turns from being efficient to inefficiency. Here, even a turnover to total inefficiency can be demonstrated. Let us assume without simplifying the setting that the search space S equals $\{0,1\}^n$, i.e., we investigate pseudo-Boolean functions. Moreover, the investigated mutation-based steady-state (μ+1) EA$^{\text{rank}}$ is defined as follows (see [7]).

Algorithm 6. (μ+1) EA$^{\text{rank}}$

1. Choose μ different individuals $x_i \in \{0,1\}^n$, $1 \le i \le \mu$, uniformly at random. These individuals constitute the population $\mathcal{P}$, i.e., $\mathcal{P} := \{x_1, \ldots, x_\mu\}$.
2. Choose an individual x from the population $\mathcal{P}$ uniformly at random. Create y by flipping each bit in x independently with probability $1/n$.
3. If $y \notin \mathcal{P}$, i.e., $y \neq x_i$ for all $1 \le i \le \mu$, then let $z \in \mathcal{P} \cup \{y\}$ be randomly chosen among those individuals with the worst f-value and let the population be $\mathcal{P} \cup \{y\} - \{z\}$, goto 2., and else let the population be $\mathcal{P}$, goto 2.

The population merely consists of different elements which implies that the structure of the population is not only a multiset, but a set. Hence, only choices $\mu \le 2^n$ are feasible. In the rest of the paper, let the size of the population μ be polynomially bounded. Otherwise, even the initialization of the algorithm implicates inefficiency.

Now, we will demonstrate the possible turnover to total inefficiency of the (μ+1) EA$^{\text{rank}}$, if an arbitrary not t.s.i. transformation is applied.

Theorem 7. *If $g : \mathbb{Z} \to \mathbb{Z}$ is not t.s.i., then there exist classes of functions $\mathcal{F}$, where the* (μ+1) EA$^{\text{rank}}$ *needs for every $f \in \mathcal{F}$ an expected polynomial number of steps for optimization (in particular $\mathcal{O}(\mu n^3)$). The* (μ+1) EA$^{\text{rank}}$ *needs for all but an exponential small fraction of functions $f \in \mathcal{F}$, i.e., $2^{-\Omega(n)}$, with an exponential small failure probability an exponential number of steps to optimize $g \circ f$. Every black-box algorithm needs at least an expected exponential number of steps to optimize $g \circ f$, $f \in \mathcal{F}$.*

Proof. We will present for each function g a class of functions $\mathcal{F}$ with the proposed properties. The results concerning black-box algorithms follow directly by Lemma 23 in Section 5.

We will distinguish three cases. The first case considers the situation that there exists at least one integer which is transformed to a larger value than its successor (Case 1). Afterwards, we just have to investigate functions, where $g(v) \le g(v+1)$ holds for all $v \in \mathbb{Z}$ and furthermore, for at least one integer

v holds $g(v) < g(v+1)$ (otherwise, g would be t.s.i. with $z(g) = -\infty$). We distinguish two further cases – for technical reasons only. One case considers the situation that there are enough integers (what enough means will be specified later) which are transformed to the same non-optimal value (Case 2). To be more precise, for some ℓ_2 large enough there are at least $\ell_2 - 1 < \infty$ integers $v_1 < \cdots < v_{\ell_2-1}$, where $g(v_i) = g(v_{i+1})$, $1 \leq i < \ell_2 - 1$, holds and at least one more integer $v_{\ell_2} > v_{\ell_2-1}$, where $g(v_{\ell_2-1}) < g(v_{\ell_2})$ holds. If this is not the case, then there are at least two integers $v_{\ell_3-1} < v_{\ell_3}$ transformed to the same non-optimal value $g(v_{\ell_3-1}) = g(v_{\ell_3})$ and since we are not in the second case and there are infinite many integers smaller than v_{ℓ_3-1} there exist at least $\ell_3 - 2 < \infty$ more integers $v_1 < \cdots < v_{\ell_3-2} < v_{\ell_3-1}$, where $g(v_i) < g(v_{i+1})$, $1 \leq i < \ell_3 - 1$, holds. I.e., there are enough integers (what enough means will be specified later) transformed to different values (Case 3).

Case 1. There is an integer v, where $g(v) > g(v+1)$ holds. Then, we investigate the class of functions $\mathcal{F}_1$ that consists of all functions $f_{1,a} : \{0,1\}^n \to \mathbb{Z}$, where for $a \in \{0,1\}^n$

$$f_{1,a}(x) := \begin{cases} v & \text{if } x = a \\ v+1 & \text{otherwise} \end{cases} .$$

Each function contains one non-optimal element only. Hence, it is even not probable to find this element instead of an optimal one. After applying the transformation the algorithm has to search for the needle in the haystack.

The expected number of steps of the (μ+1) EA$^{\text{rank}}$ on $f_{1,a}$ is bounded by $\mathcal{O}(1)$. If $\mu = 1$ and the initial element is even the single non-optimal element a, then with a probability of $\sum_{i=1}^{n} \binom{n}{i} 1/n^i (1-1/n)^{n-i} = 1 - (1-1/n)^n \geq 1 - 1/e$ an optimum is created since an arbitrary mutation changing at least one bit has to be performed to generate an optimum. If $\mu \geq 2$, then at the latest the second element of the initialization is an optimum.

Let us consider the class of functions $g \circ \mathcal{F}_1$ now. Therefore, let $h : \{0,1\}^n \to \mathbb{Z}$ be the constant function $h(x) := g(v+1)$ for all $x \in \{0,1\}^n$. We will investigate the (μ+1) EA$^{\text{rank}}$ on h, where an optimum of h is created within the first step. Moreover, let $p_t(x)$, $t \geq 1$, denote the probability that the (μ+1) EA$^{\text{rank}}$ on h creates the element x in step t and let Q denote the set of elements which have a probability of at least $2^{-n/3}$ to be generated at least once within the first $\lfloor 2^{n/3} \rfloor$ steps of the (μ+1) EA$^{\text{rank}}$. It holds $|Q| \leq \lceil 2^{2n/3} \rceil$. Therefore, we assume $|Q| \geq \lceil 2^{2n/3} \rceil + 1$ which leads to a contradiction since $(\lceil 2^{2n/3} \rceil + 1) 2^{-n/3} > \lfloor 2^{n/3} \rfloor$ and the (μ+1) EA$^{\text{rank}}$ has created at most $\lfloor 2^{n/3} \rfloor$ different elements for the first time within the first $\lfloor 2^{n/3} \rfloor$ steps. Hence, at least as long as the optimum of $g \circ f_{1,a}$, $f_{1,a} \in \mathcal{F}_1$, was not created the next query is chosen with equal probability for the (μ+1) EA$^{\text{rank}}$ on h and $g \circ f_{1,a}$. The probability that the optimum is created within the first $\lfloor 2^{n/3} \rfloor$ steps is bounded by $2^{-n/3}$ for all functions $g \circ f_{1,a}$, where $a \notin Q$. Moreover, Q represents only an exponential small fraction of all investigated functions since $|Q|/|\mathcal{F}_1| \leq \lceil 2^{2n/3} \rceil / 2^n = 2^{-\Omega(n)}$.

Case 2. There exist at least n integers $v_0 < \cdots < v_{n-1}$, where $g(v_i) = g(v_{i+1})$, $0 \leq i < n-1$, holds and at least one further integer $v_n > v_{n-1}$, where $g(v_{n-1}) <$

$g(v_n)$ holds. Then, we investigate the class of functions $\mathcal{F}_2$ which consists of all functions $f_{2,a} : \{0,1\}^n \to \mathbb{Z}$, where for $a \in \{0,1\}^n$

$$f_{2,a}(x) := v_{n-\mathbf{H}(x,a)} \quad .$$

Let $\mathbf{H}(x,a)$ denote the Hamming distance of x and a. The fitness increases with a decrease of the Hamming distance up to the optimum. This makes it easy for the (μ+1) EA$^{\text{rank}}$ to optimize the function. After applying the transformation the algorithm has to search for the needle in the haystack.

