A Rigorous View On Neutrality

Benjamin Doerr*   Michael Gnewuch†   Nils Hebbinghaus*   Frank Neumann*

∗ Algorithms and Complexity Group
Max-Planck-Institut für Informatik
Saarbrücken, Germany

† Department of Computer Science
Christian-Albrechts-University of Kiel
Kiel, Germany

Abstract—**Motivated by neutrality observed in natural evolution often redundant encodings are used in evolutionary algorithms. Many experimental studies have been carried out on this topic. In this paper we present a first rigorous runtime analysis on the effect of using neutrality. We consider a simple model where a layer of constant fitness is distributed in the search space and point out situations where the use of neutrality significantly influence the runtime of an evolutionary algorithm.**

I. INTRODUCTION

Evolutionary algorithms have been successfully applied to different kinds of optimization problems. One important issue when designing such an algorithm is the encoding of possible solutions. Often there is no direct representation but a special genotype-phenotype mapping which in many cases introduces redundancy. The use of redundancy is often motivated by the neutrality which can be found in natural evolution. It has been considered in several experimental studies whether redundancy can significantly help to come up with better algorithms [1, 14, 15, 18, 19].

The main aim of our investigations is to understand the effect of neutrality a little bit deeper from a theoretical point of view. This is also the case for all rigorous analyses carried out in the past. The first rigorous result on the runtime behavior of EAs has been obtained in [10] where it is proven that the well-known (1+1) EA is able to optimize the function ONEMAX within an expected number of $O(n \log n)$ iterations. The function ONEMAX has also been considered in other scenarios as the starting point for different runtime analyses [2, 3, 12]. After the first result for the (1+1) EA in [10] it has been shown in [16] that this algorithm can solve some Long Path functions in expected polynomial time. Later on different results have been obtained for the optimization of pseudo Boolean functions with different properties (see e. g. [4, 8]). The (1+1) EA has also been subject of different investigations on some of the best-known combinatorial optimization problems [7, 11, 21] and seems to be a "standard" algorithms for investigating the behavior of EAs with respect to their runtime behavior. Algorithms working with a larger population size have been considered in [17, 20].

We examine situations where neutrality does (or does not) help to speed up computations compared with algorithms not using such a mechanism. In our investigations we analyze a simple model of neutrality already examined in [5, 6] by an experimental study. Our aim is to carry out a rigorous analysis of the runtime for the model and functions investigated in these papers.

The model of neutrality analyzed in our investigations uses a single layer of constant fitness. In the case of the function ONEMAX using such a neutrality mechanism may lead to an exponential runtime while the optimization time is a small polynomial if no neutrality mechanism is used [4]. In [5, 6] it has been stated that neutrality might be useful when considering a special class of deceptive functions. We prove that the optimization time of the (1+1) EA is exponential for all possible function values the layer can attain. Afterwards, we present a function where neutrality is provably helpful. In particular we show that neutrality might turn an optimization time that is exponential with probability close to 1 into an expected polynomial optimization time if the right neutrality layer is used. The analysis for our function considers the mixing time of the (1+1) EA. Using mixing time arguments similar to [13] for the analysis of evolutionary algorithms may be of independent interest and also helpful for analyzing evolutionary algorithms in other situations.

The outline of the paper is as follows. In Section II we introduce the considered algorithm and our model of neutrality. The functions considered in [5, 6] are analyzed rigorously in Section III and IV. Section V shows our example where using neutrality is provably helpful. We finish with some conclusions and the statement of some topics for future work.

II. ALGORITHMS AND NEUTRALITY

We consider the optimization of pseudo Boolean functions $f: \{0,1\}^n \to \mathbb{R}$. Our aim is to investigate how neutrality may influence the search process when dealing with functions of different kinds. We consider the model investigated in [5, 6] where a constant fitness layer that is identically distributed in the whole search space is used. This is done by adding an extra neutrality bit which decides whether a search point is on the neutrality layer are not.

Using neutrality, we consider the search space $\{0,1\}^m$, where $m = n + 1$. The function value of the function $f^*: \{0,1\}^m \to \mathbb{R}$ with neutrality is given by the original function f if the neutrality bit x_m is set to 0. In the case that $x_m = 1$, the search point is on the neutrality layer and the function value f_{layer} is returned. Hence, we set

$$f^*(x) := \begin{cases} f(x_1, \ldots, x_n) & : \quad x_m = 0 \\ f_{layer} & : \quad x_m = 1. \end{cases}$$

We consider the well-known (1+1) EA working on bit strings of length m and flipping in each mutation step each bit with probability $1/m$. In addition the algorithm uses an elitism selection method, where the offspring replaces its parent if its function value is at least as good as the value of the parent.

Algorithm 1: (1+1) EA

1) Choose an initial solution $x \in \{0,1\}^m$
2) Repeat
   - Create x' by flipping each bit of x with probability $1/m$.
   - If $f^*(x') \geq f^*(x)$, set $x := x'$.

In the case that the original function f is considered the algorithm works with bit strings of length n and each bit is flipped with probability $1/n$.

