On the Analysis of the (1+1) Memetic Algorithm

Dirk Sudholt

Nr. CI-201/06

Interner Bericht ISSN 1433-3325 Januar 2006

On the Analysis of the (1+1) Memetic Algorithm

Dirk Sudholt*

FB Informatik, Univ. Dortmund, 44221 Dortmund, Germany
Dirk.Sudholt@udo.edu

Abstract

Memetic algorithms are evolutionary algorithms incorporating local search to increase exploitation. This hybridization has been fruitful in countless applications. However, theory on memetic algorithms is still in its infancy.

Here, we introduce a simple memetic algorithm, the (1+1) Memetic Algorithm ((1+1) MA), working with a population size of 1 and no crossover. We compare it with the well-known (1+1) EA and randomized local search and show that these three algorithms can outperform each other drastically.

On problems like, e. g., long path problems it is essential to limit the duration of local search. We investigate the (1+1) MA with a fixed maximal local search duration and define a class of fitness functions where a small variation of the local search duration has a large impact on the performance of the (1+1) MA.

All results are proved rigorously without assumptions.

*supported by the Deutsche Forschungsgemeinschaft (DFG) as part of the Collaborative Research Center "Computational Intelligence" (SFB 531).

1 Introduction

Memetic algorithm is a term for a hybridization of evolutionary algorithms and local search. Combining these two types of algorithms, the hope is to preserve good properties of two algorithms and to create a more powerful hybrid algorithm. This approach has led to many implementations of memetic algorithms that have shown their usefulness in many applications, see Moscato [7] for an overview.

Due to the rapid development in this research area, it is hard for theory to keep up with the state-of-the-art. An example of a theoretical analysis on memetic algorithms is the work of Merz [6] based on an empirical approach. However, rigorous theoretical results are rare and difficult to obtain.

The purpose of this paper is to build a rigorous theory on memetic algorithms, i. e., a theory not based on assumptions. Such a theory has been established for simple evolutionary algorithms such as the (1+1) EA (see, e. g., Droste, Jansen, and Wegener [1]) and for population-based evolutionary algorithms (see Witt [9] for a rigorous analysis of a (μ+1) EA and Jansen and Wegener [5] for a rigorous analysis of genetic algorithms with crossover).

When dealing with a new class of algorithms, it makes sense to start with a simple algorithm. Thus, we define the so-called (1+1) Memetic Algorithm ((1+1) MA) working with a population of size 1 and no crossover. We will see that the analysis of this simple memetic algorithm provides valuable insights into the interaction of mutation and local search. Furthermore, this work can be used as a basis for results on more complex algorithms.

The fitness functions considered in this work are artificial functions that have been constructed explicitly to prove specific properties of the (1+1) MA. This approach has shown to be fruitful in the past since it is possible to enforce a certain behavior of the considered algorithm with a custom-built fitness function. That way the investigation of constructed example functions may disprove simple conjectures on the considered algorithm (see, e. g., Jansen and Wegener [4]) and yield various insights into the behavior of the algorithm (see, e. g., Garnier, Kallel, and Schoenauer [2]).

This paper is structured as follows. In Section 2 we define the three algorithms considered here: the (1+1) MA, the (1+1) EA, and randomized local search. Section 3 introduces a construction plan that is used in Sections 4 and 5 to define functions with specific properties. In Section 4 we define three functions and prove that each algorithm beats the other two algorithms drastically on one function. In Section 5 we consider the (1+1) MA with maximal local search duration $d(n)$. A class of functions is presented where only specific values of $d(n)$ for the (1+1) MA lead to an efficient optimization while even small deviations from these values are likely to lure the (1+1) MA into a trap.

2 Definitions

First, we will define the algorithms considered in this work. All algorithms are defined for the maximization of a function $f\colon \{0,1\}^n \to \mathbb{R} \cup \{-\infty\}$.

Randomized local search (RLS) chooses a Hamming neighbor of the current search point uniformly at random. The offspring replaces its parent if its fitness is not worse.

Algorithm 1 (RLS)**.**

```
1. Choose $x \in \{0,1\}^n$ uniformly at random.
2. $y := x$. Flip a bit in $y$ chosen uniformly at random.
3. If $f(y) \geq f(x)$, $x := y$.
4. If the stopping criterion is not met, continue at line 2.
```

We do not explicitly define a stopping criterion. Instead, we consider the algorithm as an infinite stochastic process since we are only interested in the random variable describing the time until a global optimum is found.

The (1+1) EA almost works like RLS. The only difference is the mutation operator flipping each bit with a fixed mutation probability p_m.

Algorithm 2 ((1+1) EA)**.**

```
1. Choose $x \in \{0,1\}^n$ uniformly at random.
2. $y := x$. Flip every bit in $y$ independently with probability $p_m$.
```

```
3. If $f(y) \geq f(x)$, $x := y$.

4. If the stopping criterion is not met, continue at line 2.
```

The (1+1) MA uses a local search procedure defined as follows. $\mathrm{H}(x, y)$ denotes the Hamming distance of two search points $x, y \in \{0,1\}^n$.

Procedure 1 (Local Search(y))**.**

```
    $S := \{z \mid H(z,y) = 1 \wedge f(z) > f(y)\}$.
    While $S \neq \emptyset$ do {
        Choose $y \in S$ uniformly at random.
        $S := \{z \mid H(z,y) = 1 \wedge f(z) > f(y)\}$.
    }
    Return $y$.
```

The difference between the local search strategies of RLS and the (1+1) MA's local search procedure is that in a region of search points with equal fitness, a so-called *plateau*, RLS can move around since search points with equal fitness are accepted. The (1+1) MA's local search procedure, however, stops since only search points with larger fitness are accepted.

Having defined the local search procedure, we can now define the (1+1) MA.

Algorithm 3 ((1+1) Memetic Algorithm ((1+1) MA))**.**

```
1. Choose $x \in \{0,1\}^n$ uniformly at random.
   $x :=$ Local Search($x$).

2. $y := x$. Flip every bit in $y$ independently with probability $p_m$.
   $y :=$ Local Search($y$).

3. If $f(y) \geq f(x)$, $x := y$.

4. If the stopping criterion is not met, continue at line 2.
```

A common choice for p_m is $p_m := 1/n$ implying that one bit flips in expectance. In the following, we will investigate values for p_m with $\varepsilon/n \leq p_m \leq 1/2$ for some constant $0 < \varepsilon \leq 1$. Obviously, values larger than 1/2 do not make sense since individuals with a large Hamming distance to the current search point are preferred. If the mutation probability is very small, say

$p_m = o(1/n)$, then most mutations will not flip any bit in the parent. While RLS and the (1+1) MA can use local search strategies to have progress, the (1+1) EA has a large waiting time until it finds a new search point. Thus, it makes sense to restrict p_m to values $p_m \geq \varepsilon/n$.

Next, we define notions to classify the performance of an algorithm.