The expected number of steps of the (μ+1) EA$^{\text{rank}}$ on $f_{2,a} \in \mathcal{F}_2$ is bounded by $\mathcal{O}(\mu n \log n)$. The function $f_{2,a}$ has the same properties than the well-known function OneMax, OneMax$(x) := |x|$. A proof that the expected number of steps of the (μ+1) EA$^{\text{rank}}$, where $\mu = 1$, on OneMax equals $\Theta(n \log n)$ can be found in [2]. Let x denote an element with the current largest function value $v_{n-\mathbf{H}(x,a)}$. For all non-optimal elements x (has a probability of $1/\mu$ to be selected for mutation) at least $\mathbf{H}(x,a)$ special 1-bit mutations (has a probability of $\mathbf{H}(x,a)/n(1-1/n)^{n-1} \geq \mathbf{H}(x,a)/(en)$ to be performed) increase the function value. Moreover, at the beginning the largest function value is at least v_0. Hence, we can bound the expected number of steps of the (μ+1) EA$^{\text{rank}}$ by $\sum_{i=1}^{n} e\mu n/i = \mathcal{O}(\mu n \log n)$ to optimize $f_{2,a}$.

Let us consider the class of functions $g \circ \mathcal{F}_2$ now. This class of functions is similar to the one in the first case. Moreover, the same arguments lead to the proposed result.

Before we investigate the completing case we will analyze the behavior of the (μ+1) EA$^{\text{rank}}$ on plateaus first. Therefore, let $\text{SP}_a : \{0,1\}^n \to \mathbb{Z}$, where (SP stands for SmallPlateau)

$$\text{SP}_a(x) := \begin{cases} 2 & \text{if } x = p_n \\ 1 & \text{if } x = p_i,\ 0 \leq i < n, \text{ and } p_i \neq p_n \\ 0 & \text{otherwise} \end{cases}$$

for $a = a_1 \cdots a_n \in \{0,1\}^n$ with $p_i = a_1 \cdots a_i 0^{n-i}$, $0 \leq i \leq n$, and let $\text{LP}_{\epsilon,a} : \{0,1\}^n \to \mathbb{Z}$, where (LP stands for LargePlateau)

$$\text{LP}_{\epsilon,a}(x) := \begin{cases} 2 & \text{if } x = a \\ 1 & \text{if } x \in \{1^{n-\lceil \epsilon n \rceil} y \mid y \in \{0,1\}^{\lceil \epsilon n \rceil}\} - \{a\} \\ 0 & \text{otherwise} \end{cases}$$

for a constant $0 < \epsilon \leq 1$ and $a \in \{1^{n-\lceil \epsilon n \rceil} y \mid y \in \{0,1\}^{\lceil \epsilon n \rceil}\}$.

A (connected) subset of elements which all have the same function value is called a *plateau.* A sequence $s_0, \ldots, s_r$, $r \geq 0$, of elements is called a *path* (of constant fitness) if the Hamming distance of s_i and s_{i+1} is (at most) one for $0 \leq i < r$ (and the elements constitute a plateau).

Lemma 8. *Let the population consist of at least one element with fitness* 1 *and let a be chosen arbitrarily. The expected number of steps until the* (μ+1) EA$^{\text{rank}}$

on SP_a *creates the optimum is upper bounded by* $\mathcal{O}(\mu n^2)$ *if* $\mu > n$ *and by* $\mathcal{O}(\mu n^3)$ *if* $\mu \le n$*. For* $\mathrm{LP}_{\epsilon,a}$ *and the* $(\mu{+}1)$ $\mathrm{EA}^{\mathrm{rank}}$ *the expected number of steps is upper bounded by* $\mathcal{O}(\mu 2^{\epsilon n})$*.*

Proof. In order to simplify the notation we first define the $(1{+}1)$ $\mathrm{EA}^\star$ to be the $(\mu{+}1)$ $\mathrm{EA}^{\mathrm{rank}}$, where $\mu = 1$, but it accepts elements with the same fitness for sure (and not with probability $1/2$ only if the offspring is not a duplicate). The $(1{+}1)$ $\mathrm{EA}^\star$ is well-studied by, e.g., [2], [4], [5], or [9].

Let us first investigate the $(\mu{+}1)$ $\mathrm{EA}^{\mathrm{rank}}$ on SP_a. As long as the optimum is not generated or it exists an element outside the plateau (which is also a path) there is at least one element in the population (has a probability of $1/\mu$ to be selected for mutation) with the following property. At least one special 1-bit mutation (has a probability of $1/n(1-1/n)^{n-1} \ge 1/(en)$ to be performed) of this individual creates either the optimum or at least a plateau element which is not contained in the population. Moreover, such an offspring replaces an element outside the plateau. Hence, the expected number of steps for such an event is upper bounded by $e\mu n$. Since the length of the path is at most n and at the beginning there exists at least one plateau element in the population within an expected number of at most $n \cdot e\mu n = \mathcal{O}(\mu n^2)$ steps either the whole population consists of different plateau elements or the optimum is created. The optimum is generated for sure if $\mu > n$. For this case, the first part of the result follows.

In the following let $\mu \le n$. We reinterpret how the descendant population is determined. This will help us to simplify the argumentation but it does not modify the behavior of the algorithm. If a plateau element is created we assume that this element stays in the population for sure and a duplicate if one exists is deleted or an individual of the parent population chosen uniformly at random otherwise. We remark that if an element with function value 0 is created, then such an offspring will directly be removed and if the optimum is generated we have successfully finished.

The following analysis is similar to the one of Witt [8] for a similar situation but a different algorithm. Therefore, we recall the definition of *family trees*. A family tree $T_t(x)$ of an individual x at time $t > 1$ contains $T_{t-1}(x)$ (a directed and labeled tree) and the edge (v, w) if w is the result of a mutation of the individual $v \in T_{t-1}(x)$. The tree $T_1(x)$ contains the root x only. Thus, the individuals of the first investigated (typically initial) population constitute the roots of the family trees. Since the $(\mu{+}1)$ $\mathrm{EA}^{\mathrm{rank}}$ describes an infinite stochastic process the growing process of the family trees is infinite as well (at least for one individual of the first investigated population). At some time $t \ge 1$ we call a *route* from the root x to the node y in a family tree $T_t(x)$ *dead* if y has been deleted by time t and *alive* otherwise. Hence, there exists at least one route which is alive in at least one family tree. Let us consider such a route $x_1, \ldots, x_{k+1}$ of length $k \ge 0$ which does not reach the optimum. There are plateau elements only on this route. The $(\mu{+}1)$ $\mathrm{EA}^{\mathrm{rank}}$ defines a distribution over all family trees at each time t and therefore, the distribution over such routes as well. Furthermore, we investigate the corresponding distribution of the (1+1) EA* beginning with x_0. By induction it follows that both distributions on the elements of such a route are the same. Therefore, we remember that if a specific individual (respectively a node in a family tree) is selected, then the mutation operators and the acceptance behaviors are the same for the (μ+1) EA$^{\text{rank}}$ and the (1+1) EA*. This holds especially since the behavior of the mutation operator and the acceptance is independent of the current time and the other individuals. Here, we have to remember that all investigated individuals have the same function value and that we have reinterpreted how the descendant population is determined. We observe that in such a situation, the behavior is even the same if the optimum is created.

Let us now prove that the expected number of steps until a route which is alive reaches length $k \geq 0$ is upper bounded by $2e\mu k$ or the optimum is created. In the second case, nothing has to be shown. In the first case, in order to lengthen such a route and to keep it alive it is sufficient to select the individual on the route with maximal distance to the root (has a probability of $1/\mu$). Afterwards, this individual has to create an arbitrary element on the plateau (has a probability of at least $(1-1/n)^n \geq 1/(2e)$ since a 0-bit mutation creates a duplicate) or the optimum. Moreover, the parent has to be deleted prior to its offspring (has a probability of 1 in the particular investigated situation, where a duplicate is created – otherwise it equals 1/2). Hence, the expected number of steps to increase the length of a route is upper bounded by $2e\mu$.

