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 Optimization of Monotone Polynomials by Simple Randomized Search Heuristics

Ingo Wegener and Carsten Witt

No. CI-141/02

Technical Report ISSN 1433-3325 October 2002

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 Optimization of Monotone Polynomials by Simple Randomized Search Heuristics

Ingo Wegener*
FB Informatik, LS 2
Univ. Dortmund
44221 Dortmund, Germany
wegener@ls2.cs.uni-dortmund.de

Carsten Witt*
FB Informatik, LS 2
Univ. Dortmund
44221 Dortmund, Germany
witt@ls2.cs.uni-dortmund.de

October 14, 2002

Abstract

Randomized search heuristics like evolutionary algorithms and simulated annealing find many applications, especially in situations where no full information on the problem instance is available. In order to understand how these heuristics work, it is necessary to analyze their behavior on classes of functions. Such an analysis is performed here for the class of monotone pseudo-boolean polynomials. Results depending on the degree and the number of terms of the polynomial are obtained. The class of monotone polynomials is of special interest since simple functions of this kind can have an image set of exponential size, improvements can increase the Hamming distance to the optimum and in order to find a better search point, it can be necessary to search within a large plateau of search points with the same fitness value.

1 Introduction

Randomized search heuristics like random local search, simulated annealing, and all variants of evolutionary algorithms have many applications and practitioners report surprisingly good results. However, there are only few theoretical papers on the design and analysis of randomized search heuristics. Most of the algorithm papers are problem-oriented, i. e., for a given well-known problem one looks for a better algorithm. In the case of minimizing the worst-case expected runtime

*Supported in part by the Deutsche Forschungsgemeinschaft as a part of the Collaborative Research Center "Computational Intelligence" (SFB 531).

the best algorithms typically are clever-tuned problem-specific algorithms. Sometimes, these algorithms are difficult to implement and, sometimes, they are only asymptotically efficient. This has led to the new area called algorithm engineering or experimental algorithms, where the issue is the design (and analysis) of practically efficient algorithms. Theses heuristics with problem-specific modules can beat the clever algorithms mentioned above.

In this paper, we investigate general randomized search heuristics, namely a random local search algorithm and a mutation-based evolutionary algorithm. It should be obvious that they do not improve heuristics with well-chosen problem-specific modules. Our motivation to investigate these algorithms is that such algorithms are used in some applications and that only an analysis will provide us with some knowledge to understand these algorithms better. This will give us the chance to improve the heuristics, to decide when to apply them, and also to teach them.

General randomized search heuristics are known as "robust" algorithms which can be applied (with at most minor changes) to many different problems. Those practitioners who do not have enough resources (time, money, or experts) to design "problem-specific" modules like such robust algorithms.

In the theory of algorithms the first assumption is that the problem is given and the algorithms can make full use of the properties of the problem instance. However, this assumption is not met in many applications, especially, in engineering disciplines. As an example, we assume that the rough draft of a machine is given and n binary design decisions still are open. Then the set of possible machines can be represented as the search space $\{0,1\}^n$ describing the possible alternatives. There exists a function $f\colon \{0,1\}^n \to \mathbb{R}$ such that $f(a)$ measures the quality of the resulting machine. The main issue is that f may be not known. The problem setting may be so complex that the engineers cannot describe f in a closed form. Perhaps they know some properties of f. However, it is possible to measure $f(a)$ by an experiment (or its simulation). The obvious aim is that a search heuristic quickly finds a good design of a machine. Then randomized search heuristics are the right choice.

However, in such a scenario it seems to be impossible to analyze heuristics. What can be done for an "unknown" problem instance f? No search heuristic will be efficient on classes of functions which are difficult in a complexity-theoretical sense. The hope is that f is somehow easy, which can be formalized as "belonging to a class of functions which is easy." Hence, one should analyze randomized search heuristics on classes of functions which are easy in the scenario where the problem, i. e., the class of functions, and the considered scenario are known.