Definition 1. *An event E occurs with overwhelming probability if the probability $\mathit{Prob}\,(E)$ converges to 1 exponentially fast with the search space dimension n, i. e., $\mathit{Prob}\,(E) = 1 - 2^{-\Omega(n^{\varepsilon})}$ for some constant $\varepsilon > 0$.*

We say that an algorithm $\mathcal{A}$ is efficient *on a function f iff $\mathcal{A}$ finds a global optimum on f in polynomial time with overwhelming probability.*

We say that an algorithm $\mathcal{A}$ fails *on a function f iff $\mathcal{A}$ does not find a global optimum in exponential time with overwhelming probability.*

For the (1+1) MA, we have to take into account all steps of the local search process. Note that the (1+1) MA's local search procedure performs n fitness evaluations per step while (1+1) EA and RLS use only one fitness evaluation per step. Our notions of efficiency are not affected by this additional factor n, though. Even polynomially large factors do not affect these notions as we do not differentiate between polynomials of different orders. This perspective resembles the one taken in complexity theory and is explicit in the notion of NP-completeness and the Church-Turing thesis.

In Sections 4 and 5 we will define functions where certain subfunctions are embedded. Amongst others, we will make use of so-called *long path problems* introduced by Horn, Goldberg, and Deb [3]. A long path of Hamming neighbors is embedded into a fitness function such that on the path the fitness is strictly increasing and all other search points give hints to reach the start of the path.

Long paths can be generalized to so-called long K-paths where the probability to take a shortcut on the path by mutation decreases rapidly with increasing K. This generalization was informally described in [3] and then formally defined by Rudolph [8].

Definition 2. *Let $K, N \in \mathbb{N}$ with $(N-1)/K \in \mathbb{N}$. The long K-path of dimension N is a sequence of bitstrings from $\{0,1\}^N$ defined recursively* *as follows. The long K-path of dimension 1 is defined as $P_1^K := (0, 1)$. Let $P_{N-K}^K = (v_1, \dots, v_\ell)$ be the long K-path of dimension $N - K$. Then the long K-path of dimension N is defined by prepending K bits to the search points from $\{v_1, \dots, v_\ell\}$: let $S_0 := (0^K v_1, 0^K v_2, \dots, 0^K v_\ell)$, $S_1 := (1^K v_\ell, 1^K v_{\ell-1}, \dots, 1^K v_1)$, and $B := (0^{K-1}1v_\ell, 0^{K-2}1^2 v_\ell, \dots, 01^{K-1}v_\ell)$. The search points in S_0 and S_1 differ in the K leading bits and the search points in B represent a bridge between them. The long K-path of dimension N is constructed by concatenating S_0, B, and S_1.*

We cite two important properties of long K-paths that will be referred to in Section 5. Proofs are given in [1].

Lemma 1.

1. *The length of the long K-path is $(K + 1)2^{(N-1)/K} - K + 1$. (We will often choose $N := (K^2 + 1)$ yielding an exponential length.) All points on the path are different.*

2. *Let x be a point on the long path. For all $0 < i < K$ the following holds. If x has at least i successors on the path, then the i-th successor has Hamming distance i of x and all other successors of x on the path have Hamming distances different from i.*

The second property is important, stating that unless an algorithm performs jumps of at least K bits, the algorithm cannot take shortcuts on the long K-path.

3 A Common Construction Plan

The functions defined in Sections 4 and 5 are constructed by a common construction plan. A function g defined on a bitstring of length N is to be maximized and we want the optimization of g to start with a bitstring containing almost only zeros. Hence, we embed the function g into a function f. The function f is defined on a larger bitstring where N bits are appended to the bits of g.

The following function ZZO (zeros, zeros, ones) gives hints to turn the first N bits into zeros and the last N bits into ones.

Definition 3. *Let $|x|_i$ be the number of bits with value i in x. Let $x \in \{0,1\}^n$ be divided into two halves, $x = x'x''$ with $x', x'' \in \{0,1\}^N$. Then we define* $\mathrm{ZZO}\colon \{0,1\}^n \to \mathbb{R}$ *as*

$$\mathrm{ZZO}(x) := \begin{cases} |x''|_0 - 3N & \text{if } x' \neq 0^N, x'' \neq 0^N \\ |x'|_0 - 2N & \text{if } x' \neq 0^N, x'' = 0^N \\ |x''|_1 - N & \text{if } x' = 0^N. \end{cases}$$

In a typical run of one of the algorithms considered here, the algorithm starts with an initial search point x where $x'' \neq 0^N$ and then maximizes the number of zeros in x''. Once $x'' = 0^N$, the number of zeros in x' is maximized and once $x' = 0^N$, the number of ones in x'' is maximized. Note that ZZO is *unimodal*, i. e., there is a path of Hamming neighbors with strictly increasing fitness that leads to the global optimum 0^N1^N.

Now we embed the function ZZO into a function such that the fitness depends on the function g once the right half contains only ones. ZZO gives hints to maximize the number of ones in the right half only if the left half consists of zeros. Thus, the first search point where g comes into play is likely to contain (almost) only zeros.

Lemma 2. *Let $x \in \{0,1\}^n$ be divided into two halves, $x = x'x''$ with $x', x'' \in \{0,1\}^N$. Let $g\colon \{0,1\}^N \to \mathbb{R}^+ \cup \{-\infty\}$ be an arbitrary function and $f\colon \{0,1\}^n \to \mathbb{R} \cup \{-\infty\}$ be defined as*

$$f(x) := \begin{cases} \mathrm{ZZO}(x) & \text{if } x'' \neq 1^N \\ g(x') & \text{if } x'' = 1^N. \end{cases}$$

A search point x is called well-formed *iff its right half contains only ones. Consider an algorithm $\mathcal{A} \in \{$RLS, (1+1) EA, (1+1) MA$\}$ on f with mutation probability $p_m := \Theta(1/n)$. Then with overwhelming probability the following statements hold.*

1. *The embedded function g is first evaluated with a well-formed search point $x = x'1^N$ where either $x' = 0^N$ or x' is a mutant of 0^N.*
2. *If, in addition, $g(0^N) > -\infty$, the algorithm accepts some well-formed search point $y = y'1^N$ within $O(n^2)$ fitness evaluations.*

Proof. We investigate a typical run of the algorithm and show that the claimed events occur in a typical run. Events preventing a run from being typical are called errors. Showing that the sum of all error probabilities is exponentially small completes the proof.

By Chernoff's bounds, $\mathcal{A}$ is initialized with some search point $x = x'x''$ where $|x''|_1 \leq 2N/3$ with overwhelming probability. (If the complementary event occurs, this is considered an error and the run is not typical.) The probability that at least $N/3 = \Omega(n)$ bits flip in one single mutation is exponentially small. The probability that this happens at least once in cn^2 steps for some $c > 0$ is still exponentially small and this is considered an error, too.

If $|x''|_1 \leq 2N/3$, the only search points with a larger number of ones in the right half that can be reached by local search or mutations of less than $N/3$ bits contain only zeros in the left half. Once the left half consists of zeros, these values must be maintained until a well-formed search point is reached. Now, consider the variation step creating the first well-formed offspring $y = y'1^N$. Then y' results from 0^N by the very same variation, thus either $y' = 0^N$ or y' is a mutant of 0^N.