Jansen and Wegener [5] have especially proven that the (1+1) EA* needs an expected number of at most $\mathcal{O}(n^3)$ steps to create the optimum of SP$_a$ for every a and starting with an arbitrary element on the path. An application of Markov's inequality (see e.g., [6]) shows that within cn^3 steps for a large enough constant $c > 0$ with a probability of at least 1/2 the optimum is created by the (1+1) EA*. Hence, within an expected number of at most $2e\mu cn^3$ steps the (μ+1) EA$^{\text{rank}}$ creates the optimum. In the case of a failure we can repeat the argumentation. The expected number of such repetitions is bounded by 2. This upper bounds the expected number of steps of the (μ+1) EA$^{\text{rank}}$, where $\mu \leq n$, on SP$_a$ by $4e\mu cn^3 = \mathcal{O}(\mu n^3)$ in total.

Let us now investigate the (μ+1) EA$^{\text{rank}}$ on LP$_{\epsilon,a}$, where at the beginning exists at least one plateau element in the population. We can apply the same proof technique as for SP$_a$. Within an expected number of at most $e\mu^2 n$ steps either the optimum is created and the (μ+1) EA$^{\text{rank}}$ has successfully finished or the whole population consists of plateau elements only. Therefore, at most $\mu - 1$ special 1-bit mutations of plateau elements with the following property are sufficient. For this element exists at least one plateau element with Hamming distance one which is not contained in the population. Moreover, the distribution of the elements on a route in a family tree is again the same than for the (1+1) EA*. Furthermore, the expected number of steps to increase the length of a route is also upper bounded by $2e\mu$. Garnier, Kallel, and Schoenauer [4] have especially proven that the (1+1) EA* needs an expected number of at most $\mathcal{O}(2^{\epsilon n})$ steps to create the optimum of LP$_{\epsilon,a}$ for every a and starting with an arbitrary element on the plateau. With the same arguments as for SP$_a$ this upper bounds the expected number of steps of the (μ+1) EA$^{\text{rank}}$ on LP$_{\epsilon,a}$ by

$e\mu^2 n + \mathcal{O}(\mu 2^{\epsilon n}) = \mathcal{O}(\mu 2^{\epsilon n})$ since ϵ is a positive constant.

Altogether, we have shown the proposed result. $\square$

We are now able to prove the completing case.

Case 3. There exist at least n integers $v_1 < \cdots < v_n$, where $g(v_i) < g(v_{i+1})$, $1 \leq i < n$, at least one integer $v_{n+1} > v_n$, where $g(v_n) = g(v_{n+1})$, and at least one more integer $v_{n+2} > v_{n+1}$, where $g(v_{n+1}) < g(v_{n+2})$ holds. Then, we investigate the class of functions $\mathcal{F}_3$ that consists of all functions $f_{3,a} : \{0,1\}^n \to \mathbb{Z}$, where

$$f_{3,a}(x) := \begin{cases} v_{n+2} & \text{if } x = p_n \\ v_{n+1} & \text{if } x = p_i,\ 0 \leq i < n, \text{ and } p_i \neq p_n \\ v_n & \text{if } x \in T \\ v_{|x|} & \text{otherwise} \end{cases}$$

for $a = a_1 \cdots a_n \in \{1^{n-\lceil n/20 \rceil} y \mid y \in \{0,1\}^{\lceil n/20 \rceil}\}$ with $p_i := a_1 \cdots a_i 0^{n-i}$, $0 \leq i \leq n$, and $T := \{1^{n-\lceil n/20 \rceil} y \mid y \in \{0,1\}^{\lceil n/20 \rceil}\} - \{p_0, \ldots, p_n\}$ (Trap). The sequence $p_0, \ldots, p_{n-1}$ describes a path of constant fitness, where some elements can be equal. A sequence $s_0, \ldots, s_r$ is called a *path* if the Hamming distance of s_i, $0 \leq i < r$, and s_{i+1} is (at most) one. The beginning of the path will be created probably and quickly by the $(\mu{+}1)$ EA$^{\text{rank}}$. Afterwards, such an element will not be replaced by an element of the plateau T. A (connected) subset of elements which all have the same function value is called a *plateau.* The path guides the $(\mu{+}1)$ EA$^{\text{rank}}$ through the plateau and furthermore, the optimum will be created quickly. The probability to create an element of T prior to an p_i, $0 \leq i \leq n$, is so small that the expected optimization time in these cases does not effect the expected number of steps in total strongly. After applying the transformation the algorithm has to search for the needle in the haystack $T \cup \{p_i \mid 1 \leq i \leq n-1, p_i \neq p_n\}$ which is now smaller than in the previous cases, but still of exponential size.

The expected number of steps of the $(\mu{+}1)$ EA$^{\text{rank}}$ on $f_{3,a} \in \mathcal{F}_3$ is bounded by $\mathcal{O}(\mu n^3)$. The following holds for n large enough. At first, let us neglect the possibility to create an element of T. Within $2e\mu n^2$ steps an element of the path p_i, $0 \leq i \leq n$, is first generated with a probability of at least $1 - 2^{-n/4}$. Moreover, even the expected number of steps for this event is bounded by $\mathcal{O}(\mu n^2)$. This holds similar to the second case since after at most n special 1-bit mutations of an individual with the largest function value an element of the path is generated. The probability for such an event is lower bounded by $1/(e\mu n)$ and the expected number of steps results directly. An application of Chernoff bounds (see e.g., [6]) shows that within $2e\mu n^2$ steps with a probability of at most $2^{-n/4}$ less than n special 1-bit mutations are performed. Generating the population by choosing each element independently and uniformly at random leads with an exponential small probability to a different initial population than creating it by the investigated strategy. This holds since we can upper bound the probability that the selection of μ out of 2^n elements results in the choice of a duplicate by $\sum_{t=1}^{\mu} 1/(2^n - t - 1) \leq 2^{-n/2}$. Moreover, an application of Chernoff bounds shows that the probability to create initially an element with more than $\lceil 3n/4 \rceil$ ones is again upper bounded by $2^{-n/16}$. Hence, the probability that at least one of the initially chosen elements consists of more than $\lceil 3n/4 \rceil$ ones is upper bounded by $2^{-n/2} + \mu 2^{-n/16} \leq 2^{-n/17}$. Apart from elements of $T \cup \{p_i \mid \lceil 3n/4 \rceil + 1 \leq i \leq n\}$, afterwards, only elements with at most $\lceil 3n/4 \rceil$ ones are accepted in the population. Hence, at least $\lceil n/6 \rceil$ bits have to change to create an element of T. The probability to create within $2e\mu n^2$ steps an element of T can be bounded by $2e\mu n^2/n^{n/6} \leq 2^{-n}$. Moreover, we can upper bound the probability to create an element of T prior to an element p_i, $\lceil 3n/4 \rceil + 1 \leq i \leq n$, by $2^{-n/4} + 2^{-n/17} + 2^{-n} \leq 2^{-n/18}$ in total. We recall that an element of T never replaces an element of the path and especially the optimum has a larger function value than every element of T. In this situation, we can bound the expected number of steps by $\mathcal{O}(\mu 2^{n/20})$ until an element p_i, $0 \leq i \leq n$, is created as Lemma 8 shows. Hence, we can upper bound the expected number of steps until an element p_i, $0 \leq i \leq n$, is generated by $(1 - 2^{-n/18})\mathcal{O}(\mu n^2) + 2^{-n/18} \max\{\mathcal{O}(\mu n^2), \mathcal{O}(\mu 2^{n/20})\} = \mathcal{O}(\mu n^2)$ in total. If an element $p_i \neq p_n$, $0 \leq i < n$, is created, then the expected number of steps to create the optimum is bounded by $\mathcal{O}(\mu n^3)$ as Lemma 8 demonstrates again. In summary, this leads to an expected number of $\mathcal{O}(\mu n^3)$ steps on $f_{3,a}$.

Let us consider the class of functions $g \circ \mathcal{F}_3$ now. The function $g \circ f_{3,a}$, $f_{3,a} \in \mathcal{F}_3$, is similar to $f_{3,a}$ but there is one essential exception. All elements $T \cup \{p_i \mid 0 \leq i < n, p_i \neq p_n\}$ have the same non-optimal function value. Therefore, $g \circ \mathcal{F}_3$ is similar to $g \circ \mathcal{F}_1$, but the size of its plateau is $2^{\lceil n/20 \rceil}$ only. Hence, a similar analysis as for $g \circ \mathcal{F}_1$ shows that with exception of an exponential small fraction of $g \circ \mathcal{F}_3$ the (μ+1) EA$^{\text{rank}}$ needs with a failure probability of at most $2^{-n/60}$ more than $\lfloor 2^{n/60} \rfloor$ steps to optimize $g \circ f_{3,a}$. □

2.3 Not Fitness-Based EAs

Let us again investigate functions with an arbitrary discrete search space first. In the following let the A$^{\text{not fitness}}$ be an arbitrary not fitness-based algorithm operating on S.