Each pseudo-boolean function $f\colon \{0,1\}^n \to \mathbb{R}$ can be written uniquely as a polynomial

$$f(x) = \sum_{A \subseteq \{1,\dots,n\}} w_A \cdot \prod_{i \in A} x_i$$

and the degree $d := \max\{|A| \mid w_A \neq 0\}$ and the number N of non-vanishing terms $w_A \neq 0$ are parameters describing properties of f. Note that the value of N can vary if we exchange the meanings of ones and zeros for some variables, i. e., replace some x_i by their negations $1-x_i$. For instance, the product of all $(1-x_i)$ has the maximal number of 2^n non-vanishing terms but only one non-vanishing term if we replace x_i by $y_i := 1-x_i$. The parameter N will be relevant in some upper bounds presented in this paper. However, all search heuristics that we will consider treat zeros and ones in the same way. Therefore, we may silently assume that in the polynomial representation of some monotone polynomial f, variables x_i have possibly been replaced by their negations $1-x_i$ in such a way that N takes its minimum value. Droste, Jansen, and Wegener (2002) have analyzed evolutionary algorithms on polynomials of degree $d = 1$ and Wegener and Witt (2002) have investigated polynomials of degree $d = 2$. The last case is known to be NP-hard in general. A simpler subcase is the case of monotone polynomials where f can be written as a polynomial with non-negative weights on some variable set $z_1, \ldots, z_n$ where $z_i = x_i$ or $z_i = 1-x_i$. In the first case, the function is monotone increasing with respect to x_i and, in the second case, monotone decreasing. In this paper, we investigate randomized search heuristics for the maximization of monotone polynomials of degree bounded by some parameter d. Since all considered heuristics treat zeros and ones in the same way, we can restrict our analysis to monotone increasing polynomials where $z_i = x_i$ for all i. The results hold for all monotone polynomials.

The investigation of polynomials of small degree is well motivated since many problems lead to polynomials of bounded degree. Monotonicity is a restriction which simplifies the problem. However, in the general setting it is unknown whether the function is monotone increasing or decreasing with respect to x_i.

Evolutionary algorithms are general problem solvers which eventually optimize each $f\colon \{0,1\}^n \to \mathbb{R}$. For monotone polynomials we do not see any advantage when using crossover and large populations. Therefore, we investigate a simple standard evolutionary algorithm (EA) which is mutation-based and works with population size 1. This so-called (1+1) EA consists of an initialization step and an infinite loop.

(1+1) EA

Initialization:	Choose $a \in \{0,1\}^n$ randomly.
Loop:	The loop consists of a mutation and a selection step.
Mutation:	For each position i, decide independently whether a_i should be flipped (replaced by $1-a_i$), the flipping probability equals $1/n$.
Selection:	Replace a by a' iff $f(a') \geq f(a)$.

The advantage of the (1+1) EA is that each point can be created from each point with positive probability but steps flipping only a few bits are preferred. Therefore, it is not necessary (as in simulated annealing) to accept worsenings.

Random local search (RLS) flips only one bit per step.

RLS
This works like the (1+1) EA with a different mutation operator.
Mutation: Choose $i \in \{1, \ldots, n\}$ randomly and flip a_i.

RLS cannot escape from local optima where the local neighborhood is the Hamming ball with distance 1. However, it can optimize monotone polynomials (by our assumption the optimum is 1^n) since, for each a, there exists a sequence $a_0 = a, a_1, \ldots, a_m = 1^n$ such that $m \leq n$, $H(a_i, a_{i+1}) = 1$, and $f(a_0) \leq f(a_1) \leq \cdots \leq f(a_m)$. The analysis of RLS will be much easier than the analysis of the (1+1) EA. However, only the (1+1) EA is a general problem solver. The difficulty is that accepted steps can increase the number of zeros and the Hamming distance to the optimum. The problem for all heuristics is that it can be necessary to change many bits (up to d) until one finds a search point with larger f-value (also called fitness).

We have to discuss how we analyze RLS and the (1+1) EA which are defined as infinite loops. In applications, we need a stopping criterion. However, this is not the essential problem. Hence, we are interested in the random optimization time X_f defined as the minimum time step t where an optimal search point is created. Its mean value $E(X_f)$ is called the expected optimization time and $\text{Prob}(X_f \leq t)$ describes the success probability within t steps.