Analyzing the (1+1) EA with respect to its runtime behavior, we are interested in the number of constructed solutions until an optimal one has been created for the first time. This is called the runtime or optimization time of the considered algorithm. Often, the expectation of this value is considered and called the expected optimization time.

For convenience we make some definitions. For every bit string $x \in \{0,1\}^m$ we denote by $\widetilde{x} \in \{0,1\}^n$ the string obtained from x by removing the neutrality bit. Furthermore, we define $|x|_1 := \sum_{i=1}^n x_i$ and $|x|_0 := n - |x|_1$. Note that $|x|_1$ respectively $|x|_0$ are the numbers of 1s respectively 0s of $\widetilde{x}$ (ignoring the neutrality bit).

In the following we proof some properties that are very useful for our analyses of the effect of neutrality. Lemma 1 provides the probability theoretical basis, whereas Lemma 2 is formulated more related to the (1+1) EA and can thus be directly applied in some of our proofs. We use the following convention: the s-fold composition $\varphi \circ \cdots \circ \varphi$ of a function $\varphi : \Omega \to \Omega$, Ω some domain, is denoted by φ^s. Furthermore, φ^0 should denote the identical mapping on Ω.

Lemma 1: Let $\Omega := \{0,1\}^n$ and $x \in \Omega$ be uniformly distributed. Let $N \subseteq \Omega$ and $\psi, \varphi : \Omega \to \Omega$ be uniformly distributed random variables. Moreover, let X_0 be the event that $x \in N$, and for $s \geq 1$ let X_s be the event that $(\psi \circ \varphi^{s-1})(x) \in N$. Then

$$\mathrm{Prob}\left(\bigcap_{s=0}^{t} X_s \right) \geq 1 - (t+1)\frac{|\Omega \setminus N|}{|\Omega|}$$

for every $t \in \mathbb{N}$

Proof: We have

$$1 - \mathrm{Prob}\left(\bigcap_{s=0}^{t} X_s \right) \leq \mathrm{Prob}(x \in \Omega \setminus N)$$
$$+ \mathrm{Prob}(\exists\, 1 \leq s \leq t : \neg X_s)$$
$$\leq \frac{|\Omega \setminus N|}{|\Omega|} + \sum_{s=1}^{t} \mathrm{Prob}(\neg X_s).$$

Since x, ψ, and φ are uniformly distributed, it is immediate by trivial induction that the random variable $(\psi \circ \varphi^{s-1})(x)$ is uniformly distributed for $s = 1, \ldots, t$. Thus,

$$\mathrm{Prob}(\neg X_s) = \frac{|\Omega \setminus N|}{|\Omega|}$$

for all $1 \leq s \leq t$ and the statement of the Lemma follows. ∎

Lemma 2: Let $\Omega := \{0,1\}^n$ and define a special subset $N := \{y \in \Omega : f(y) < f_{layer}\}$. Let x be the initial solution of the $(1+1)$ EA. Let us assume that x is neutral and $\widetilde{x}$ is uniformly distributed in Ω. Then all solutions x up to step t of the $(1+1)$ EA (including the initial solution) stay neutral and satisfy $\widetilde{x} \in N$ with probability P_{neu} at least $1 - (t+1)\frac{|\Omega \setminus N|}{|\Omega|}$.

Proof: For convenience we denote here the initial solution by x_0 and the solution after step s by x_s. Recall that x_0 is neutral and observe that $x_1, \ldots, x_i$ stay in any case neutral if $\widetilde{x}_0, \ldots, \widetilde{x}_i \in N$. Thus we can lower bound the probability P_{neu} by the probability

$$\mathrm{Prob}(\forall\, 0 \leq s \leq t : \widetilde{x}_i \in N).$$

To estimate this probability, let us define the random variables $\psi, \varphi : \Omega \to \Omega$, where ψ changes any bit of a given string in Ω independently with probability $1/m$, and

$$\varphi(y) = \begin{cases} \psi(y) & \text{with probability } 1 - 1/m, \\ y & \text{with probability } 1/m. \end{cases}$$

Both random variables are uniformly distributed. We simulate the generation of an offspring x' of a solution x of the $(1+1)$ EA by $x' = (\psi(\widetilde{x}), \nu)$, where the neutral bit ν differs from the neutral bit of x with chance $1/m$. If x is neutral and $\widetilde{x}, \widetilde{x}' \in N$, then the first n bits of the new solution are generated by $\varphi(\widetilde{x}_i)$. We claim that

$$\mathrm{Prob}(\forall\, 0 \leq s \leq t : \widetilde{x}_i \in N) \geq$$
$$\mathrm{Prob}(\widetilde{x}_0 \in N, \forall\, 1 \leq s \leq t : (\psi \circ \varphi^{s-1})(\widetilde{x}_0) \in N).$$