We now prove that the time bound cn^2 holds with overwhelming probability. The (1+1) MA reaches a search point $x = 0^N x''$ with $|x''|_1 \geq N - 1$ during the initial local search process within $(c-1)n^2$ fitness evaluations if $c \geq 5/2$. For RLS and the (1+1) EA, the probability to increase the fitness is bounded below by $\Omega(1/n)$ if $f(x) < -1$ for the current population x. By Chernoff's bounds, the probability to have less than $3N - 1$ fitness-increasing steps in $(c-1)n^2$ steps is exponentially small if $c > 0$ is chosen large enough. Thus, all three algorithms reach either $x = 0^N x''$ with $|x''|_1 = N - 1$ or some well-formed search point in $(c-1)n^2$ fitness evaluations with overwhelming probability.

If the current population is $x = 0^N x''$ with $|x''|_1 = N - 1$, the probability to create its well-formed Hamming neighbor $0^N 1^N$ as an offspring is $\Omega(1/n)$ for all three algorithms. Thus, as long as no other well-formed search point is accepted, the probability of creating $y = 0^N 1^N$ in one step is $\Omega(1/n)$ and the probability that y is created in n^2 trials is $1 - (1 - \Omega(1/n))^{n^2} = 1 - e^{-\Omega(n)}$. This proves the time bound cn^2 and the first statement. Since $0^N 1^N$ has a non-negative fitness if $g(0^N) > -\infty$, we have also shown that a well-formed search point is accepted by the algorithm in cn^2 fitness evaluations with overwhelming probability.

It is easy to see that the sum of all error probabilities is exponentially small. □

The first well-formed search point reached by the (1+1) MA typically contains many zeros in the left half. This is due to the structure of ZZO and the random initialization. The (1+1) MA's local search procedure, however, can behave differently since it is initialized with a mutant of the (1+1) MA's current population.

Consider a step where the (1+1) MA mutates some well-formed search point $x = x'1^N$ and creates an offspring $y = y'y''$ with $|y''|_1 = N - 1$. The fitness of y is determined by $\mathrm{ZZO}(y)$ and local search is started from y. Then the (1+1) MA's local search procedure can only increase the fitness by two means: either by increasing the number of zeros in y'' or by flipping back the only zero-bit in y'' if $g(y') > -\infty$. In the latter case, a well-formed search point $y'1^N$ with non-negative fitness is reached. In the former case or in case $|y''|_1 < N - 1$, the Hamming distance to all well-formed search points is at least 2 and the only fitness-increasing path leading to a well-formed search point traverses 0^N1^N. Lastly, if $y'' = 1^N$ and $g(y') = -\infty$, either $z = z'1^N$ is reached with non-negative fitness where z' is a Hamming neighbor of y' or a bit flips in the right half and 0^N1^N is traversed.

We summarize these arguments in the following corollary. With regard to Section 5 we also consider the case that the (1+1) MA can stop the local search process at some point of time.

Corollary 1. *Consider the (1+1) MA on a function f as defined in Lemma 2. If the (1+1) MA performs a mutation of a well-formed search point x with non-negative fitness creating $y = y'y''$, then LocalSearch(y) either*

- *stops with a search point with negative fitness or*
- *traverses the search point 0^N1^N or*
- *reaches a search point $z = z'1^N$ with $H(z', y') \leq 1$ and non-negative fitness.*

In the first case the result of the local search process is rejected by the (1+1) MA. In the second case, we are in the situation of a typical run as described in the proof of Lemma 2. The third case yields a well-formed search point where the left half is almost a mutant of x'. Hence, the outcome of the local search process and the following selection is restricted to search points on a fitness-increasing path starting with either $0^N 1^N$ or $z = z' 1^N$.

4 RLS, (1+1) EA, and (1+1) MA Can Beat Each Other Drastically

We now present three functions showing that each algorithm can beat the other two algorithms drastically. W. l. o. g. n is a multiple of 4 and $N := n/2$. Then we divide the bitstring of x into two halves of length N, $x = x'x''$ with $x', x'' \in \{0,1\}^N$.

Let $T \subset \{0,1\}^N$ be defined as

$$T := \{x' \mid |x'|_1 \le N/2, \forall 0 \le i \le N \colon \mathrm{H}(x', 1^i 0^{N-i}) \ge 2\}.$$

A common idea behind the three functions is the following. Using the construction defined in Section 3, the optimization of some subfunction in the x' part starts close to 0^N. Concentrating on the subspace defined by the x' part, there is a ridge of search points $1^i 0^{N-i}$ where the fitness increases with i. Close to this ridge, but with Hamming distance of at least 2, there is a set T of search points with larger fitness that can be either a trap or a target area of global optima.

First, we define the function f_{MA} where only the (1+1) MA is efficient.

$$f_{\mathrm{MA}}(x) := \begin{cases} \mathrm{ZZO}(x) & \text{if } x'' \ne 1^N \\ i & \text{if } x'' = 1^N, x' = 1^i 0^{N-i}, i \ne N-1 \\ N - \frac{1}{2} & \text{if } x'' = 1^N, x' \in T \\ -\infty & \text{otherwise.} \end{cases}$$

The idea behind this function is that local search allows the (1+1) MA to walk past the trap T and mutation allows the (1+1) MA to jump across a small gap in the ridge. This gap prevents RLS from reaching the global optimum while the (1+1) EA is very likely to run into the trap by mutations flipping more than one bit.

The function f_{EA} is defined such that only the (1+1) EA is efficient.

$$f_{\mathrm{EA}}(x) := \begin{cases} \mathrm{ZZO}(x) & \text{if } x'' \neq 1^N \\ i & \text{if } x'' = 1^N, x' = 1^i 0^{N-i} \\ N + \frac{1}{2} & \text{if } x'' = 1^N, x' \in T \\ -\infty & \text{otherwise.} \end{cases}$$

Here T is not a trap but a target area of global optima. Both RLS and the (1+1) MA are likely to run into the local optimum 1^N while the (1+1) EA can reach T easily by mutations flipping more than one bit.

Finally, the function f_{RLS} is efficient only for RLS.

$$f_{\mathrm{RLS}}(x) := \begin{cases} \mathrm{ZZO}(x) & \text{if } x'' \neq 1^N \\ 2j & \text{if } x'' = 1^N, x' \in \{1^{2j}0^{N-2j}, 1^{2j+1}0^{N-2j-1}\} \\ N - \frac{1}{2} & \text{if } x'' = 1^N, x' \in T \\ -\infty & \text{otherwise.} \end{cases}$$

Here the idea is that RLS walks past the trap T while mutation is likely to lead the (1+1) EA into the trap. For the (1+1) MA, small plateaus in the ridge stop local search processes and mutation leads the (1+1) MA into the trap.

Now, we want to concretize these ideas and give rigorous proofs for these claims.