We present monotone polynomials of degree d where the expected optimization time equals $\Theta((n/d) \cdot \log(n/d + 1) \cdot 2^d)$ for RLS and the (1+1) EA and we believe that the upper bound holds for all monotone polynomials. This can be proved for RLS but our best bound for the (1+1) EA is worse and depends on N. For this reason we also investigate a class of algorithms which bridge the difference between RLS and the (1+1) EA. The first idea is to reduce the mutation probability $1/n$ of the (1+1) EA. However, then we increase the probability of useless steps flipping no bit. Hence, we guarantee that at least one bit is flipped. We call the new algorithm RLS_p since it is a modification of RLS.

RLS_p
This works like the (1+1) EA and RLS with a different mutation operator.
Mutation: Choose $i \in \{1, \ldots, n\}$ randomly and flip a_i. For each $j \neq i$, flip a_j independently of the other positions with probability p.

Obviously, RLS_p equals RLS for $p = 0$. For $p = 1/n$, RLS_p is close to the (1+1) EA which omits steps without flipping bit. Hence, we investigate RLS_p only for $0 \leq p \leq 1/n$ and try to maximize p such that we can prove the upper bound $O((n/d) \cdot \log(n/d + 1) \cdot 2^d)$ on the expected optimization time of RLS_p on monotone polynomials.

The paper is structured as follows. Search heuristics with population size 1 lead to a Markov chain on $\{0, 1\}^n$. Therefore, we develop some results on such

Markov chains in Section 2. The results are stated in general form, but some motivation from our applications will be given. The reader may skip this section and may come back to it whenever the results are needed. In Section 3, we investigate the very special case of monomials, i. e., monotone polynomials where $N = 1$. These results are crucial since we later consider how long it takes to maximize a special monomial in the presence of many other monomials in the polynomial. In the Sections 4, 6, and 7, we prove upper bounds on the expected optimization time of the algorithms RLS, RLS_p and the (1+1) EA on monotone polynomials. In Section 5, we present a worst-case monotone polynomial for RLS which is conjectured to be also a worst-case monotone polynomial for RLS_p and the (1+1) EA. We finish with some conclusions. Some of the results of this paper are contained in Wegener (2001). However, that paper contains only proof ideas.

2 Some Results on Markov Chains

The behavior of randomized search heuristics on single monomials is of special interest. For a monomial of degree d, the current state can be identified with the number of ones among the variables of the monomial. This leads to the state space $D = \{0, \ldots, d\}$. In order to obtain an ergodic Markov chain, we replace the selection operator by the selection operator which accepts each a', i. e., a' always replaces a. Then we are interested in the minimal t such that in time step t the state d is reached. This equals the optimization time for the single monomial. Later we will investigate time phases where it is unlikely that there is one step where more than two bits of the considered monomial flip. The transition probabilities for the (1+1) EA under the condition that each step flips at most two bits are denoted by $Q(i, j)$ and the corresponding transition probabilities for RLS_p by $R(i, j)$.

We prove that these Markov chains have the property that it is more likely to reach from i "higher" states than from $i - 1$. This intuitive notion is formalized as follows.

Definition 1 *Let $P(i, j)$ be the transition probabilities of a time-homogeneous Markov chain on $D = \{0, \ldots, d\}$. The Markov chain has an ε-advantage, $\varepsilon \geq 0$, if for all $i \in \{0, \ldots, d-2\}$ the following properties hold.*

1. $P(i, j) \geq (1 + \varepsilon) \cdot P(i + 1, j)$ *for* $j \leq i$,
2. $P(i + 1, j) \geq (1 + \varepsilon) \cdot P(i, j)$ *for* $j > i$.

Lemma 1 *Let $\varepsilon \geq 0$ and $d \leq (n - 1)/(1 + \varepsilon)$. Then the Markov chain with transition probabilities $Q(i, j)$ has an ε-advantage.*

Proof: Since $Q(i,j)=0$ for $|i-j|\geq 3$, only the cases $j=i-1\geq 0$, $j=i$, $j=i+1$, and $j=i+2\leq d$ are not trivial. Let α be the probability that the (1+1) EA flips at most two bits among the d important positions. Let $c:=1+\varepsilon$.