Indeed, let $\widetilde{x}_0 \in N$ and $(\psi \circ \varphi^{s-1})(\widetilde{x}_0) \in N$ for all $1 \leq s \leq t$. By induction on s we show that $\widetilde{x}_s \in N$ and $\widetilde{x}_s = \varphi^s(\widetilde{x}_0)$. Obviously, $\widetilde{x}_0 = \varphi^0(\widetilde{x}_0) \in N$. Assume now that $\widetilde{x}_i \in N$ and $\widetilde{x}_i = \varphi^i(\widetilde{x}_0)$ for $0 \leq i \leq s-1$. It is easily seen (e.g., by trivial induction) that x_{s-1} is necessarily neutral. If x'_{s-1} denotes its offspring, then $\widetilde{x}'_{s-1} = \psi(\widetilde{x}_{s-1}) = (\psi \circ \varphi^{s-1})(\widetilde{x}_0) \in N$. Thus $\widetilde{x}_s \in N$ and $\widetilde{x}_s = \varphi(\widetilde{x}_{s-1}) = \varphi^s(\widetilde{x}_0)$. Hence our claim is verified. Due to Lemma 1 the proof is complete. ∎

III. ANALYSIS FOR ONEMAX

In [5, 6] experiments for the function ONEMAX have been carried out. ONEMAX is the simplest non-trivial function that can be considered and has been served as a starting point for analyzing a wide range of evolutionary computation methods [2–4, 10, 12]. It is well known that the expected optimization time of the (1+1) EA without using neutrality is $O(n \log n)$ on each linear pseudo-boolean function (including ONEMAX) [4]. The function ONEMAX is defined as

$$\mathrm{ONEMAX}(x) = \sum_{i=1}^{n} x_i.$$

In the following we will carry out a runtime analysis of the (1+1) EA on its extension to neutrality. The following result shows that if the value of the neutrality layer is only a little bit smaller than the expected ONEMAX-value of the initial solution, neutrality does not effect the magnitude of the expected optimization time.

Theorem 3: Choosing $f_{layer} = cn$, $c < 1/2$ a constant, the expected optimization time of the (1+1) EA on ONEMAX* is $\Theta(n \log n)$.

Proof: The lower bound holds for any pseudo-boolean function with a single global optimum using the ideas of the coupon collectors theorem [9]. The corresponding proof can be found in [4]. It remains to show the upper bound.

If $x_m = 1$ holds for the initial solution x the solution is on the neutrality layer. The expected time until a solution z with $z_m = 0$ is determined is $O(m)$. Up to this moment, all solutions are accepted, since they all have the same function value f_{layer}. Thus, for each individual x the string $\widetilde{x}$ is uniformly distributed in $\{0,1\}^n$. Hence, using standard Chernoff bounds one gets $|z|_1 > cn$ with probability $1 - e^{-\Omega(n)}$. If such a z with $z_m = 0$ and $|z|_1 > f_{layer}$ has been produced no solution with neutrality bit 1 is accepted by the (1+1) EA afterwards. Therefore, the (1+1) EA behaves in this case as the (1+1) EA on the original function ONEMAX and its expected optimization time is upper bounded by $O(n \log n)$ due to fitness layer arguments given in [4].

It remains to reduce the case that the first solution z with $z_m = 0$ has ONEMAX-value at most cn to the first one. The probability for this event is $e^{-\Omega(n)}$ as already stated. We show that in this case after an expected number of $O(n^2)$ steps of the (1+1) EA a solution y with ONEMAX$(y) > f_{layer}$ and $y_m = 0$ will be determined. Note that the contribution of this $O(n^2)$ mutation steps to the expected runtime is $O(n^2)e^{-\Omega(n)} = e^{-\Omega(n)}$. Let us consider a solution x with $|x|_1 \leq cn$ and its offspring z. The expected number of 1s in $\widetilde{z}$ is $|x|_1 + \frac{|x|_0 - |x|_1}{n} \geq |x|_1 + (1 - 2c)$. Thus, in an expected number of less than $\frac{n}{1-2c}$ steps the (1+1) EA produces a solution x with $|x|_1 > cn$. Since the probability that the neutrality bit flips in this last step is $\frac{1}{m} = \Omega(\frac{1}{n})$, after an expected number of $O(n^2)$ (in this special situation) a solution z with $|z|_1 > cn$ and $z_m = 0$ is produced. Thus, we have reduced this special case to the normal case considered above. Altogether, we have shown that the (1+1) EA determines the optimal solution in expected time $\Theta(n \log n)$. ∎

The next result shows that if the value of the neutrality layer is a bit larger than the expected ONEMAX-value of the initial solution, neutrality does effect the magnitude of the expected optimization time in a counterproductive way.

Theorem 4: Choosing $f_{layer} = cn$, $1/2 < c < 1$ a constant, the (1+1) EA does not determine the optimum on ONEMAX* in $e^{\frac{(2c-1)^2}{12}n}$ steps with probability $1/2 - e^{-\Omega(n)}$. In particular, the expected optimization time of the (1+1) EA is exponential.