Theorem 1. *While the (1+1) MA with mutation probability $p_m := 1/n$ is efficient on f_{MA}, both RLS and the (1+1) EA with mutation probability $\varepsilon/n \leq p_m \leq 1/2$, $\varepsilon > 0$, fail on f_{MA}.*

Proof. By Lemma 2 and due to the fact that local search is performed right after initialization, the (1+1) MA reaches the search point $0^N 1^N$ with overwhelming probability. Due to the construction of f_{MA}, for every well-formed search point x with $x' = 1^i 0^{N-i}$ and $i < N - 2$ there is exactly one Hamming neighbor with larger fitness. Thus, the (1+1) MA reaches $x' = 1^{N-2}0^2$ during the initial local search process within $O(n^2)$ fitness evaluations.

Then, local search processes climbing the ridge stop due to the gap in the ridge. Moreover, it is unlikely that the trap T is hit by mutation and local search since by Corollary 1 at least $N/2-3$ bits have to flip in one mutation. The probability that this happens at least once in $O(n^2)$ trials is exponentially small.

A sufficient condition to increase the fitness is to mutate $x = 1^{N-2}0^2 1^N$ into $y = 1^N 1^N$. The probability for this event is $1/n^2 \cdot (1-1/n)^{n-2} \geq 1/(en^2)$. The probability that this does not happen within n^3 mutations is exponentially small.

If the (1+1) MA's local search procedure is initialized with a non-well-formed search point, it may spend time to optimize ZZO and/or to climb the ridge again. Since f_{MA} has $O(n)$ different fitness values, our time bound increases by a factor of $O(n^2)$ yielding a bound of $O(n^5)$. Since the sum of all error probabilities is exponentially small, we have shown that the (1+1) MA is efficient on f_{MA}.

By Lemma 2, RLS reaches the search point $0^N 1^N$ with overwhelming probability. Then the only fitness-increasing path of Hamming neighbors ends with $x' = 1^{N-2}0^2$ and the global optimum cannot be reached by variation steps flipping only one single bit. Thus, with overwhelming probability RLS does not find the optimum in an exponential number of steps.

Finally, we have to show that the (1+1) EA fails on f_{MA}. With probability $1-2^{-N}$, $x'' \neq 1^N$ holds for the initial search point x. We now distinct two cases according to the value of the mutation probability p_m.

If $n^{-1/2} \leq p_m \leq 1/2$, we concentrate on the last N bits in the bit string and remark that these bits must be turned into ones to reach the optimum. The probability to mutate an arbitrary bit string x'' of length N into the string 1^N is bounded above by

$$(\max\{p_m, 1-p_m\})^N \leq \left(1 - \frac{1}{n^{1/2}}\right)^{n/2} \leq e^{-n^{1/2}/2}$$

and so the (1+1) EA fails in this case.

Let $\varepsilon/n \leq p_m < n^{-1/2}$. The probability to flip at least cN bits in one mutation, $0 < c < 1/12$, is at most

$$\binom{N}{cN} \cdot p_m^{cN} \leq 2^N \cdot n^{-\Omega(n)} = n^{-\Omega(n)}.$$

The probability that such an event occurs in $2^{o(n)}$ steps is still of order $n^{-\Omega(n)}$, so we can assume that such an event does not occur in $2^{o(n)}$ steps and introduce only an exponentially small error probability.

Using this result, we remark that the first statement of Lemma 2 also holds in this case: the probability of flipping at least $N/3$ bits in at least one of $2^{o(n)}$ mutations exponentially small and a well-formed search point is created in time $2^{O(n^{1/2})}$ with overwhelming probability.

Let $x = x'1^N$ be the first well-formed search point reached by the (1+1) EA. Since $f_{\mathrm{EA}}(x) > -\infty$, either $x' \in T$ or x' belongs to the ridge implying $x' = 1^i0^{N-i}$ and $i \leq cN$.

Let y be a mutant of x and $k := \mathrm{H}(x, y)$. We estimate the conditional probability of $y' \in T$ given that $x' = 1^i0^{N-i}$ with $i \leq N/2 - cN$ and $k < cN$. In the subspace induced by the N bits of x', among all search points with i one-bits, $i \leq N/2$, there are at most $O(N)$ search points not in T. For symmetry reasons, y' is distributed uniformly at random among all individuals with Hamming distance k from x', $2 \leq k \leq cN$. The probability that $k \geq 2$ is at least $\binom{n}{2}p_m^2 = \Omega(1)$ since $p_m \geq \varepsilon/n$. Then, since $x' = 1^i0^{N-i}$ with $i \leq N/2 - cN$, the probability of $y' \notin T$ is $O(kN)/\binom{N}{k}$. This term is maximal for $k = 2$ if $2 \leq k < cN$. Altogether,

$$\mathrm{Prob}\,(y' \in T) \geq \mathrm{Prob}\,(k \geq 2) \cdot (1 - \mathrm{Prob}\,(y' \notin T \mid k)) = \Omega(1).$$

We now show that with overwhelming probability the algorithm spends at least $n^{3/2}/8$ steps near the trap. To have progress on the ridge, it is necessary that the leftmost zero-bits flip. These bits are called *relevant zero-bits*. We consider a random 0-1-string of infinite length where a position is set to 1 with probability p_m. Then we can interpret this string as follows. The number of ones between the $(i-1)$-th zero and the i-th zero, $i \geq 1$, is the number of flipping relevant zero-bits in the i-th step on the path. If there are less than $n/6$ ones among the first $n^{3/2}/8 + n/6$ bits, the number of flipping relevant zero-bits in $n^{3/2}/8$ steps is smaller than $n/6$ and the progress on the ridge is at most $n/6$. Since the expected number of ones in $n^{3/2}/8 + n/6$ bits is $p_m(n^{3/2}/8 + n/6) \leq n/8 + n^{1/2}/6$, the probability of having at least $n/6$ ones is $2^{-\Omega(n)}$ by Chernoff's bounds.

Thus, with an error probability of $2^{-\Omega(n)}$, the progress is less than $n/6 = N/3$ implying that for the current population x either $x' \in T$ or $x' = 1^i 0^{N-i}$ with $i \leq N/2 - cN$ holds.

As long as we have not hit the trap, the algorithm spends $n^{3/2}/8$ steps near the trap and the probability to hit the trap in one step is $\Omega(1)$. Thus, the probability to hit the trap in that period of time is $1 - 2^{-\Omega(n^{3/2})}$. Once the trap is hit, the global optimum can only be reached by a mutation flipping at least $N/2$ bits. The probability that this happens in $2^{o(n)}$ steps is exponentially small.

Summing up all error probabilities completes the proof for the (1+1) EA. $\square$

Having proved the properties of the function f_{MA}, the following proofs get easier since we can reuse some of the arguments presented in Theorem 1.