Case 1: $j=i-1\geq 0$. Then

$$\begin{aligned}
Q(i,j)-c\cdot Q(i+1,j) &= Q(i,i-1)-c\cdot Q(i+1,i-1)\\
&= \frac{1}{\alpha}\cdot\binom{i}{1}\cdot\frac{1}{n}\cdot\left(1-\frac{1}{n}\right)^{d-1}-\frac{c}{\alpha}\cdot\binom{i+1}{2}\cdot\frac{1}{n^2}\cdot\left(1-\frac{1}{n}\right)^{d-2}\\
&= \frac{1}{\alpha n}\cdot\left(1-\frac{1}{n}\right)^{d-2}\cdot i\cdot\left(1-\frac{1}{n}-\frac{c\cdot(i+1)}{2n}\right)\\
&\geq \frac{1}{\alpha n}\cdot\left(1-\frac{1}{n}\right)^{d-2}\cdot i\cdot\left(1-\frac{1}{n}-\frac{cd}{n}\right)\ \geq\ 0
\end{aligned}$$

since $i+1\leq d$ and $d\leq (n-1)/c$ by our assumptions.

Case 2: $j=i$. Then $Q(i,i)\geq (1/\alpha)\cdot(1-1/n)^d$ since it is sufficient (but not necessary) not to change any bit among the d important positions in order to stay in state i. Hence, we obtain the following lower bound

$$\begin{aligned}
Q(i,j)-c\cdot Q(i+1,j) &= Q(i,i)-c\cdot Q(i+1,i)\\
&\geq \frac{1}{\alpha}\cdot\left(1-\frac{1}{n}\right)^{d}-c\cdot\frac{1}{\alpha}\cdot\binom{i+1}{1}\cdot\frac{1}{n}\cdot\left(1-\frac{1}{n}\right)^{d-1}\\
&\geq \frac{1}{\alpha}\cdot\left(1-\frac{1}{n}\right)^{d-1}\cdot\left(1-\frac{1}{n}-\frac{cd}{n}\right)\ \geq\ 0.
\end{aligned}$$

Case 3: $j=i+1$. Then

$$\begin{aligned}
Q(i+1,j)-c\cdot Q(i,j) &= Q(i+1,i+1)-c\cdot Q(i,i+1)\\
&\geq \frac{1}{\alpha}\cdot\left(1-\frac{1}{n}\right)^{d}-c\cdot\frac{1}{\alpha}\cdot\binom{d-i}{1}\cdot\frac{1}{n}\cdot\left(1-\frac{1}{n}\right)^{d-1}\\
&\geq \frac{1}{\alpha}\cdot\left(1-\frac{1}{n}\right)^{d-1}\cdot\left(1-\frac{1}{n}-\frac{cd}{n}\right)\ \geq\ 0.
\end{aligned}$$

Case 4: $j=i+2\leq d$. Then

$$\begin{aligned}
Q(i+1,j)-c\cdot Q(i,j) &= Q(i+1,i+2)-c\cdot Q(i,i+2)\\
&= \frac{1}{\alpha n}\cdot\left(1-\frac{1}{n}\right)^{d-2}\cdot\left((d-i-1)\cdot\left(1-\frac{1}{n}\right)-\binom{d-i}{2}\cdot\frac{c}{n}\right)\\
&\geq \frac{1}{\alpha n}\cdot\left(1-\frac{1}{n}\right)^{d-2}\cdot(d-i-1)\cdot\left(1-\frac{1}{n}-\frac{cd}{n}\right)\ \geq\ 0. \qquad \Box
\end{aligned}$$

Lemma 2 *Let $\varepsilon\geq 0$ and $d\leq (n-1)/(3+2\varepsilon)$. Then the Markov chain with transition probabilities $R(i,j)$ has an ε-advantage.*

Proof: Since $R(i,j) = 0$ for $|i-j| \geq 3$, only the cases $j = i-1 \geq 0$, $j = i$, $j = i+1$, and $j = i+2 \leq d$ are not trivial. Let α be the probability that RLS_p flips at most two bits among the d important positions. Let $c := 1+\varepsilon$.