Proof: With probability $1/2$ the initial solution of the (1+1) EA is neutral and we can apply Lemma 2. We define the set $N := \{y \in \Omega : \text{ONEMAX}(y) < cn\}$. Using Chernoff

bounds for an x chosen uniformly at random from $\{0,1\}^n$ we get

$$\frac{|\Omega \setminus N|}{|\Omega|} = \text{Prob}(\text{ONEMAX}(x) \geq cn) \leq e^{-\frac{(2c-1)^2}{6}n}.$$

Applying Lemma 2, in the first $e^{\frac{(2c-1)^2}{12}n}$ steps of the (1+1) EA the currently best solution stays neutral and in N with probability $1 - e^{-\Omega(n)}$. This proves the theorem. ∎

IV. TRAP FUNCTION

In this section we study deceptive functions also considered from an experimental point of view in [5, 6]. Deceptive functions seem to be particularly hard to optimize by evolutionary algorithms as they mislead the search process. We investigate

$$\text{TRAP}_z(x) := \begin{cases} \frac{n-1}{z}(z - |x|_1) & \text{if } |x|_1 \leq z \\ \frac{n}{n-z}(|x|_1 - z) & \text{if } |x|_1 > z. \end{cases}$$

In the case $z = n - 1$, we are dealing with the function TRAP for which a lower bound n^n on the expected runtime has been given in [4].

We are considering the case where the slope-change location z is given by $z = \lambda n$ for some $1/2 < \lambda < 1$.

Theorem 5: If $1/2 < \lambda < 1$ and $z := \lambda n$, the optimization time of the (1+1) EA on the function TRAP$_z$ is lower bounded by $n^{n/2}$ with probability $1 - e^{-\Omega(n)}$. In particular, the expected optimization time of the (1+1) EA is exponential.

Proof: Let x be the initial individual of the (1+1) EA. It satisfies $|x|_1 < \frac{\lambda + 1/2}{2}n$ with probability $1 - e^{-\Omega(n)}$ using Chernoff bounds. The only steps accepted by the (1+1) EA are the following. Either an offspring with at least the number of 1s as the current solution or an offspring with more than $z = \lambda n$ 1s is produced. In the former case the (1+1) EA is mislead in the direction of the local optimum 0^n. For the latter case, a mutation step with at least $d := \frac{2\lambda-1}{4}n$ flipping bits is needed. The probability of such a step is at most

$$\binom{n}{d}\frac{1}{n^d} \leq \frac{1}{d!} = n^{-\Omega(n)}.$$

We consider a phase of n^3 mutation steps. The probability that an individual x with $|x|_1 > z$ is produced in this phase is clearly $e^{-\Omega(n)}$ by the last argument. On the other hand, in an expected number of $O(n \log n)$ steps the (1+1) EA has reached the search point 0^n since the function TRAP$_z$ behaves just like the function ZEROMAX on the set $\{x \in \{0,1\}^n \mid |x|_1 \leq z\}$. Using Markov's inequality, there is a constant $c > 0$ such that with probability at least $1/2$ the search point 0^n is reached in at most $cn \log n$ steps. In the phase of n^3 steps we repeat this short phase of $cn \log n$ steps more than n times. Thus, the probability that the (1+1) EA has determined the string 0^n as current solution after n^3 steps is at least $1 - e^{-\Omega(n)}$.

Now the only mutation step accepted by the (1+1) EA is the step flipping all bits of the current solution 0^n at once. The probability for such a mutation is n^{-n}. Thus, the (1+1) EA determines the optimum 1^n in the next $n^{n/2}$ steps with probability $n^{-\Omega(n)}$. This proves the Theorem. ∎

We now investigate the effect of neutrality for the behavior of the (1+1) EA on the function TRAP_z. Like in the last section we choose the neutrality level as $f_{layer} = cn$ for a fixed constant $0 < c < 1$. More precisely, we examine this effect for two different ranges of c depending on the slope-change location $z = \lambda n$. In the next theorem we consider the range $0 < c < (1 - \frac{1}{2\lambda})$. If we choose c in this range, the set $N = \{y \in \Omega : \text{TRAP}_z(y) < f_{layer}\}$ (with $\Omega = \{0,1\}^n$) satisfies $\frac{|N|}{|\Omega|} = e^{-\Omega(n)}$. This is the essential property for the proof of the Theorem.

Theorem 6: If $0 < c < (1 - \frac{1}{2\lambda})$, the optimization time of the (1+1) EA on the function TRAP_z^* is lower bounded by $n^{n/2}$ with probability $1 - e^{-\Omega(n)}$.

Proof: Set $\varepsilon := (1 - \frac{1}{2\lambda}) - c > 0$. We now show that all x with $|x|_1 \leq (1 + \varepsilon)\frac{n}{2}$ satisfy $Trap_z(\widetilde{x}) > f_{layer}$. For this aim we use the fact that $\frac{\varepsilon}{2\lambda} + \frac{2\lambda - 1 - \varepsilon}{2\lambda n} < \varepsilon$ for n large enough. Let $x \in \{0,1\}^n$ with $|x|_1 \leq (1 + \varepsilon)\frac{n}{2}$. Then

$$
\begin{aligned}
\text{TRAP}_z(\widetilde{x}) &= \tfrac{n-1}{z}(z - |x|_1) \\
&\geq \tfrac{n-1}{\lambda n}(\lambda - \tfrac{1+\varepsilon}{2})n \\
&= \tfrac{n-1}{n}(1 - \tfrac{1}{2\lambda} - \tfrac{\varepsilon}{2\lambda})n \\
&= (1 - \tfrac{1}{2\lambda} - \tfrac{\varepsilon}{2\lambda} - \tfrac{2\lambda - 1 - \varepsilon}{2\lambda n})n \\
&> (1 - \tfrac{1}{2\lambda} - \varepsilon)n \\
&= cn.
\end{aligned}
$$