Theorem 2. *While the (1+1) EA with mutation probability $p_m := 1/n$ is efficient on f_{EA}, both RLS and the (1+1) MA with mutation probability $\varepsilon/n \leq p_m \leq 1/2$, $\varepsilon > 0$, fail on f_{EA}.*

Proof. Except fitness values larger than $N - 2$, the function f_{EA} equals the function f_{MA}. We have seen in the proof of Theorem 1 that the (1+1) EA reaches the target T with overwhelming probability. The time until some well-formed search point is reached is $O(n^2)$ with overwhelming probability. Afterwards, the time until the target T is reached is bounded by n with overwhelming probability since there is a constant probability of jumping into T. Thus, the (1+1) EA is efficient on f_{EA}.

By Lemma 2 both RLS and the (1+1) MA reach $0^N 1^N$ with overwhelming probability. In that case, both algorithms deterministically run into the local optimum x with $x' = 1^N$. By Corollary 1 the (1+1) MA can only escape from this local optimum and reach T by mutating x with $x' = 1^N$ into some y with $y' \in T$ or a Hamming neighbor thereof. Moreover, y'' must not contain more than one bit with value zero since otherwise the Hamming distance to all well-formed search points is at least 2 and $0^N 1^N$ is traversed during the local search process. Using these arguments we can show that the probability of the considered mutation is exponentially small: if $p_m \geq n^{1/2}$, the probability that at most one bit flips in x'' is exponentially small. If $p_m < n^{1/2}$, the probability to flip at least $N/2 - 1$ bits in x' is exponentially small. Thus, both RLS and the (1+1) MA fail on f_{EA}. □

Theorem 3. *While RLS is efficient on f_{RLS}, both the (1+1) EA and the (1+1) MA with mutation probability $\varepsilon/n \leq p_m \leq 1/2$, $\varepsilon > 0$, fail on f_{RLS}.*

Proof. By Lemma 2, RLS reaches the well-formed search point $0^N 1^N$ in $O(n^2)$ fitness evaluations with overwhelming probability. On the ridge the fitness is monotonically increasing due to small plateaus of size 2. We now show that the expected time until RLS crosses a non-optimal plateau and thereby increases the fitness is bounded above by $3n$.

Let P_i be a well-formed search point where the left half is $1^i 0^{N-i}$, then P_{2j} and P_{2j+1} form a plateau with fitness $2j$. RLS first reaches P_{2j} when entering the plateau. Then the only accepted search point is P_{2j+1} and the expected time to reach P_{2j+1} is n. Once P_{2j+1} is reached, RLS can reach either P_{2j} or P_{2j+2}. The expected waiting time for a transition is $n/2$ and with probability $1/2$, P_{2j+2} is reached and the fitness increases. Thus, in expectance, P_{2j} has to be left at most 2 times and the expected time to increase the fitness is at most $2(n + n/2) = 3n$.

An application of Markov's inequality yields that the probability to increase the fitness in $6n$ steps is at least $1/2$. To climb up the ridge, the fitness has to be increased $N/2$ times. By Chernoff's bounds the probability of having less than $N/2$ fitness increases in $2N$ independent phases of $6n$ steps is $2^{-\Omega(n)}$. Hence, RLS needs at most $O(n^2) + 6n^2$ steps with overwhelming probability.

Consider the case $\varepsilon/n \leq p_m < n^{-1/2}$. With overwhelming probability both the (1+1) EA and the (1+1) MA reach either T or the ridge at some search point x with $x' = 1^i 0^{N-i}$ and $i \leq cN$ for some $0 < c < 1/12$ (refer to the proof of Theorem 1). On the ridge the (1+1) MA behaves differently for search points with an odd or an even number of one-bits, resp. Consider the subspace induced by the N bits of x'. In case the (1+1) MA reaches some search point $1^{2j} 0^{N-2j}$ on the ridge by local search, this local search process is stopped since there is no Hamming neighbor with larger fitness. Thus, the (1+1) MA performs a mutation in the next generation. In case the

(1+1) MA reaches some search point $1^{2j+1}0^{N-2j-1}$ on the ridge by mutation, local search is called and $1^{2j+2}0^{N-2j-2}$ is reached deterministically.

In the proof of Theorem 1 we have shown that with overwhelming probability the progress on the ridge defined by f_{MA} in $n^{3/2}/8$ mutations is at most $N/3$. By the same arguments, the progress on the ridge defined by f_{RLS} in $n^{3/2}/16$ mutations is at most $N/6$ with overwhelming probability. Moreover, we have seen that the probability to jump into the trap in one mutation is $\Omega(1)$ if $x' = 1^i0^{N-i}$ with $i \leq N/2 - cN$ holds for the current search point and less than cN bits flip in one mutation.

This proves that the (1+1) EA fails on f_{RLS} since the trap is reached with overwhelming probability. For the (1+1) MA each progress by mutation is followed by a progress by local search. However, the progress by a local search call is at most 1. Thus, after $n^{3/2}/16$ mutations the progress of the (1+1) MA by both mutation and local search is at most $N/3$ and the (1+1) MA also fails on f_{RLS}.

In case $n^{-1/2} \leq p_m \leq 1/2$, the (1+1) EA fails to create a well-formed search point (see proof of Theorem 1). The (1+1) MA fails to create a well-formed offspring $y \neq 0^N1^N$ from $x = 0^N1^N$ as the probability to flip at most one bit in x'' is exponentially small and local search stops with 0^N1^N in case more than one bit flips in x''. $\square$

5 On the Local Search Duration

It is a general and important question for memetic algorithms whether local search should be performed to local optimality or not. On some functions it would be unwise to perform local search until a local optimum is found.

An example is the long path problem defined by Horn, Goldberg, and Deb [3] based on a long 2-path. Although in [3] the authors were convinced to observe exponential runtimes of the (1+1) EA on this function, Rudolph [8] proved that the expected runtime of the (1+1) EA is bounded by $O(n^3)$. This is due to the fact that the (1+1) EA can take shortcuts by mutations flipping more than one bit. The (1+1) MA, however, is likely to reach the start of the long 2-path within the initial local search process and then climbs up the whole path of length $(2+1)2^{(n-1)/2} - 2 + 1 = 2^{\Omega(n)}$ (see Lemma 1). Thus, it is essential to stop local search before a local optimum is reached, here.

A simple way of limiting local search is to set a threshold of maximal $d(n)$ steps within one local search call, thus using the following local search procedure instead of Procedure 1.

Procedure 2 (Local Search(y) with maximal duration $d(n)$)**.**

```
 $t := 1$.
 $S := \{z \mid H(z,y) = 1 \wedge f(z) > f(y)\}$.
 While $S \neq \emptyset$ and $t \leq d(n)$ {
     Choose $y \in S$ uniformly at random.
     $t := t + 1$.
     if $t \leq d(n)$ then $S := \{z \mid H(z,y) = 1 \wedge f(z) > f(y)\}$.
 }
 Return $y$.
```

In the following, we investigate the (1+1) MA using Procedure 2. Since the (1+1) MA cannot be efficient if a local search process takes superpolynomial time, we only consider polynomial values for $d(n)$. We will see that even for polynomial values the accurate choice of $d(n)$ can have a large impact on the performance of the (1+1) MA.

We now investigate the following function LPT_m where the duration of the local search process has an impact on the optimization process. The abbreviation LPT stands for long path with trap and the parameter m defines the length of the path.