Theorem 9. *Let $p_{t,f}$ be the probability that the* A$^{\text{not fitness}}$ *needs more than $t \geq 1$ steps to optimize $f : S \to \mathbb{Z}$. If $g : \mathbb{Z} \to \mathbb{Z}$ is increasing, then it needs with a probability of $p_{t,g\circ f} \leq p_{t,f}$ more than t steps to optimize $g \circ f$.*

Proof. The proof is similar to the one of Theorem 5. We have to remember that an optimal element for f is surely one for $g \circ f$. Moreover, for every $t \geq 1$ it holds, if the A$^{\text{not fitness}}$ on f queries a sequence of elements (s_t) with probability $p_{(s_t),f}$, then with probability $p_{(s_t),f}$ the A$^{\text{not fitness}}$ on $g \circ f$ queries (s_t), too. This holds for all sequences (s_t). □

For a simple and well-known not fitness-based algorithm, the random search, we will mention similar to the previous section and also for search space $\{0,1\}^n$ that the class of neutral transformations consists of all increasing functions. The considered algorithm A$^{\text{random}}$ queries in each step a search point chosen independently and uniformly at *random*.

Theorem 10. *If $g : \mathbb{Z} \to \mathbb{Z}$ is not increasing, then there exist classes of functions $\mathcal{F}$, where the* $\mathrm{A}^{\mathrm{random}}$ *needs for every $f \in \mathcal{F}$ an expected number of $1/(1-2^{-n}) = \mathcal{O}(1)$ steps for optimization. The* $\mathrm{A}^{\mathrm{random}}$ *needs for every function $f \in \mathcal{F}$ with an exponentially small failure probability, i.e., $2^{-\Omega(n)}$, at least an exponential number of steps to optimize $g \circ f$. Every black-box algorithm needs at least an expected number of $(2^n+1)/2$ steps to optimize $g \circ f$, $f \in \mathcal{F}$.*

Proof. The result concerning black-box algorithms follows directly by Lemma 23 in Section 5.

There exists at least one integer v, where $g(v) > g(v+1)$ holds, otherwise g would be increasing. Similar as in Theorem 7, we consider the class of functions $\mathcal{F}$ containing all functions $f_a : \{0,1\}^n \to \mathbb{Z}$, where

$$f_a(x) := \begin{cases} v & \text{if } x = a \\ v+1 & \text{otherwise} \end{cases}$$

for $a \in \{0,1\}^n$.

For each $f_a \in \mathcal{F}$, the probability that the $\mathrm{A}^{\mathrm{random}}$ creates an optimal element of f_a equals $1 - 2^{-n}$ in each step.

For each $f_a \in \mathcal{F}$, the probability equals 2^{-n} in each step that the $\mathrm{A}^{\mathrm{random}}$ generates the optimum of $g \circ f_a$. Hence, within $\lfloor 2^{n/2} \rfloor$ steps the probability equals $\lfloor 2^{n/2} \rfloor 2^{-n} \le 2^{-n/2}$ that the optimal element was created. □

3 Not Rank-Based EAs

Beside fitness-proportional selection which is also called roulette wheel selection, Boltzmann selection and stochastic universal sampling are well-known not rank-based selection schemes.

We consider in particular the following simple (μ+1) $\mathrm{EA}^{\mathrm{prop}}$ which is similar to the (μ+1) $\mathrm{EA}^{\mathrm{rank}}$ but comes up with fitness-proportional selection.

Algorithm 11. (μ+1) $\mathrm{EA}^{\mathrm{prop}}$

1. Choose μ different individuals $x_i \in \{0,1\}^n$, $1 \le i \le \mu$, uniformly at random. These individuals constitute the population $\mathcal{P}$, i.e., $\mathcal{P} := \{x_1, \ldots, x_\mu\}$.

2. Choose the individual x from the population $\mathcal{P}$ with a probability of $f(x)/\sum_{x' \in \mathcal{P}} f(x')$. Create y by flipping each bit in x independently with probability $1/n$.

3. If $y \notin \mathcal{P}$, i.e., $y \neq x_i$ for all $1 \le i \le \mu$, then let $z \in \mathcal{P} \cup \{y\}$ be randomly chosen among those individuals with the worst f-value and let the population be $\mathcal{P} \cup \{y\} - \{z\}$, goto 2., and else let the population be $\mathcal{P}$, goto 2.

Obviously, the algorithm operates on functions with positive function values only. Therefore, we investigate objective functions $f : \{0,1\}^n \to \mathbb{Z}^{>0}$. We remark that the ($\mu$+1) EA$^{\text{prop}}$ and the (μ+1) EA$^{\text{rank}}$ are the same for $\mu = 1$.

Let us first investigate the (μ+1) EA$^{\text{prop}}$ and not t.s.i. functions g. The same example functions f with similar proofs than for the (μ+1) EA$^{\text{rank}}$ in Theorem 7 show that g is not neutral for the (μ+1) EA$^{\text{prop}}$ on f. Hence, the (μ+1) EA$^{\text{prop}}$ reaches the same situations as the (μ+1) EA$^{\text{rank}}$ with similar probabilities and within a similar number of steps. In order to show this, the main observation is that the (μ+1) EA$^{\text{prop}}$ and the (μ+1) EA$^{\text{rank}}$ behave equivalently on plateaus. Moreover, the probability to select an element with largest function value for mutation is lower bounded by $1/\mu$ for the (μ+1) EA$^{\text{prop}}$. As proposed, we will prove that a t.s.i. transformation g is not neutral for a particular function f with (μ+1) EA$^{\text{prop}}$. We will develop such a function f.

3.1 First Results

The first function does not satisfy all the proposed properties, but we make observations that help us to identify functions later, where all the desired properties hold.

Let us consider the function PP : $\{0,1\}^n \to \mathbb{Z}^{>0}$, where (PP stands for PathPeak):

$$\text{PP}(x) := \begin{cases} 3n^2 + 2n + i & \text{if } x = 0^{n-i}1^n =: p_i \text{ and} \\ & \qquad i = n - \lceil n/3 \rceil \text{ or } i = n \\ 3n^2 + n + i & \text{if } x = 0^{n-i}1^i =: p_i,\ 0 \le i < n, \\ & \qquad \text{and } i \ne n - \lceil n/3 \rceil \\ 3n^2 + n - |x| & \text{otherwise} \end{cases}$$

Hence, we observe that the decision space consists of the values $\{3n^2, \ldots, 3n^2 + 3n\}$ at most and it is $3n^2 + 3n < 3(n+1)^2$. The function is similar to the one defined by Storch [7]. The element $p_{n-\lceil n/3 \rceil}$ is called a *peak* since an element with at least the same function value has a large Hamming distance. Here, the distance is linear in n. Moreover, for a path $s_0, \ldots, s_r$, $r \ge 0$, and a population $\mathcal{P}$ we call the element $s_{\max\{i \mid s_i \in \mathcal{P}\}}$ the *element with largest index.* The beginning of the path will be created probably and quickly by the (μ+1) EA$^{\text{prop}}$. If the population contains at least two (different) elements and none is optimal, then a 1-bit mutation of the individual with largest index will create an element with an even larger index. Moreover, such an element will also be accepted in the population. This holds since in particular the probability to select the element with largest index for mutation is high. However, let g be defined as follows:

$$g(v) := \begin{cases} v & \text{if } v < 0 \\ v + 2^{n^2} & \text{if } 3n^2 \le v \le 3n^2 + 2n,\ n \ge 0 \\ v + 2^{n^2+n} & \text{if } 3n^2 + 2n < v < 3(n+1)^2,\ n \ge 0 \end{cases}$$

The transformation g is well-defined and t.s.i. since it holds $g(3n^2) = 3n^2 + 2^{(n-1)^2+(n-1)} < 3n^2 + 1 + 2^{n^2} = g(3n^2+1)$ for all $n \geq 1$. For all other values of v this property is obvious. After applying this transformation the function value of the peak (and the optimum) is increased enormously. Hence, the probability not to select the peak for mutation is extremely small, if this individual is contained in the population, but not the optimum. Moreover, the probability to create the optimum by a mutation of the peak is extremely small, too.