After an expected number of at most em steps the (1+1) EA has determined a solution x with $x_m = 0$. Until this moment all offsprings were accepted as solution by the (1+1) EA, since they all share the same function value f_{layer}. Hence, $\widetilde{x}$ is uniformly distributed in $\{0,1\}^n$. Using Chernoff bounds, the probability that $|x|_1 \leq (1 + \frac{\varepsilon}{2})\frac{n}{2}$ is $1 - e^{-\Omega(n)}$. The only chance for the (1+1) EA to produce an accepted solution y with $|y|_1 > |x|_1$ is a mutation step flipping $\Theta(n)$ 0–bits. The probability for such a step is $n^{-\Omega(n)}$ as already shown in the proof of Theorem 5. Now we are in the situation of the proof of Theorem 5, since the (1+1) EA is not affected by the neutrality henceforth (apart from switching the neutrality bit from time to time). This implies that the (1+1) EA has not determined the optimum in $n^{n/2}$ mutation steps with probability $1 - e^{-\Omega(n)}$. ∎

The performance of the (1+1) EA on the function TRAP_z^* with $f_{layer} = cn$ and $(1 - \frac{1}{2\lambda}) < c < 1$ is the content of the next theorem. $N = \{y \in \Omega : \text{TRAP}_z(y) < f_{layer}\}$ satisfies $\frac{|\Omega \setminus N|}{|\Omega|} = e^{-\Omega(n)}$, if c is in this range. This is the reason, why the (1+1) EA behaves different in this situation. However, the optimization time of the (1+1) EA stays exponential with probability exponentially close to 1.

Theorem 7: If $(1 - \frac{1}{2\lambda}) < c < 1$, the optimization time of the (1+1) EA on the function TRAP_z^* is lower bounded by $e^{\alpha n}$, $\alpha > 0$ a constant, with probability $1 - e^{-\Omega(n)}$.

Proof: Set $\varepsilon := c - (1 - \frac{1}{2\lambda}) > 0$. We prove that all x with $(1 - \varepsilon)\frac{n}{2} \leq |x|_1 \leq z$ satisfy $\text{TRAP}_z(\widetilde{x}) < f_{layer}$. For this aim we use the fact that $\frac{\varepsilon}{2\lambda} + \frac{2\lambda - 1 + \varepsilon}{2\lambda n} < \varepsilon$ for n large enough.

Let $x \in \{0,1\}^n$ with $(1 - \varepsilon)\frac{n}{2} \leq |x|_1 \leq z$. Then

$$
\begin{aligned}
\text{TRAP}_z(\widetilde{x}) &= \tfrac{n-1}{z}(z - |x|_1) \\
&\leq \tfrac{n-1}{\lambda n}(\lambda - \tfrac{1-\varepsilon}{2})n \\
&= \tfrac{n-1}{n}(1 - \tfrac{1}{2\lambda} + \tfrac{\varepsilon}{2\lambda})n \\
&= (1 - \tfrac{1}{2\lambda} + \tfrac{\varepsilon}{2\lambda} + \tfrac{2\lambda - 1 + \varepsilon}{2\lambda n})n \\
&< (1 - \tfrac{1}{2\lambda} + \varepsilon)n \\
&= cn.
\end{aligned}
$$

We already know that $|x|_1 \leq \frac{\lambda + 1/2}{2}n$ holds for the initial solution with probability $1 - e^{-\Omega(n)}$. If $x_m = 1$ we can apply Lemma 2. Since $N = \{y \in \Omega : \text{TRAP}_z(y) < f_{layer}\} \supseteq \{y \in \Omega : (1 - \varepsilon)\frac{n}{2} \leq |x|_1 \leq z\}$, we get $\frac{|\Omega \setminus N|}{|\Omega|} = e^{-\Omega(n)}$ by Chernoff bounds. Hence, Lemma 2 assures that there is a constant $\alpha > 0$ such that after $e^{\alpha n}$ steps of the (1+1) EA the current solution is on the neutrality level and thus not optimal with probability $1 - e^{-\Omega(n)}$.

We now consider the case where the neutrality bit of the initial solution is 0. The behavior is the same as the (1+1) EA on TRAP_z as long as the neutrality bit is not set to 1. Hence, the number of ones can not decrease within this phase with probability $1 - e^{-\Omega(n)}$. If a solution x with $\text{TRAP}_z(x) > f_{layer}$ has been obtained before the neutrality bit is set to one, no solution on the neutrality layer is accepted and we are again in the situation of the (1+1) EA on TRAP_z.