In the following, we derive lower bounds on $R(i,j)$ by inspecting only the case that the first flipping bit of RLS_p is not among the d important positions, which happens with probability $1-d/n$. For upper bounds on $R(i,j)$, we apply the law of total probability with respect to the event that the first flipping bit of RLS_p is among the d important bits or not. The calculations and estimations used in the following cases are very similar to those in the respective cases in the proof of Lemma 1.

Case 1: $j = i-1 \geq 0$. Then

$$\begin{aligned}
R(i,j) - c \cdot R(i+1,j) &= R(i,i-1) - c \cdot R(i+1,i-1) \\
&\geq \frac{1}{\alpha} \cdot \left(\left(1 - \frac{d}{n}\right) \cdot \binom{i}{1} \cdot p \cdot (1-p)^{d-1} - c \cdot \frac{i+1}{n} \cdot \binom{i}{1} \cdot p \cdot (1-p)^{d-2} \right. \\
&\qquad \left. - c \cdot \frac{n-d}{n} \cdot \binom{i+1}{2} \cdot p^2 \cdot (1-p)^{d-2} \right) \\
&\geq \frac{1}{\alpha} \cdot i \cdot p \cdot (1-p)^{d-2} \cdot \left(\left(1 - \frac{d}{n}\right) \cdot (1-p) - c \cdot \frac{d}{n} - c \cdot \frac{n-d}{n} \cdot \frac{d}{2} \cdot p \right) \\
&\geq \frac{1}{\alpha} \cdot i \cdot p \cdot (1-p)^{d-2} \cdot \left(1 - \frac{d}{n} - p - \frac{cd}{n} - c \cdot d \cdot p\right) \\
&\geq \frac{1}{\alpha} \cdot i \cdot p \cdot (1-p)^{d-2} \cdot \left(1 - \frac{d}{n} - \frac{1}{n} - \frac{2cd}{n}\right) \;\geq\; 0.
\end{aligned}$$

The last but one inequality follows since $1/n$ is the maximum value of p we consider. The last one holds since $d \leq (n-1)/(3+2\varepsilon)$ implies $n-d-1-2cd \geq 0$.

Case 2: $j = i$. Then

$$\begin{aligned}
R(i,j) - c \cdot R(i+1,j) &= R(i,i) - c \cdot R(i+1,i) \\
&\geq \frac{1}{\alpha} \cdot \left(\left(1 - \frac{d}{n}\right) \cdot (1-p)^{d} - c \cdot \frac{i+1}{n} \cdot (1-p)^{d-1} \right. \\
&\qquad \left. - c \cdot \frac{n-d}{n} \cdot \binom{i+1}{1} \cdot p \cdot (1-p)^{d-1} \right) \\
&\geq \frac{1}{\alpha} \cdot (1-p)^{d-1} \cdot \left(\left(1 - \frac{d}{n}\right) \cdot (1-p) - c \cdot \frac{d}{n} - c \cdot \frac{n-d}{n} \cdot d \cdot p \right) \\
&\geq \frac{1}{\alpha} \cdot (1-p)^{d-1} \cdot \left(1 - \frac{d}{n} - p - \frac{cd}{n} - c \cdot d \cdot p\right) \\
&\geq \frac{1}{\alpha} \cdot (1-p)^{d-1} \cdot \left(1 - \frac{d}{n} - \frac{1}{n} - \frac{2cd}{n}\right) \;\geq\; 0
\end{aligned}$$

Case 3: $j = i+1$. This case is completely analogous to Case 2 with the only exception that a flipping bit has to be chosen from $d-i$ zeros in the important positions. However, $d-i$ can be estimated above by the same bound d as $i+1$ in Case 2.

Case 4: $j = i+2 \le d$. This case is completely analogous to Case 1 with the only exception that the flipping bit (bits) has (have) to be chosen from $d-i-1$ $(d-i)$ zeros in the important positions. Again, we estimate $d-i$ by d. □

We are interested in the random variable τ^k describing for a time-homogeneous Markov chain Y on D with transition probabilities $P(i,j)$ the first point of time when it reaches state d if it starts in state k. If Y has a 0-advantage, it should be advantageous to start in a "higher state." This is made precise in the following lemma.