Theorem 12. *The expected number of steps until the* $(\mu{+}1)$ EA$^{\text{prop}}$*, where* $\mu \geq 2$*, has optimized* PP *is upper bounded by* $\mathcal{O}(\mu n^2)$*.*

Proof. The proof is similar to the one in [7]. After an expected number of $\mathcal{O}(\mu n^2)$ steps an element p_i, $i > n - \lceil n/3 \rceil$, is included in the population. Therefore, at most $2n - \lceil n/3 \rceil$ special 1-bit mutations of an element with the largest function value creates such an element. Hence, the probability for such a mutation is bounded by $1/(e\mu n)$ since the probability to select an element with the largest function value is at least $1/\mu$ and the probability for a special 1-bit mutation equals $1/n(1-1/n)^{n-1} \geq 1/(en)$. Afterwards, a special 1-bit mutation of the individual with largest index creates an element with an even larger index which will be accepted in the population. This element has either also the largest function value (if $p_{n-\lceil n/3 \rceil}$ is not contained in the population) or the second-largest function value (if $p_{n-\lceil n/3 \rceil}$ is contained in the population). The probability to select the desired individual is lower bounded by $1/(2\mu)$. This holds since for the considered cases the second-largest function value is at least $3n^2 + 2n - \lceil n/3 \rceil + 1$ while the largest function value equals $3n^2 + 3n - \lceil n/3 \rceil$. Hence, the expected number of steps for these at most $\lceil n/3 \rceil$ successes is upper bounded by $\mathcal{O}(\mu n^2)$ and therefore, the expected number of steps in total as well. □

We remark that the upper bound of $\mathcal{O}(\mu n^2)$ expected steps holds also for the $(\mu{+}1)$ EA$^{\text{rank}}$, where $\mu \geq 2$.

Theorem 13. *The probability that the* $(\mu{+}1)$ EA$^{\text{prop}}$ *has optimized* $g \circ \text{PP}$ *within* $2^{\Omega(n)}$ *steps is lower bounded by* $1 - \mathcal{O}(1/n)$*. The expected number of steps is lower bounded by* $2^{\Omega(n)}$*.*

Proof. The following holds for n large enough. We can bound the probability to create the optimum prior to the element $p_{n-\lceil n/3 \rceil}$ by $1 - \mathcal{O}(1/n)$. Therefore, we observe that the probability is bounded by $2^{-\Omega(n)}$ to create initially an element with more than $\lceil 7n/12 \rceil$ ones as an application of Chernoff bounds shows (see e.g., [6]). The probability to create by a mutation of an element which consists of at most $\lceil 7n/12 \rceil$ ones an individual on the path behind the peak is bounded by $2^{-\Omega(n)}$ since therefore, at least $\lceil n/13 \rceil$ bits have to change. Afterwards, to the best an element of the path before the peak $p_{n-\lceil n/3 \rceil - k}$, $k \geq 1$, generates the peak or an element behind it. Hence, the probability to create an element $p_{n-\lceil n/3 \rceil + i}$, $i \geq 1$, prior to $p_{n-\lceil n/3 \rceil}$ is bounded by $\mathcal{O}(1/n)$ since a special $k+i$-bit mutation (has a probability of at most $\sum_{i=k+1}^{n} 1/n^i (1-1/n)^{n-i} \leq 2/n^{k+1}$)

has to be performed prior to a special k-bit mutation (has a probability of at least $1/n^k(1-1/n)^{n-k} \geq 1/(en^k)$). If $p_{n-\lceil n/3\rceil}$ is contained in the population, then the failure probability to select this element for mutation is upper bounded by $(\mu-1)\cdot(2^{n^2}+3n^2+2n-1)/(2^{n^2+n}+3n^2+3n-\lceil n/3\rceil) = 2^{-\Omega(n)}$. This holds since the other $\mu-1$ individuals of the population have a function value of at most $2^{n^2}+3n^2+2n-1$ and even the peak has a function value of $2^{n^2+n}+3n^2+3n-\lceil n/3\rceil$. Moreover, the probability is bounded by $2^{-\Omega(n)}$ that a mutation of the peak generates the optimum which is also the only element which has a function value of more than $2^{n^2}+3n^2+2n-1$. This holds since therefore, a special $\lceil n/3\rceil$-bit mutation is necessary. Hence, the probability either to create the optimum or to select another element than the peak for mutation is bounded by $2^{-\Omega(n)}$. □

Nevertheless, we are interested in a function, whereon the $(\mu$+$1)$ EA$^{\text{prop}}$ is even totally inefficient and not only highly inefficient.

3.2 Main Results

We will present a function, where the $(\mu$+$1)$ EA$^{\text{rank}}$ even turns to total inefficiency now. Therefore, the previous result will be *amplified in the function* similar than it was realized in [7]. Let $s_0,\ldots,s_{\lceil n/\log n\rceil\lceil n/\log n\rceil}$, where $s_0 := 1^{n-\lceil n/3\rceil}0^{\lceil n/3\rceil}$, denote a path which consists of elements with at least $n-\lceil n/3\rceil$ ones and furthermore, the Hamming distance of s_i, $i \geq 0$, and s_{i+j} equals j for $0 \leq j \leq \lceil n/\log n\rceil$ and the Hamming distance is at least $\lceil n/\log n\rceil$ for $j > \lceil n/\log n\rceil$. A construction of such a path is presented in [7]. In order to describe the construction we remember the Gray code. The ℓ-digit *Gray code* $\mathbf{G}_\ell$, $\ell \in \mathbb{N}$, maps the integer x, $0 \leq x \leq 2^\ell-1$, bijective to the binary space $\{0,1\}^\ell$. But in contrast to *Binary code* the values x and $x+1$ have Hamming distance one in Gray code for all $0 \leq x < 2^\ell-1$. Similar to Binary code it holds $\mathbf{G}_\ell(0) = 0^\ell$. Moreover, for an element $x \in \{0,1\}^n$ we call *block* j, $0 \leq j \leq \lceil\log(n/\log n)\rceil+1$, the substring $x_{j\lceil n/\log n\rceil+1}\cdots x_{(j+1)\lceil n/\log n\rceil}$ and associate a bit

$$x_{(j)} := \begin{cases} x_{(j+1)\lceil n/\log n\rceil} & \text{if } x_{j\lceil n/\log n\rceil+1} = \cdots \\ & \qquad \cdots = x_{(j+1)\lceil n/\log n\rceil} \\ \texttt{undefined} & \text{otherwise} \end{cases}$$

with each such block. The element $s_{i\lceil n/\log n\rceil}$ is defined as the unique bitstring, where $x_{(\lceil\log(n/\log n)\rceil+1)}\cdots x_{(0)} = \mathbf{G}_{\lceil\log(n/\log n)\rceil+2}(i)$, the last $n-\lceil n/3\rceil$ bits of the string are ones, and the others are zeros. Let k_i denote the single position where $\mathbf{G}_{\lceil\log(n/\log n)\rceil+2}(i)$ and $\mathbf{G}_{\lceil\log(n/\log n)\rceil+2}(i+1)$ differ. Hence, the element $s_{i\lceil n/\log n\rceil+j}$, $0<j<\lceil n/\log n\rceil$ is defined as the unique bitstring, where exactly the first j bits of block k_i in $s_{i\lceil n/\log n\rceil}$ are changed. This sequence of elements has the desired properties.