It remains to consider the case where the neutrality bit is set to one before the (1+1) EA has obtained a solution x with $\text{TRAP}_z(x) > f_{layer}$. Let us denote the initial individual of the (1+1) EA by x and the first solution of the (1+1) EA with neutrality bit one by y. With probability $1 - e^{-\Omega(n)}$ we have $x, y \in N$. Moreover, since with probability $1 - e^{-\Omega(n)}$ no mutation steps of the (1+1) EA that increase the number of ones of the current solution were accepted, the probability distributions of $|x|_1$ and $|y|_1$ have the following property. We can get the probability distribution of $|y|_1$ from the one of $|x|_1$ by shifting weight from the right to the left. Let us consider $v, w, \in \{0,1\}^n$ with $|v|_1 \leq |w|_1 < k$ for a $k \leq n$. The probability that the (1+1) EA produces in the next t steps a solution with at least k ones when starting in v is clearly less or equal than the success probability for the (1+1) EA starting in w. Using these two arguments, the probability to reach the optimum in $e^{\alpha n}$ steps, $\alpha > 0$ an appropriate constant, starting from the element y is upper bounded by the probability to reach the optimum when starting in x (the initial solution of the (1+1) EA). This reduces our considerations to the case that the initial solution x satisfies $x_m = 1$ and completes the proof. ∎

V. Neutrality helps

The results of the previous section show that the use of the considered neutrality mechanism is not able to reduce the runtime significantly for the class of deceptive functions considered in [5, 6]. The aim of this section is to point out situations where the use of neutrality can significantly reduce the runtime, i. e. from exponential to polynomial. We consider

a deceptive function with two local optimal search points 0^n and 1^n. In addition there is a polynomial fraction of search points in the search space that have a better fitness value than the two local optima. In the case that such a search point of fitness at least n has been obtained the well-known function LeadingOnes [4] directs the search to a single global optimum. W. l. o. g. we assume that n is even in the following. We investigated the function *PEAK* defined as

$$PEAK(x) := \begin{cases} \frac{n}{2} - |x|_1 & : \ |x|_1 < \frac{n}{2} \\ |x|_1 - \frac{n}{2} & : \ |x|_1 > \frac{n}{2} \\ n + LeadingOnes(x) & : \ |x|_1 = \frac{n}{2}. \end{cases}$$

as well as its extension to neutrality *PEAK**. We define by $P = \{x \in \{0,1\}^n : \sum_{i=1}^n x_i = \frac{n}{2}\}$ the set of search points that are on the peak level. First, we investigate the (1+1) EA on the original function. As long as no search point with fitness value at least n has been obtained the search is mislead to one of the local optima. This has the following consequence.

Theorem 8: The optimization time of the (1+1) EA on *PEAK* is $\Omega((\frac{n}{4})^{n/2})$ with probability $1 - o(1)$.

Proof: W. l. o. g. we assume the n is even. The number of bit strings whose number of ones is exactly $n/2$ is

$$\binom{n}{n/2} \sim \sqrt{\frac{2}{\pi}} n^{-1/2} 2^n$$

using Stirling's formula.

Hence, the initial solution consists of exactly $n/2$ ones with probability

$$\binom{n}{n/2} \cdot 2^{-n} = \Theta(n^{-1/2}),$$

and the probability that the number of ones lies in the interval $[n/2 - n^{1/4}, n/2 + n^{1/4}]$ is $O(n^{1/4} n^{-1/2}) = O(n^{-1/4})$.

We consider a phase of $n^{3/2}$ steps under the condition that the initial solution has Hamming distance at least $n^{1/4}$ to any solution with exactly $n/2$ ones. The number of flipping bits in a single mutation step is asymptotically Poisson-distributed and therefore the probability of having a step with $n^{1/4}$ flipping bits within this phase is $o(1)$. This implies that the smallest Hamming distance to a solution with exactly $n/2$ ones does not decrease within this phase with probability $1 - o(1)$. Let x be such a solution whose fitness value is k. The probability of achieving a solution with a higher fitness value is at least

$$\left(\frac{n/2 - k}{n} \right) (1 - 1/n)^{n-1} \geq \frac{n/2 - k}{en}$$

as there are at least $n/2 - k$ 1-bit flips which lead to an improvement. This implies that a solution with fitness value exactly $n/2$ (one of the search points 0^n and 1^n) has been obtained after an expected number of $O(n \log n)$ steps. The probability that such a solution has not been obtained within a phase of $n^{3/2}$ steps is $o(1)$ using Markov's inequality. Afterwards, the probability of producing from one of the

search points 0^n of 1^n a solution with exactly $n/2$ is upper bounded by

$$\binom{n}{n/2} (1/n)^{n/2} \cdot (1 - 1/n)^{n/2} = O\left(\left(\frac{n}{4} \right)^{-n/2} \right).$$

All failure probabilities to do not obtain the search point 0^n or 1^n are $o(1)$, which implies that the optimization time is $\Omega((\frac{n}{4})^{n/2})$ with probability $1 - o(1)$. ∎

While the optimization is exponential with probability asymptotically close to 1, a polynomial optimization can be proven when neutrality is used. In this case the neutrality layer may prevent the algorithm from being mislead into the direction of a local optimum.