Lemma 3 *Let $P(i,j)$ be the transition probabilities of a time-homogeneous Markov chain with 0-advantage on $D = \{0,\dots,d\}$. Then*

$$\mathrm{Prob}(\tau^i \ge t) \ge \mathrm{Prob}(\tau^{i+1} \ge t)$$

for $0 \le i \le d-1$ and each t. Moreover, $E(\tau^i) \ge E(\tau^{i+1})$.

Proof: The second claim follows from the first one. The first claim is proved by induction on t where the base step $t=0$ is obvious. We can assume $i \le d-1$ since $\mathrm{Prob}(\tau^d = 0) = 1$. By the law of total probability

$$\mathrm{Prob}(\tau^i \ge t+1) - \mathrm{Prob}(\tau^{i+1} \ge t+1) = \sum_{k=0}^{d-1} (P(i,k) - P(i+1,k)) \cdot \mathrm{Prob}(\tau^k \ge t).$$

By induction hypothesis and the assumption of a 0-advantage,

$$P(i,k) - P(i+1,k) \ge 0 \ \text{ and } \ \mathrm{Prob}(\tau^k \ge t) \ge \mathrm{Prob}(\tau^i \ge t) \ \text{ if } k \le i,$$

and

$$P(i,k) - P(i+1,k) \le 0 \ \text{ and } \ \mathrm{Prob}(\tau^k \ge t) \le \mathrm{Prob}(\tau^i \ge t) \ \text{ if } k \ge i+1.$$

Hence,

$$\begin{aligned}
&\mathrm{Prob}(\tau^i \ge t+1) - \mathrm{Prob}(\tau^{i+1} \ge t+1) \\
&\qquad \ge \mathrm{Prob}(\tau^i \ge t) \cdot \sum_{k=0}^{d-1} (P(i,k) - P(i+1,k)) \\
&\qquad = \mathrm{Prob}(\tau^i \ge t) \cdot (1 - P(i,d) - (1 - P(i+1,d))) \ \ge \ 0,
\end{aligned}$$

where $P(i+1,d) - P(i,d) \ge 0$ again follows from the assumption of a 0-advantage of the Markov chain. □

Later, we compare different Markov chains. The complicated Markov chain Y_1 describing a randomized search heuristic on a monotone polynomial with many terms is compared with the simple Markov chain Y_0 describing a randomized search heuristic on a single monomial. The idea is to use results for Y_0 to obtain results for Y_1. We denote by τ_0^i and τ_1^i the random time to reach state d from state i according to Y_0 and Y_1, respectively.

Definition 2 *Let $P_0(i,j)$ and $P_1(i,j)$ be the transition probabilities of the time-homogeneous Markov chains Y_0 and Y_1 on $D = \{0, \ldots, d\}$. The Markov chain Y_1 has an advantage compared to Y_0 if $P_1(i,j) \geq P_0(i,j)$ for $j \geq i+1$ and $P_1(i,j) \leq P_0(i,j)$ for $j \leq i-1$.*

Lemma 4 *If Y_1 has an advantage compared to Y_0 and Y_0 has a 0-advantage, then $\mathrm{Prob}(\tau_1^i \geq t) \leq \mathrm{Prob}(\tau_0^i \geq t)$ and $E(\tau_1^i) \leq E(\tau_0^i)$.*

Proof: Again it is sufficient to prove the first claim by induction on t and the case $t = 0$ is trivial. The result holds if $E(\tau_0^i) = \infty$. By the law of total probability and the induction hypothesis we obtain for $i < d$

$$\begin{aligned}
&\mathrm{Prob}(\tau_0^i \geq t+1) - \mathrm{Prob}(\tau_1^i \geq t+1) \\
&\qquad = \sum_{k=0}^{d-1} (P_0(i,k) \cdot \mathrm{Prob}(\tau_0^k \geq t) - P_1(i,k) \cdot \mathrm{Prob}(\tau_1^k \geq t)) \\
&\qquad \geq \sum_{k=0}^{d-1} (P_0(i,k) - P_1(i,k)) \cdot \mathrm{Prob}(\tau_0^k \geq t).
\end{aligned}$$