Let us now consider the function PPs : $\{0,1\}^n \to \mathbb{Z}^{>0}$, where (PPs stands for PathPeaks):

$$\mathrm{PPs}(x) := \begin{cases} n^3 + 2n + & \text{if } x = s_{i\lceil n/\log n\rceil} \\ \quad + (i+1)\lceil n/\log n\rceil & \\ n^3 + 2n + & \text{if } x = s_{i\lceil n/\log n\rceil + j}, \\ \quad + i\lceil n/\log n\rceil + j & \quad 0 < j < \lceil n/\log n\rceil \\ n^3 + n + j & \text{if } x = 1^j 0^{n-j}, \\ & \quad 0 \le j < n - \lceil n/3\rceil \\ n^3 + n - |x| & \text{otherwise} \end{cases}$$

We observe that for $n \ge 5$ the decision space contains the values $\{n^3, \ldots, n^3{+}n^2\}$ at most and it is $n^3{+}n^2 < (n{+}1)^3$. The elements $s_{i\lceil n/\log n\rceil}$, $0 \le i < \lceil n/\log n\rceil$, are peaks and $s_{\lceil n/\log n\rceil\lceil n/\log n\rceil}$ is the optimum. In contrast to PP, where the $(\mu{+}1)$ EA$^{\text{prop}}$ reachs an awkward situation once the $(\mu{+}1)$ EA$^{\text{prop}}$ reachs such an awkward situation with overwhelming probability quite often. The other properties are similar especially if we consider the following transformation:

$$g(v) := \begin{cases} v & \text{if } v < 5^3 \\ v + 2^{n^3} & \text{if } n^3 \le v < n^3 + 2n,\ n \ge 5 \\ v + 2^{n^3 + i\cdot n} & \text{if } n^3 + 2n + i\lceil n/\log n\rceil \le v \text{ and} \\ & \quad v < n^3 + 2n + (i+1)\lceil n/\log n\rceil, \\ & \quad 0 \le i \le \lceil n/\log n\rceil,\ n \ge 5 \\ v + 2^{n^3+n^2} & \text{if } n^3 + 2n + (\lceil n/\log n\rceil + 1)\cdot \\ & \quad \cdot\lceil n/\log n\rceil \le v < (n+1)^3,\ n \ge 5 \end{cases}$$

Similar investigations as in the previous section show that g is well-defined and t.s.i.

Theorem 14. *The expected number of steps until the* $(\mu{+}1)$ EA$^{\text{prop}}$*, where* $\mu \ge 2$*, has optimized* PPs *is upper bounded by* $\mathcal{O}(\mu n^3/\log^2 n)$.

Proof. The proof is similar to the one of Theorem 12. The expected number of steps to create an arbitrary element of the path s_i, $i \ge 0$, is upper bounded by $\mathcal{O}(\mu n^2)$. Afterwards, a special 1-bit mutation of the individual with largest index creates an element with an even larger index which will be accepted in the population. This element has either the largest or the second-largest function value and the probability to choose the desired individual is lower bounded by $1/(2\mu)$. Hence, the expected number of steps for such a mutation is upper bounded by $2e\mu n$. Moreover, at most $\lceil n/\log n\rceil\lceil n/\log n\rceil$ times such an operation has to be performed until the end of the path is reached and therefore, also the optimum is generated. This leads to an expected number of at most $\mathcal{O}(\mu n^3/\log^2 n)$ steps in total. □

We remark that the upper bound of $\mathcal{O}(\mu n^3/\log^2 n)$ expected steps holds again also for the $(\mu{+}1)$ EA$^{\text{rank}}$, where $\mu \ge 2$.

Theorem 15. *The probability that the* $(\mu{+}1)$ $\mathrm{EA}^{\mathrm{prop}}$ *has optimized* $g \circ \mathrm{PPs}$ *within* $2^{\Omega(n)}$ *steps is lower bounded by* $1 - 2^{-\Omega(n)}$.

Proof. The proof is similar to the one of Theorem 13. The probability that an element of the path s_i, $i \geq 0$, is created within the initialization is exponentially small. The probability is also exponentially small to create an element s_i, $i \geq 0$, for the first time by a mutation of an individual which does not equals $1^j 0^{n-j}$, $0 \leq j < n - \lceil n/3 \rceil$. Let s_i be the element in the population with largest index and there exists no peak in the population. The probability to create an element s_{i+j}, $j \geq \lceil n/\log n \rceil$ is exponentially small. This holds since all these less than $\lceil n/\log n \rceil \lceil n/\log n \rceil$ elements have a Hamming distance of at least $\lceil n/\log n \rceil$. Therefore, the probability to create an element following the next but one peak prior to an element on the path between the next and the next but one peak is exponentially small as well. Hence, the probability to create the next peak prior to an element behind it on the path is lower bounded by $1 - \mathcal{O}(1/n)$. Moreover, the probability either to take a shortcut, i.e., s_i creates s_{i+j}, $j \geq \lceil n/\log n \rceil$ or otherwise to jump over all $\lceil n/\log n \rceil$ peaks is bounded by $2^{-\Omega(n)} + \mathcal{O}(1/n)^{\lceil n/\log n \rceil} = 2^{-\Omega(n)}$. If a peak is generated as element with largest index and therefore, also as individual with the largest function value, the failure probability to select the peak is exponentially small. Furthermore, the probability to create an individual which function value differs by a factor of at most $2^{\Omega(n)}$ from the one of the peak is exponentially small as well. $\square$

4 Minimal and Maximal Classes

In Section 4.1 we will prove that the t.i. are neutral for every algorithm first and afterwards, we will present an algorithm $\mathrm{A}^{\mathrm{ids}}$, where the class of neutral transformations consists of all t.i. only. Moreover, in Section 4.2 the other extreme case will be investigated. We will present an algorithm $\mathrm{A}^{\mathrm{all}}$, where all transformations are neutral transformations. Both algorithms optimize all functions within a finite expected number of steps.

4.1 Minimal Class of Neutral Transformations

Let the search space S be a discrete one. In the following let the A be an arbitrary algorithm operating on S.

Theorem 16. *Let* $p_{t,f}$ *be the probability that the* A *needs more than* $t \geq 1$ *steps to optimize* $f : S \to \mathbb{Z}$. *If* $g : \mathbb{Z} \to \mathbb{Z}$ *is a t.i., then the* A *needs with a probability of* $p_{t,g\circ f} \leq p_{t,f}$ *more than* t *steps to optimize* $g \circ f$.

Proof. The proof is similar to the one of Theorem 5. For every $t \geq 1$ it holds, if the A on f has history (s_t) with probability $p_{(s_t),f}$, then with equal probability $p_{(s_t),f}$ the A on $g \circ f$ has history (s_t). This holds for all histories (s_t) which do not contain an optimum with respect to $g \circ f$ and therefore, especially not with respect to f. $\square$

Let us now define the A$^{\text{ids}}$ operating on $S = \{0,1\}^n$ and where the class of neutral transformations consists of the truncated *id*entitie*s* only. Let $a : \mathbb{Z} \to \mathbb{N}$ be the bijective function, where

$$a(v) := \begin{cases} 2|v| & \text{if } v \geq 0 \\ 2|v| - 1 & \text{if } v < 0 \end{cases} .$$

Algorithm 17. A$^{\text{ids}}$

- Step 1. Query 0^n.
- Step 2. to 2^n. Query x_0, where
  $\text{num}(x_0) = a(f(0^n)) \bmod (2^n - 1) + 1$.
- Step $t > 2^n$. Choose $x_t \in \{0,1\}^n$ independently and
  uniformly at random. Query x_t.

If the function value of 0^n does not direct to an optimum or the element is optimal by itself, then the optimization takes at least an exponential number of steps.

Theorem 18. *If $g : \mathbb{Z} \to \mathbb{Z}$ is not a t.i., then there exists a function f, where the* A$^{\text{ids}}$ *needs at most two steps for optimization. The* A$^{\text{ids}}$ *needs at least* $2^n + 1$ *steps to optimize $g \circ f$.*

Proof. There exist at least two integers v and w, where $g(v) \neq v < w$ and $g(v) \neq g(w)$ holds. We distinguish the two cases that $g(v) > g(w)$ (Case 1) and $g(v) < g(w)$ (Case 2).
Case 1. We investigate the function $f_1 : \{0,1\}^n \to \mathbb{Z}$, where

$$f_1(x) := \begin{cases} w & \text{if } x = 0^n \text{ or} \\ & \qquad \text{num}(x) = a(g(w)) \bmod (2^n - 1) + 1 \\ v & \text{otherwise} \end{cases} .$$

The A$^{\text{ids}}$ on f_1 queries 0^n first which is also an optimum.