Theorem 9: Let $\varepsilon > 0$. Choosing $f_{layer} = cn, 0 < c < 1/2$ a constant, the optimization time of the (1+1) EA on *PEAK** is $O(n^2 \log n)$ with probability $1/2 - \varepsilon$.

Proof: With probability $1/2$ the initial solution satisfies $x_m = 1$ and therefore $PEAK^*(x) = f_{layer}$. The probability to produce a solution z with $PEAK(z) \geq f_{layer}$ is $e^{-\Omega(n)}$ which also holds for a polynomial number of steps. The probability to obtain a solution z with $PEAK(z) \geq n$ is $\Theta(n^{-1/2})$ and the probability of flipping the bit z_m in this step is $1/m$ if the failure has not happened. Using Markov's inequality such a solution has been obtained within n^2 steps with probability $1 - o(1)$.

It remains to consider the time until a solution z with exactly $n/2$ leading ones has been obtained. Note, that no solution z with $z_m = 1$ is accepted if we have obtained for the first time a solution with exactly $n/2$ ones. Let k be the number of leading ones in the current solution x. An improvement can be achieved by flipping the bit x_{k+1} of x and one of the $n/2 - k$ 1-bits not contributing to the leading ones values. Hence, the probability for an improvement is at least

$$\frac{1}{n} \cdot \frac{n/2 - k}{n} (1 - 1/n)^{n-2} \geq \frac{n/2 - k}{en^2}$$

The expected time until a LeadingOnes-value of $n/2$ has been obtained is $O(n^2 \log n)$ by summing up the different waiting times to achieve an improvement. Hence, using Markov's inequality $O(n^2 \log n)$ steps are enough with probability $1 - \varepsilon$. After having chosen $x_m = 1$ for the initial solution x all failure probabilities have been bounded by $o(1)$, which proves the theorem. ∎

For the fitness layer values investigated in Theorem 9 there is only a constant probability that the search is successful. However a failure with another constant probability obtaining one of the local optima is still possible. This would imply an expected exponential optimization time. We can prevent such an inefficient behavior by choosing a larger value of f_{layer} which leads to an expected polynomial optimization time.

Theorem 10: Choosing $f_{layer} = cn, 1/2 < c < 1$ a constant, the expected optimization time of the (1+1) EA on *PEAK** is $O(n^2 \log n)$.

Proof: We consider the following three phases: (i) the phase until the first individual with set neutrality bit is produced, (ii) the phase until an individual on the peak level P is

determined *and* the neutrality bit is unset, and (iii) the phase of optimizing the leading ones problem on the peak level.

In phase (i) the neutrality bit has to be set to one, which happens after an expected number of $O(n)$ mutation steps. The main problem in phase (ii) is, that we cannot assume a uniform distribution of the current individual, since in the first phase (until an individual with $x_m = 1$ is produced) the (1+1) EA guides the current individual either to the string 0^n or to the string 1^n. In the following, we show that the current individual is almost uniformly distributed in $\{0,1\}^n$ after $\Theta(n \log n)$ mutation steps by analyzing the mixing time of the (1+1) EA. Let x denote the first individual with $x_m = 1$ and let y be the N's individual with neutrality bit set after x is produced, where $N \geq n \log n$. For every bit of the y_i we upper bound the difference $\mathrm{Prob}(y_i = x_i) - \mathrm{Prob}(y_i \neq x_i)$. We have

$$
\begin{aligned}
&\mathrm{Prob}(y_i = x_i) - \mathrm{Prob}(y_i \neq x_i) \\
&= \sum_{\substack{i=0 \\ 2|i}}^{N} \left(\tfrac{1}{n}\right)^i \left(1 - \tfrac{1}{n}\right)^{N-i} - \sum_{\substack{i=0 \\ 2|(i+1)}}^{N} \left(\tfrac{1}{n}\right)^i \left(1 - \tfrac{1}{n}\right)^{N-i} \\
&= \sum_{i=0}^{N} \left(-\tfrac{1}{n}\right)^i \left(1 - \tfrac{1}{n}\right)^{N-i} = \left(-\tfrac{1}{n} + \left(1 - \tfrac{1}{n}\right)\right)^N \\
&\leq \left(1 - \tfrac{2}{n}\right)^{n \log n} \leq e^{-2 \log n} = \tfrac{1}{n^2}.
\end{aligned}
$$

Using $\mathrm{Prob}(y_i = x_i) + \mathrm{Prob}(y_i \neq x_i) = 1$, we obtain $\mathrm{Prob}(y_i = 1) \geq \frac{1}{2}(1 - \frac{1}{n^2})$ and $\mathrm{Prob}(y_i = 0) \geq \frac{1}{2}(1 - \frac{1}{n^2})$, respectively. Thus, we get for every $z \in \{0,1\}^n$

$$
\mathrm{Prob}(y = z) \geq \left[\tfrac{1}{2}(1 - \tfrac{1}{n^2})\right]^n \geq \tfrac{1}{e}\tfrac{1}{2^n}.
$$

Hence, after $n \log n$ steps or more, independent of the starting vertex, the probability to be on a particular vertex is $\Theta(2^{-n})$. Therefore, the probability for an individual to be on the peak level P is $\Theta(|P|2^{-n}) = \Theta(n^{-1/2})$ after this number of steps. The probability of turning off neutrality in a step that produces a solution of P is $1/n$. Hence, an expected number of $O(n^{3/2})$ steps suffices to produce a solution $x \in P$ where $x_m = 0$ holds.