Definition 4. *Let $n = 4N$ and $N := (K^2+1)$ for some $K \in \mathbb{N}$. Let P_i be the i-th point on the long K-path with dimension N. Then for some $m = 2^{o(n^{1/2})}$ we define the function $\mathrm{LPT}_m \colon \{0,1\}^N \to \mathbb{R} \cup \{-\infty\}$ as*

$$\mathrm{LPT}_m(x') := \begin{cases} i & \text{if } x' = \mathrm{P}_i, i \leq m \\ m - \frac{1}{2} & \text{if } x' \in T_m \\ -\infty & \text{otherwise} \end{cases}$$

where $T_m \subset \{0,1\}^N$ is defined as

$$T_m := \{x' \mid \exists i, N^{1/3} + 2 < i < m - N^{1/3} - 2 \colon H(x', \mathrm{P}_i) = 2,$$
$$\forall j, 0 \leq j \leq m \colon H(x', \mathrm{P}_j) \geq 2\}.$$

Now, if the local search duration is large enough, the (1+1) MA walks past the trap T_m and reaches the final search point P_m on the path. However, if the local search duration is small, the (1+1) MA performs a mutation near the trap, thus having a chance to jump into it. This is shown in the following lemma.

Lemma 3. *Let $n := 4N$, $N := (K^2 + 1)$ for some $K \in \mathbb{N}$, and $m = 2^{o(n^{1/2})}$. Consider the (1+1) MA with mutation probability $\Theta(1/N)$ and maximal local search duration $d(n)$, $d(n) \geq n^{1/2}$, on LPT_m. If the (1+1) MA starts by performing at least $d(n) - 1$ iterations of a local search process from a search point in $\{\mathrm{P}_0, \ldots, \mathrm{P}_{N^{1/3}}\}$, the following statements hold.*

1. *If $d(n) > m$, Prob (reaching T_m before P_m) $= 0$.*
2. *If $d(n) \leq m - n^{1/2} + 1$, Prob (reaching T_m before P_m) $= \Omega(1)$.*

In both cases the number of mutations until either T_m or P_m is reached is bounded by $O(n)$ with overwhelming probability.

Proof. The first statement is trivial since due to construction of LPT_m, local search ends with P_m deterministically.

For the second statement, we observe that the local search process ends with a search point P_i where $i \geq d(n) - 1 > N^{1/3} + 2$ and $i \leq N^{1/3} + d(n) < m - N^{1/3} - 2$ if n large enough. Due to these bounds for i and the definition of T_m, all search points with Hamming distance 2 from P_i either belong to T_m or have Hamming distance at most 1 to a search point in $\{\mathrm{P}_{i-3}, \mathrm{P}_{i-2}, \mathrm{P}_{i-1}, \mathrm{P}_{i+1}, \mathrm{P}_{i+2}, \mathrm{P}_{i+3}\}$. The number of search points where the latter condition holds is $O(N)$. Thus, among all search points with Hamming distance 2 from P_i, an $(1 - O(1/N))$-fraction belongs to T_m. The probability for an arbitrary 2-bit-mutation is $\binom{N}{2} \cdot (\Theta(1/N))^2 \cdot (1 - \Theta(1/N))^{N-2} = \Omega(1)$. Thus, T_m is reached via a direct mutation from P_i with probability $\Omega(1)$.

For the time bound we show that after the first local search process the probability to run into either T_m or P_m in one mutation is $\Omega(1)$. Let $x = \mathrm{P}_i$ be the search point that is mutated. After the first local search process, the probability to jump into T_m is $\Omega(1)$ if $i < m - N^{1/3} - 2$. If $i \geq m - N^{1/3} - 2$ there is a constant probability that mutation creates a replica of x. In that case, local search runs into P_m deterministically. Hence, there is a constant probability to reach either T_m or P_m in one mutation and the probability that this does not happen in n mutations is exponentially small. $\square$

The probability to hit the trap is $\Omega(1)$ if the local search duration is small. This result is weak since we want to obtain overwhelming probabilities. We can achieve better results by using LPT as a subfunction that has to be optimized several times to reach the optimum of the superior function. We will see that the (1+1) MA then reaches the trap in at least one of these trials with overwhelming probability.

Moreover, we want to trap the (1+1) MA if the local search duration is too large, i. e., if it is at least $D+n^{1/2}-1$ for some fixed value D. This can be done as follows. We define subfunctions $\mathrm{LPT}_{D+n^{1/2}-1}$ such that these subfunctions have to be optimized one after another under the starting conditions given in Lemma 3. If the local search duration is too large, these subfunctions are optimized deterministically and in all trials the trap is avoided. However, the global optimum of the superior function is a set of search points where the trap is hit in one of these trials.

This leads to the following definition of a superior function. Since this definition is somehow complicated, a detailed explanation is given below.

Again, we use the construction introduced in Section 3 and divide the bitstring into two halves. The difference to the functions defined in Section 4, however, is that the left half itself is divided into two quarters of length N. Thus, we have search points $x = x'x''x'''$ with $x', x'' \in \{0,1\}^N$ and $x''' \in \{0,1\}^{2N}$ where $x'x''$ represents the left half and x''' represents the right half of x. Note that the term *well-formed* now applies to search points x where $x = x'x''1^{2N}$ for some $x', x'' \in \{0,1\}^N$.

Definition 5. *Let $n := 4N$ and $N := (K^2 + 1)$ for some $K \in \mathbb{N}$. Let $x = x'x''x'''$ with $x', x'' \in \{0,1\}^N$ and $x''' \in \{0,1\}^{2N}$. Let $D \in \mathbb{N}$.*

We define

$$f_D(x) := \begin{cases} \mathrm{ZZO}(x) & \textit{if } x''' \neq 1^{2N} \\ i \cdot 2^{N+1} + h_i(x') & \begin{array}{l} \textit{if } x \notin \mathrm{OPT}, x''' = 1^{2N}, \\ \exists i, 1 \leq i \leq 4n \colon x'' = \mathrm{P}_{i \cdot K}, \\ h_i(x') < \max\{h_i\} \end{array} \\ i \cdot 2^{N+1} + h_i(x') + j & \begin{array}{l} \textit{if } x \notin \mathrm{OPT}, x''' = 1^{2N}, \\ \exists i, 0 \leq i < 4n \colon x'' = \mathrm{P}_{i \cdot K + j}, 0 \leq j \leq K-2, \\ h_i(x') = \max\{h_i\} \end{array} \\ 2^n & \textit{if } x \in \mathrm{OPT} \\ -\infty & \textit{otherwise} \end{cases}$$

where the subfunctions $h_0, \ldots, h_{4n} \colon \{0,1\}^N \to [0, 2^N] \cup \{-\infty\}$ *and the set* $\mathrm{OPT} \subseteq \{0,1\}^n$ *are defined as follows. For* $0 \leq i \leq 4n$*, let* $h_0(x') := 0$*,* $h_i(x') := |x'|_0$ *if* i *odd,* $h_i(x') := \mathrm{LPT}_D(x')$ *if* i *even and* $2 \leq i \leq 2n$*,* $h_i(x') := \mathrm{LPT}_{D+n^{1/2}-1}(x')$ *if* i *even and* $i > 2n$*. Finally,*

$$\begin{aligned} \mathrm{OPT} := \{x \,|\, & x''' = 1^{2N}, \\ & \exists i \in \{2n+2, 2n+4, \ldots, 4n\} \colon x'' = \mathrm{P}_{i \cdot K}, \\ & x' \in T_{D+n^{1/2}-1}\}. \end{aligned}$$

This definition needs some explanation. The only purpose of the right half of the bitstring—the x''' part—is to ensure that the left half is close to 0^{2N} when the first well-formed search point is reached (see Section 3).