The A$^{\text{ids}}$ on $g \circ f_1$ queries within the first 2^n steps the elements 0^n and x_0 only, where it is $\text{num}(x_0) = a(g(w)) \bmod (2^n - 1) + 1$. These are the two single non-optimal elements.
Case 2. We investigate the function $f_2 : \{0,1\}^n \to \mathbb{Z}$, where

$$f_2(x) := \begin{cases} v & \text{if } x = 0^n \text{ or} \\ & \qquad \text{num}(x) = a(g(v)) \bmod (2^n - 1) + 1 \\ w & \text{otherwise} \end{cases} .$$

The A$^{\text{ids}}$ on f_2 queries 0^n first and $x_0 \neq 0^n$ second, where it is $\text{num}(x_0) = a(v) \bmod (2^n - 1) + 1$. Since $v \neq g(v)$ it holds for n large enough that also $a(v) \bmod (2^n - 1) + 1 = a(v) + 1 \neq a(g(v)) + 1 = a(g(v)) \bmod (2^n - 1) + 1$. Therefore, x_0 represents an optimum.

The A$^{\text{ids}}$ on $g \circ f_2$ queries within the first 2^n steps the elements 0^n and x_0 only, where it is $\text{num}(x_0) = a(g(v)) \bmod (2^n - 1) + 1$. These are again non-optimal elements. □

4.2 Maximal Class of Neutral Transformations

Let us define the $\mathrm{A}^{\mathrm{all}}$ operating on pseudo-Boolean functions and where *all* functions are neutral transformations.

Algorithm 19. $\mathbf{A}^{\mathbf{all}}$

- Choose $x_0 \in \{0,1\}^n$ uniformly at random.
- Step 1. to $2^{2n}-1$. Query x_0.
- Step $t \geq 2^{2n}$. Choose $x_t \in \{0,1\}^n$ independently and
  uniformly at random. Query x_t.

Theorem 20. *If and only if the* $\mathrm{A}^{\mathrm{all}}$ *optimizes f within an expected number of $t < 2^n$ steps, then the* $\mathrm{A}^{\mathrm{all}}$ *optimizes $g \circ f$ within an expected number of t steps, where g is an arbitrary transformation.*

Proof. We will show that the $\mathrm{A}^{\mathrm{all}}$ needs an expected number of $t < 2^n$ steps on the constant functions only. In this case, even the first query is optimal. Then, the proposed result follows since f is a constant function and each transformation of a constant function leads to a constant function as well. Therefore, if f is not the constant function, then there exists at least one element x which is non-optimal. With probability 2^{-n} the algorithm $\mathrm{A}^{\mathrm{all}}$ selects $x_0 = x$ initially. Afterwards, at step 2^{2n} an element which does not necessarily equal $x_0 = x$ is queried and to the best this element is optimal. This leads to an expected number of at least $2^{-n}2^{2n} = 2^n$ steps for the $\mathrm{A}^{\mathrm{all}}$ on f, if f not a constant function. $\square$

5 Black-Box Algorithms

We investigate functions with an arbitrary discrete search space S. Nevertheless, we have to define black-box algorithms more precisely first (see [3]).

Algorithm 21. Black-Box Algorithm
Step $t \geq 1$. Depending on $(x_1, f(x_1)), \ldots, (x_{t-1}, f(x_{t-1}))$ determine a probability distribution on S, choose x_t according to this distribution, query x_t, and receive its function value $f(x_t)$ from the black box.

A black-box algorithm which determines deterministically/ randomly the next query is called a *deterministic/randomized black-box algorithm.* We remark that the class of randomized black-box algorithms includes the class of deterministic ones. However, for every function with discrete search space S a black-box algorithm exists which requires an expected number of at most $(|S|+1)/2$ queries for optimization (see [3]). We will obtain results, where every black-box algorithm also needs at least $(|S|+1)/2$ queries for optimization. Let us consider the class of t.s.i. transformations first.

Theorem 22. *Let $p_{t,f}$ be the probability that a black-box algorithm* B_f *needs more than $t \geq 1$ steps to optimize $f : S \to \mathbb{Z}$. If $g : \mathbb{Z} \to \mathbb{Z}$ is t.s.i., then there exists a black-box algorithm* $\mathrm{B}_{g \circ f}$ *with access to g which needs for every t with a probability of $p_{t,g\circ f} \leq p_{t,f}$ more than t steps to optimize $g \circ f$.*

Proof. The idea of the theorem and its proof is that $\mathrm{B}_{g\circ f}$ can simulate B_f if it has access to g.

We describe how the black-box algorithm $\mathrm{B}_{g\circ f}$ works. It is sufficient to simulate each step of B_f until an optimum for $g \circ f$ is queried. We remark that if B_f reachs an optimum for f, then this is also an optimum for $g \circ f$. We prove by induction on the number $t \geq 1$ of steps that each step of B_f can be simulated properly. Prior to the first step $t = 1$ the history of B_f is empty and $\mathrm{B}_{g\circ f}$ can simulate the first step of B_f and generate the initial element $x_1 \in S$ according to the same probability distribution. For $t > 1$, let $(x_1, g \circ f(x_1)), \ldots, (x_{t-1}, g \circ f(x_{t-1}))$ be the history of $\mathrm{B}_{g\circ f}$. Either x_{t-1} is optimal for $g \circ f$ and $\mathrm{B}_{g\circ f}$ has successfully finished or since g is injective for all non-optimal function values we can evaluate $f(x_i)$, $1 \leq i \leq t-1$, from $g \circ f(x_i)$. Afterwards, B_f can be simulated with history $(x_1, f(x_1)), \ldots, (x_{t-1}, f(x_{t-1}))$ and $\mathrm{B}_{g\circ f}$ can choose $x_t \in S$ according to the same probability distribution as B_f. □

For the proof it was essential that the black-box algorithm has access to the specific transformation g. Before we investigate black-box algorithms and not t.s.i. transformations we will demonstrate that the access to the specific transformation can be essential (at least up to some degree). In the following we will investigate the search space $S = \{0,1\}^n$ again. At first, we analyze the behavior of all black-box algorithms on plateaus and therefore, let $\textsc{Plateau}_a : \{0,1\}^n \to \mathbb{Z}$, where for $a \in \{0,1\}^n$

$$\textsc{Plateau}_a(x) := \begin{cases} 1 & \text{if } x = a \\ 0 & \text{otherwise} \end{cases} .$$

The class of functions $\textsc{Plateau}$ consists of all $\textsc{Plateau}_a$.

Lemma 23. *Every black-box algorithm needs at least an expected number of* $(2^n + 1)/2$ *queries to optimize* $f \in \textsc{Plateau}$.

This was proven by Droste, Jansen, and Wegener [3].

We remark that the proposed results concerning black-box algorithms in Theorem 7 and Theorem 10 follow directly from the investigations made here.

Hence, let $\textsc{Pointer}_a : \{0,1\}^n \to \mathbb{Z}$, where for $a \in \{0,1\}^n$

$$\textsc{Pointer}_a(x) := \begin{cases} 2^{n+1} & \text{if } x = a \\ \mathrm{num}(a) & \text{otherwise} \end{cases} .$$

The class of functions $\textsc{Pointer}$ consists of all $\textsc{Pointer}_a$. Moreover, let $\mathcal{G}_{\text{t.s.i.}}$ be the class of all t.s.i. transformations.

Theorem 24. *There exists a (deterministic) black-box algorithm* $\mathrm{B}_{\textsc{Pointer}}$ *which needs for every* $f \in \textsc{Pointer}$ *at most two queries for optimization. Every (randomized) black-box algorithm without access to* $g \in \mathcal{G}_{\text{t.s.i.}}$ *needs at least an expected number of* $(2^n + 1)/2$ *queries to optimize* $g \circ f$, $f \in \textsc{Pointer}$.

Proof. A black-box algorithm $\text{B}_{\text{Pointer}}$ can query 0^n first. Either the element 0^n is optimal or the algorithm with history $(0^n, \text{num}(a))$ can compute and query the optimum a.