The optimal solution of P is obtained afterwards within $O(n^2 \log n)$ steps as shown in the proof of Theorem 9. ∎

VI. Conclusions

The influence of neutrality has been considered in several experimental studies. We have carried out the first rigorous runtime analysis of an evolutionary algorithms using neutrality. The model of neutrality that has been subject of this paper has already been considered in [5, 6]. We have shown that there is no significant advantage for using neutrality for the function ONEMAX and the class of deceptive functions investigated in these papers (i.e. in the case of the proposed deceptive functions, the expected optimization time is exponential regardless of the value of the fitness layer). In contrast to this, we have pointed out that neutrality may be helpful for deceptive functions when the number of search points being nearly optimal is at least a polynomial fraction of the search space.

The functions considered in this paper are artificial ones which should make clear some effects neutrality might have on the runtime of evolutionary algorithms. For future work, it would be interesting to see whether a positive effect of neutrality with respect to the runtime can also be proven for some real-world problem (e. g. a particular combinatorial optimization problem).

References

[1] M. Collins. Finding needles in haystacks is harder with neutrality. In *Proc. of GECCO '05*, pages 1613–1618. ACM Press, 2005.

[2] S. Droste. Analysis of the (1+1) ea for a dynamically bitwise changing onemax. In *Proc. of GECCO '03*, volume 2724 of *LNCS*, pages 909–921. Springer, 2003.

[3] S. Droste. Not all linear functions are equally difficult for the compact genetic algorithm. In *Proc. of GECCO '05*, pages 679–686. ACM, 2005.

[4] S. Droste, T. Jansen, and I. Wegener. On the analysis of the (1+1) evolutionary algorithm. *Theor. Comput. Sci.*, 276:51–81, 2002.

[5] E. Galván-López and R. Poli. An empirical investigation of how and why neutrality affects evolutionary search. In *Proc. of GECCO '06*, pages 1149–1156. ACM Press, 2006.

[6] E. Galván-López and R. Poli. Some steps towards understanding how neutrality affects evolutionary search. In *Proc. of PPSN '06*, volume 4193 of *LNCS*, pages 778–787, 2006.

[7] O. Giel and I. Wegener. Evolutionary algorithms and the maximum matching problem. In *Proc. of STACS '03*, volume 2607 of *LNCS*, pages 415–426. Springer, 2003.

[8] T. Jansen and I. Wegener. Evolutionary algorithms - how to cope with plateaus of constant fitness and when to reject strings of the same fitness. *IEEE Trans. Evolutionary Computation*, 5(6):589–599, 2001.

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

[10] H. Mühlenbein. How genetic algorithms really work: mutation and hillclimbing. In *Proc. of PPSN '92*, pages 15–26. Elsevier, 1992.

[11] F. Neumann and I. Wegener. Randomized local search, evolutionary algorithms, and the minimum spanning tree problem. In *Proc. of GECCO '04*, volume 3102 of *LNCS*, pages 713–724. Springer, 2004.

[12] F. Neumann and C. Witt. Runtime analysis of a simple ant colony optimization algorithm. In *Proc. of ISAAC '06*, volume 4288 of *LNCS*, pages 618–627. Springer, 2006.

[13] I. Pak and V. H. Vu. On mixing of certain random walks, cutoff phenomenon and sharp threshold of random matroid processes. *Discrete Applied Mathematics*, 110 (2-3):251–272, 2001.

[14] A. M. Raich and J. Ghaboussi. Implicit representation in genetic algorithms using redundancy. *Evolutionary Computation*, 2(2):277–302, 1997.

[15] F. Rothlauf. Population sizing for the redundant trivial voting mapping. In *Proc. of GECCO '03*, volume 2724 of *LNCS*, pages 618–627. Springer, 2003.

[16] G. Rudolph. How mutation and selection solve long path problems in polynomial expected time. *Evolutionary Computation*, 4(2):195–205, 1996.

[17] T. Storch. On the choice of the population size. In *Proc. of GECCO '04*, volume 3102 of *LNCS*, pages 748–760. Springer, 2004.

[18] M. Toussaint and C. Igel. Neutrality and self-adaptation. *Natural Computing*, 2(2):117–132, 2003.

[19] K. Weicker and N. Weicker. Burden and benefits of redundancy. In *Proc. of FOGA '00*, pages 313–333. Morgan Kaufmann, 2001.

[20] C. Witt. An analysis of the $(\mu+1)$ ea on simple pseudo-boolean functions. In *Proc. of GECCO '04*, volume 3102 of *LNCS*, pages 761–773. Springer, 2004.

[21] C. Witt. Worst-case and average-case approximations by simple randomized search heuristics. In *Proc. of STACS '05*, volume 3404 of *LNCS*, pages 44–56. Springer, 2005.