Now we can continue as in the proof of Lemma 3 where $P_0(i,k)$ plays the role of $P(i,k)$ and $P_1(i,k)$ plays the role of $P(i+1,k)$. The assumption that Y_1 has an advantage compared to Y_0 can be applied analogously to the assumption that the Markov chain considered in Lemma 3 has a 0-advantage. Moreover, application of the induction hypothesis in the proof of Lemma 3 is replaced by the analogous inequalities $\mathrm{Prob}(\tau_0^k \geq t) \geq \mathrm{Prob}(\tau_0^i \geq t)$ for $k \leq i$, and $\mathrm{Prob}(\tau_0^k \geq t) \leq \mathrm{Prob}(\tau_0^i \geq t)$ for $k \geq i+1$. They follow by Lemma 3 since Y_0 is assumed to have a 0-advantage. $\square$

Finally, we apply Lemma 4 to compare two Markov chains Y_0 and Y_1 where weaker conditions hold than in Lemma 4. We compare Y_0 and Y_1 by parameters $c(i,j)$ such that $P_1(i,j) = c(i,j) \cdot P_0(i,j)$. This includes an arbitrary choice of $c(i,j)$ if $P_0(i,j) = P_1(i,j) = 0$.

Definition 3 *Let $P_0(i,j)$ and $P_1(i,j)$ be the transition probabilities of Y_0 and Y_1 such that $P_1(i,j) = c(i,j) \cdot P_0(i,j)$ for some $c(i,j)$. Then Y_1 has a relative advantage compared to Y_0 if $c(i,j) \geq c(i,i+1)$ for $j \geq i+1$, $c(i,j) \leq c(i,i+1)$ for $j \leq i-1$, and $0 < c(i,i+1) \leq 1$ for all $i \leq d-1$.*

Lemma 5 *If Y_1 has a relative advantage compared to Y_0 and Y_0 has a $(c_{\min}^{-1} - 1)$-advantage, then $E(\tau_1^i) \le c_{\min}^{-1} \cdot E(\tau_0^i)$ for $c_{\min} := \min\{c(i, i+1) \mid 0 \le i \le d-1\}$.*

Proof: The proof idea is to define a third Markov chain Y_2 with transition probabilities $P_2(i, j)$ such that

- $E(\tau_2^i) \le c_{\min}^{-1} \cdot E(\tau_0^i)$,
- Y_1 has an advantage compared to Y_2,
- Y_2 has a 0-advantage.

The second and third claim imply by Lemma 4 that $E(\tau_1^i) \le E(\tau_2^i)$ and then the first claim implies the lemma.

Let $P_2(i, j) := c(i, i+1) \cdot P_0(i, j)$ for $i \le d-1$ and $j \ne i$ and $P_2(i, i) := 1 - \sum_{j \ne i} P_2(i, j)$. Moreover, $P_2(d, j) := P_0(d, j)$. Since $c(i, i+1) \le 1$, these are transition probabilities of a Markov chain Y_2 which can be interpreted as follows. As long as we have not reached state d, we stay in the present state i with probability $1 - c(i, i+1)$ and simulate Y_0 for one step otherwise (which may also imply that we stay in state i). The expected time before a step of Y_0 is simulated equals $c(i, i+1)^{-1} \le c_{\min}^{-1}$ in state i. This implies the first claim.

Next we prove the second claim. First, let $j \ge i+1$. Then

$$P_1(i, j) - P_2(i, j) \;=\; c(i, j) \cdot P_0(i, j) - c(i, i+1) \cdot P_0(i, j) \;\ge\; 0$$

since $c(i, j) \ge c(i, i+1)$ for $j \ge i+1$. Similarly, $c(i, j) \le c(i, i+1)$ for $j \le i-1$ and $P_1(i, j) - P_2(i, j) \le 0$ by an analogous calculation.