Let $g_i(v) : \mathbb{Z} \to \mathbb{Z}$ be the t.s.i. function, where

$$g_i(v) := \begin{cases} v - 2^n & \text{if } v < 0 \\ v + i & \text{if } v \geq 0 \end{cases}$$

for $-(2^n - 1) \leq i \leq 2^n - 1$. In the following, we will consider these transformations only. Hence, independent of g_i, $-(2^n - 1) \leq i \leq 2^n - 1$, the element a stays the single optimum for $g_i \circ \text{Pointer}_a$ and all other elements have the same non-optimal function value. This holds for all a. Moreover, we observe that for every $0 \leq w \leq 2^n - 1$ there exists a t.s.i. transformation, where $g_{w-\text{num}(a)} \circ \text{Pointer}_a(x) = w$, $x \in \{0,1\}^n - \{a\}$ holds. Hence, the class of investigated functions $\mathcal{G}_{\text{t.s.i.}} \circ \text{Pointer} := \{g \circ \text{Pointer} \,|\, g \in \mathcal{G}_{\text{t.s.i.}}\}$ especially contains all functions $f_{a,b} : \{0,1\}^n \to \mathbb{Z}$, where

$$f_{a,b}(x) := \begin{cases} \max & \text{if } x = a \\ \text{num}(b) & \text{otherwise} \end{cases}$$

for $a, b \in \{0,1\}^n$. Let $\mathcal{F} \subseteq \mathcal{G}_{\text{t.s.i.}} \circ \text{Pointer}$ consist of all these $f_{a,b}$. We are in a similar situation as for Plateau in Lemma 23. Since each randomized black-box algorithm can be interpreted as a probability distribution over deterministic ones (see [3]) we can apply Yao's Minimax Principle (see e.g., [6]). For this particular situation, it states that the expected number of steps of each randomized black-box algorithm on $f_{a,b} \in \mathcal{F}$ is lower bounded by the minimal average number of steps – according to an arbitrary distribution on $\mathcal{F}$ – with respect to every deterministic black-box algorithm. We choose $f_{a,b} \in \mathcal{F}$ uniformly at random and observe that, if an arbitrary non-optimal element is queried, then every element not queried before is optimal with the same probability. This is equivalent to Plateau and therefore, by Lemma 23, every black-box algorithm needs an expected number of at least $(2^n + 1)/2$ steps to optimize $f_{a,b} \in \mathcal{F}$ and moreover, especially with respect to $\mathcal{G}_{\text{t.s.i.}} \circ \text{Pointer}$. □

Finally, we investigate black-box algorithms on not t.s.i. transformations.

Theorem 25. *If $g : \mathbb{Z} \to \mathbb{Z}$ is not t.s.i., then there exist classes of functions $\mathcal{F}$, where a (deterministic) black-box algorithm $\text{B}_{\mathcal{F}}$ exists which needs at most a linear number of queries to optimize f for every $f \in \mathcal{F}$ (in particular $n + 1$). Every (randomized) black-box algorithm (with access to g) needs at least an expected number of $(2^n + 1)/2$ queries to optimize $g \circ f$, $f \in \mathcal{F}$.*

Proof. We will distinguish two cases. The first case considers the situation that there exists at least one integer which is transformed to a larger value than its successor (Case 1). The second case considers the situation that there is at least one integer which is transformed to the same non-optimal value than its successor (Case 2).

Case 1. There is an integer v, where $g(v) > g(v+1)$ holds. Then, equivalent to the corresponding case in the proof of Theorem 7 we consider the class of functions $\mathcal{F}_1$ which consists of all functions $f_{1,a} : \{0,1\}^n \to \mathbb{Z}$, where for $a \in \{0,1\}^n$

$$f_{1,a}(x) := \begin{cases} v & \text{if } x = a \\ v+1 & \text{otherwise} \end{cases} .$$

A black-box algorithm $\mathrm{B}_{\mathcal{F}_1}$ can query 0^n and 1^n. The function value of at least one of these two elements is optimal.

The class of functions $g \circ \mathcal{F}_1$ is similar to PLATEAU since $g(v+1) < g(v)$. Therefore, by Lemma 23, every black-box algorithm needs at least an expected number of $(2^n+1)/2$ steps to optimize $g \circ f_{1,a}$, $f_{1,a} \in \mathcal{F}_1$.

Case 2. There exists an integer v where $g(v) = g(v+1)$ and at least one further integer $w > v+1$ where $g(v+1) < g(w)$ holds. Then, we investigate the class of functions $\mathcal{F}_2$ which consists of all functions $f_{2,a} : \{0,1\}^n \to \mathbb{Z}$, where

$$f_{2,a}(x) := \begin{cases} w & \text{if } x = a \\ v+1 & \text{if } x = p_i,\ 0 \le i \le n, \text{ and } x \ne a \\ v & \text{otherwise} \end{cases}$$

for $a = a_1 \cdots a_n \in \{0,1\}^n$ with $p_i := a_1 \cdots a_i 0^{n-i}$, $0 \le i \le n$. The sequence of elements $p_0, \ldots, p_n$ describes a path.

A black-box algorithm $\mathrm{B}_{\mathcal{F}_2}$ can identify p_i, $1 \le i \le n$, within one query, if p_{i-1} is contained in its history. The individual p_i can be determined by querying $a_1 \cdots a_{i-1} 1 0^{n-i} =: p'_{i-1}$ which differs with p_{i-1} in position i only. If the function value of p'_{i-1} is at least as large as the function value of p_{i-1}, i.e., it is $v+1$ or w, then p_i is p'_{i-1}. Otherwise, i.e., the function value is v, the element p_i is p_{i-1}. Moreover, $\mathrm{B}_{\mathcal{F}_2}$ queries $p_0 = 0^n$ at first. Hence, within at most $n+1$ queries the optimum p_n is created.

For the class of functions $g \circ \mathcal{F}_2$ we are in the same situation as in the previous case since $g(v) = g(v+1) < g(w)$. □

6 Conclusion

We have proven rigorously that the class of neutral transformations for a simple rank-based EA consists of all truncated strictly increasing functions. This also holds for black-box algorithms with access to the transformation. This class of neutral transformations is even a subset for all rank-based algorithms. For all not fitness-based algorithms the class of neutral transformations contains at least the increasing functions, while for a simple fitness-based EA it was proven that its class of neutral transformations even does not contain all truncated strictly increasing functions. We have presented an algorithm whose class of neutral transformations is completely described by the functions that are neutral for all algorithms, the truncated identities. And we have presented an algorithm whose class of neutral transformations even consists of all transformations.

Acknowledgements

This research was supported by a Grant from the G.I.F., the German-Israeli Foundation for Scientific Research and Development. The author thanks Stefan Droste and Ingo Wegener for their help while preparing this paper.

References

[1] T. Bäck, D. Fogel, and Z. Michalewicz. *Handbook of Evolutionary Computation.* Institute of Physics Publishing and Oxford University Press, New York, 1997.

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

[3] S. Droste, T. Jansen, and I. Wegener. Upper and lower bounds for randomized search heuristics in black-box optimization. *Theory of Computing Systems*, 2005. Accepted for publication.

[4] J. Garnier, L. Kallel, and M. Schoenauer. Rigorous hitting times for binary mutations. *Evolutionary Computation*, 7:173–203, 1999.

[5] T. Jansen and I. Wegener. Evolutionary algorithms – how to cope with plateaus of constant fitness and when to reject strings with the same fitness. *IEEE Transactions on Evolutionary Computation*, 5:589–599, 2001.

[6] R. Motwani and P. Raghavan. *Randomized Algorithms.* Cambridge University Press, Cambridge, 1995.

[7] T. Storch. On the choice of the population size. In *Genetic and Evolutionary Computation Conference – GECCO 2004, LNCS 3102*, pages 748–760, 2004.

[8] C. Witt. An analysis of the (μ+1) EA on simple pseudo-boolean functions. In *Genetic and Evolutionary Computation Conference – GECCO 2004, LNCS 3102*, pages 761–773, 2004.

[9] C. Witt. Worst-case and average-case approximations by simple randomized search heuristics. In *Symposium on Theoretical Aspects of Computer Science – STACS 2005, LNCS 3404*, pages 44–56, 2005.