Consider a well-formed search point x. In the x' part some subfunction is optimized and the x'' part helps to determine that subfunction. In the x'' part there is a long K-path that is revealed piecewise. If the $(i \cdot K)$-th point on the long K-path is reached in the x'' part, the i-th subfunction is optimized in the x' part. Unless the optimum of that subfunction is reached, all successors with Hamming distance less than K on the long K-path in x'' are hidden, i. e., yield a fitness value of $-\infty$. Thus, if no jumps by at least K bits take place, $x'' = \mathrm{P}_{i \cdot K}$ is maintained until the x' part has been optimized.

If an optimum of the subfunction in x' is found, $K - 2$ successors of $\mathrm{P}_{i \cdot K}$ on the long K-path in x'' are revealed. Now the algorithm can reach $x'' =$

$\mathrm{P}_{(i+1)\cdot K}$ easily by climbing up the path to $x'' = \mathrm{P}_{(i+1)\cdot K-2}$ and jumping to $x'' = \mathrm{P}_{(i+1)\cdot K}$. The gap between $\mathrm{P}_{(i+1)\cdot K-2}$ and $\mathrm{P}_{(i+1)\cdot K}$ ensures that $\mathrm{P}_{(i+1)\cdot K}$ is reached by mutation instead of local search. Hence, a new local search process starts with $x'' = \mathrm{P}_{(i+1)\cdot K}$. Note that in the x' part, an optimum of the subfunction has to be maintained until $x'' = \mathrm{P}_{(i+1)\cdot K}$ is reached. Thus, an optimum of the i-th subfunction is the starting point for the optimization of the $(i+1)$-th subfunction. Once $x'' = \mathrm{P}_{(i+1)\cdot K}$ is reached, the $(i+1)$-th subfunction has to be optimized in the x' part, and so on. That way we can embed arbitrary subfunctions into the x' part (with slight restrictions to the range of fitness values).

The subfunctions with an even index are chosen by the ideas described above. Subfunctions with odd index maximize the number of zeros, thus the optimization of the next subfunction starts with $x' = 0^N$.

Now we are able to prove that the function f_D has the desired properties.

Theorem 4. *Let $D \in \mathbb{N}$. Consider the (1+1) MA with mutation probability $p_m := 1/n$ and maximal local search duration $d(n) \geq n^{1/2}$ with $d(n) = poly(n)$ on f_D. If $d(n) = D$, the (1+1) MA is efficient on f_D. If $d(n) \leq D - n^{1/2}$ or $d(n) \geq D + n^{1/2}$, the (1+1) MA fails on f_D.*

Proof. Again, we investigate typical runs, i. e., runs without errors, and complete the analysis by showing that the sum of all error probabilities is exponentially small.

The probability that at least $N^{1/3}$ bits flip in one mutation is $2^{-\Omega(N^{1/3}\log N)}$. The probability that this happens at least once in $2^{N^{1/3}}$ mutations is still $2^{-\Omega(N^{1/3}\log N)}$ and this is considered an error.

A closer look at the proof of Lemma 2 reveals that the proof also holds for the (1+1) MA with maximal local search duration $d(n) \geq n^{1/2}$ except for the time bound $O(n^2)$. However, since a mutation increases the fitness with probability $\Omega(1/n)$, the number of mutations performed until the first well-formed search point is reached is clearly bounded by $O(n^2)$ with overwhelming probability.

For all well-formed search points $x = x'x''1^{2N}$ with a fitness value larger than $-\infty$, $x'' \in \{\mathrm{P}_0, \mathrm{P}_1, \dots\}$ holds since all other x'' yield $f_D(x) = -\infty$. Hence, if $y = y'y''1^{2N}$ is the first well-formed search point, $y'' = \mathrm{P}_j$ for some

$0 \leq j \leq N^{1/3}$ since all P_j, $j > N^{1/3}$ have a Hamming distance larger than $N^{1/3}$ from 0^N (see Lemma 1).

Having reached y, we partition the rest of the run into $8n$ phases of two different types. Phases $R_1, \ldots, R_{4n}$ concern the right part of the partial bitstring $x'x''$ while phases $L_1, \ldots, L_{4n}$ concern the left part of $x'x''$. Let i be the variable introduced in Definition 5. Phase R_m starts when $i = m - 1$ and lasts as long as the fitness value of the current population is determined by the third case of the definition of f_D. Phase L_m starts right after phase R_m where $i = m$ and lasts as long as the fitness of the current population is determined by the second case of the definition of f_D. It is important that during L_m an operation flipping one bit in x'' or x''' results in a negative fitness value. Thus, a local search process starting with a non-negative fitness value only concerns the x' part.

The co-domains of all subfunctions h_i are subsets of $[0, 2^N] \cup \{-\infty\}$. As long as OPT is not found, all f_D-values encountered in phase L_i are larger than all f_D-values in phases $R_1, L_1, R_2, L_2, \ldots, R_i$ since in the definition of f_D, the value i is weighted by 2^{N+1} and the fitness gain in L_{i-1} and R_i is at most $2^N + K < 2^{N+1}$. Obviously, the fitness values in R_{i+1} are larger than those in L_i. Since at most $N^{1/3}$ bits flip in one step and all search points in $\{\mathrm{P}_{0 \cdot K}, \mathrm{P}_{1 \cdot K}, \ldots\}$ have a pairwise Hamming distance of at least $K > N^{1/3}$, R_i can only be reached by R_{i-1} or L_{i-1}. Thus, the order of phases is $R_1, L_1, R_2, L_2, \ldots, R_{4n}, L_{4n}$ unless a trap or OPT is reached. (However, a phase L_m may be empty if the optimization of h_i starts with an optimum thereof.)

Due to the gap between $\mathrm{P}_{i \cdot K - 2}$ and $\mathrm{P}_{i \cdot K}$, every non-empty phase L_i, $1 \leq i \leq 4n$, can only be reached by a direct mutation to $\mathrm{P}_{i \cdot K}$ or by a mutation to a Hamming neighbor thereof if the first iteration of the local search process reaches $\mathrm{P}_{i \cdot K}$.

Furthermore, L_i for i even starts with a search point x where $x' = \mathrm{P}_j$ with $0 \leq j < N^{1/3}$ (or a Hamming neighbor thereof in case $p_{i \cdot K}$ was reached by a direct mutation). This is due to the fact that 0^N is the unique optimum of $h_{i-1}(x')$ and this optimum must be maintained during phase R_i. Thus, the only accepted step that may modify x' is the mutation leading to the start of L_i. Since $\mathrm{P}_0, \ldots, \mathrm{P}_{N^{1/3}-1}$ are the only search points with an h_i-value larger than $-\infty$ and a Hamming distance less than $N^{1/3}$ to 0^N, the claim follows and the conditions of Lemma 3 are fulfilled for the subspace of the x' part.[1]

In case $d(n) \geq D$ Lemma 3 yields that on all subfunctions LPT_D the search point $x' = \mathrm{P}_D$ is reached with probability 1. Thus, the (1+1) MA does not get trapped within the phases $R_1, L_1, \ldots, R_{2n}, L_{2n}$. Contrarily, if $d(n) \leq D - n^{1/2}$, the probability that $x' \in T_D$ is reached on at least one subfunction LPT_D is $1 - 2^{-\Omega(n)}$. In that case, due to the definition of the trap T_D, at least $N^{1/3}$ bits have to flip in one mutation to reach $x' = \mathrm{P}_D$ or another search point with larger fitness. This proves the failure of the (1+1) MA in case $d(n) \leq D - n^{1/2}$.

Now we consider the phases $R_{2n+1}, L_{2n+2}, \ldots, R_{4n}, L_{4n}$. In case $d(n) \leq D$, by Lemma 3, the probability that $x' \in T_{D+n^{1/2}-1}$ is reached on at least one subfunction $\mathrm{LPT}_{D+n^{1/2}-1}$ is $1-2^{-\Omega(n)}$ and then the set OPT of global optima is found. However, if $d(n) \geq D + n^{1/2}$, Lemma 3 states that $x' = \mathrm{P}_{D+n^{1/2}-1}$ is reached on all subfunctions $\mathrm{LPT}_{D+n^{1/2}-1}$. In that case more than $N^{1/3}$ bits have to flip in one mutation to reach OPT and the (1+1) MA fails.

Finally, we have to prove that in case $d(n) = D$ the runtime of the (1+1) MA is polynomially bounded with overwhelming probability. We first count the number of mutation steps. Every mutation implies one local search call. The number of fitness evaluations in one local search call is trivially bounded by $d(n) \cdot n$, thus the number of fitness evaluations is at most by a factor of $d(n) \cdot n + 1$ larger than the number of mutation steps (plus $d(n) \cdot n$ for the initial local search process).

We have already shown that the first phase R_1 starts within $O(n^2)$ mutations with overwhelming probability. By the same arguments, in a phase R_{i+1} concerning the x'' part either $\mathrm{P}_{(i+1)\cdot K-2}$ or $\mathrm{P}_{(i+1)\cdot K}$ is reached from $\mathrm{P}_{i\cdot K}$ within $O(nK)$ mutation steps with overwhelming probability. To reach $\mathrm{P}_{(i+1)\cdot K}$ from $\mathrm{P}_{(i+1)\cdot K-2}$, a small gap has to be crossed. A sufficient condition to reach $\mathrm{P}_{(i+1)\cdot K}$ from $\mathrm{P}_{(i+1)\cdot K-2}$ is a direct jump to $\mathrm{P}_{(i+1)\cdot K}$ by mutation which happens with probability $\Omega(1/n^2)$. The probability that in all phases $R_1, \ldots, R_{4n}$ less than $4n$ of these jumps occur in cn^3 trials is exponentially small by Chernoff's bounds if $c > 0$ is chosen large enough. Thus, we obtain a term $O(n^3)$ for the number of mutations in all phases $R_1, \ldots, R_{4n}$ that

[1]It is easy to see that Lemma 3 also holds for the optimization of LPT_m as a subfunction of f_D since the $n - N$ additional bits only affect the constant values hidden in the big-O and big-Omega notations.

holds with overwhelming probability.

By Lemma 3 the number of mutations in a phase L_i, i even, is bounded by $O(n)$ with overwhelming probability. For a phase L_i, i odd, it is easy to see that the subfunction $|x'|_0$ is optimized within $O(n^2)$ mutations with overwhelming probability. Hence, we obtain another bound $O(n^3)$ for the overall number of mutations of all phases $L_1, \ldots, L_{4n}$ holding with overwhelming probability.

Together, the number of fitness evaluations is bounded by $O(d(n) \cdot n^4)$ with overwhelming probability. This term is polynomial since $d(n)$ is polynomially bounded. $\square$

6 Conclusions and Future Work

We have defined a simple memetic algorithm, the (1+1) MA, and compared it to the well-known (1+1) EA and randomized local search. For each algorithm we have defined a function where only that algorithm is efficient while the other two algorithms need exponential time with overwhelming probability. Furthermore, the analyses gave insights into the interplay of mutation and local search.

On functions like the well-known long 2-path problems it is essential to limit the duration of local search. Thus, we have investigated the (1+1) MA with a limitation to at most $d(n)$ steps of each local search process. We have defined a class of functions where, given some value D, the (1+1) MA with $d(n) = D$ is efficient. However, even with a small deviation of $d(n)$ by only $n^{1/2}$, the (1+1) MA gets trapped and does not find an optimum in exponential time with overwhelming probability.

This paper is a first step towards a theoretical analysis of memetic algorithms without assumptions. However, many questions remain open. Since local search processes are computationally expensive, it is an interesting question when to apply local search. Furthermore, the analysis of population-based memetic algorithms may reveal insights into the interaction of crossover and local search.

7 Acknowledgement

The author thanks Thomas Jansen for fruitful discussions and comments.

References

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

[2] J. Garnier, L. Kallel, and M. Schoenauer. Rigorous hitting times for binary mutations. *Evolutionary Computation*, 7(2):173–203, 1999.

[3] J. Horn, D. E. Goldberg, and K. Deb. Long path problems. In Y. Davidor, H.-P. Schwefel, and R. Männer, editors, *Parallel Problem Solving from Nature(PPSN III)*, volume 866, pages 149–158, Jerusalem, 9–14 1994. Springer.

[4] T. Jansen and I. Wegener. On the choice of the mutation probability for the (1+1) EA. In *Proceedings of the 6th Conference on Parallel Problem Solving From Nature (PPSN '00)*, volume 1917 of *Lecture Notes in Computer Science*, pages 89–98. Springer, 2000.

[5] T. Jansen and I. Wegener. Real royal road functions – where crossover provably is essential. *Discrete Applied Mathematics*, 149:111–125, 2005.

[6] P. Merz. Advanced fitness landscape analysis and the performance of memetic algorithms. *Evolutionary Computation*, 12(3):303–326, 2004.

[7] P. Moscato. Memetic algorithms: a short introduction. In D. Corne, M. Dorigo, and F. Glover, editors, *New Ideas in Optimization*, pages 219–234. McGraw-Hill, 1999.

[8] G. Rudolph. How mutation and selection solve long-path problems in polynomial expected time. *Evolutionary Computation*, 4(2):195–205, 1997.

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