We still have to prove the third claim. Observe that our assumptions imply $P_2(i+1, j) \ge c(i+1, i+2) \cdot P_0(i+1, j)$ even for $j = i+1$. Since Y_0 has a $(c_{\min}^{-1} - 1)$-advantage and $c(i, i+1) \le 1$, we obtain for $j \ge i+1$

$$\begin{aligned} P_2(i+1, j) - P_2(i, j) \;&\ge\; c(i+1, i+2) \cdot P_0(i+1, j) - c(i, i+1) \cdot P_0(i, j) \\ &\ge\; c_{\min} \cdot P_0(i+1, j) - P_0(i, j) \;\ge\; 0. \end{aligned}$$

An analogous calculation shows $P_2(i+1, j) - P_2(i, j) \le 0$ for the case $j \le i$. □

The last result in this technical section is a generalization of Wald's identity (see Feller (1971)). We do not claim to be the first to prove this result, but we have not found it in the literature.

Lemma 6 *Let D_i, $i \in \mathbb{N}$, be a sequence of random variables such that $|D_i| \le c$ for a constant c. For $s > 0$, let τ_s be the minimal i where $D_1 + \cdots + D_i = s$. If $E(\tau_s) < \infty$ and $E(D_i \mid \tau_s \ge i)$ is bounded below by a positive constant ℓ for all i where* $\mathrm{Prob}(\tau_s \ge i) > 0$, *then $E(\tau_s) \le s/\ell$.*

Proof: Since $|D_i| \le c$, $E(D_i \mid \tau_s \ge i)$ exists for all i where $\mathrm{Prob}(\tau_s \ge i) > 0$. By definition of τ_s,

$$\begin{aligned} s &= E\left(\sum_{i=1}^{\tau_s} D_i\right) \\ &= \sum_{j=1}^{\infty} \mathrm{Prob}(\tau_s = j) \cdot E(D_1 + \cdots + D_j \mid \tau_s = j) \\ &= \sum_{j=1}^{\infty} \sum_{i=1}^{j} \mathrm{Prob}(\tau_s = j) \cdot E(D_i \mid \tau_s = j) \\ &= \sum_{i=1}^{\infty} \sum_{j=i}^{\infty} \mathrm{Prob}(\tau_s = j) \cdot E(D_i \mid \tau_s = j). \end{aligned}$$

The last equality holds since $|D_i| \le c$ implies that the series is absolutely convergent. Since $j \ge i$, $\tau_s = j$ implies $\tau_s \ge i$ and we obtain that

$$\begin{aligned} s &= \sum_{i=1}^{\infty} \sum_{j=i}^{\infty} \mathrm{Prob}(\tau_s = j \mid \tau_s \ge i) \cdot \mathrm{Prob}(\tau_s \ge i) \cdot E(D_i \mid \tau_s = j) \\ &= \sum_{i=1}^{\infty} \mathrm{Prob}(\tau_s \ge i) \cdot \left[\sum_{j=i}^{\infty} \mathrm{Prob}(\tau_s = j \mid \tau_s \ge i) \cdot E(D_i \mid \tau_s = j \,\wedge\, \tau_s \ge i)\right]. \end{aligned}$$

If we can bound the $[\cdot]$-term below by ℓ, we obtain $s \ge E(\tau_s) \cdot \ell$ and, therefore, the lemma. In the $[\cdot]$-term we may add the terms for $j \in \{1 \ldots, i-1\}$. They are all equal to 0 since in this case $\mathrm{Prob}(\tau_s = j \mid \tau_s \ge i) = 0$. Hence, the $[\cdot]$-term equals

$$\sum_{j=1}^{\infty} \mathrm{Prob}(\tau_s = j \mid \tau_s \ge i) \cdot E(D_i \mid \tau_s = j \,\wedge\, \tau_s \ge i).$$

This is by definition equal to $E(D_i \mid \tau_s \ge i)$, which by assumption is bounded below by ℓ. $\square$

3 The Optimization of Monomials

Because of the symmetry of all considered algorithms with respect to the bit positions and the role of zeros and ones, we can investigate w. l. o. g. the monomial $m(x) = x_1 \cdots x_d$. As long as no optimal point is found, each new search point a' replaces the old search point a. This leads to quite simple Markov chains which can be analyzed by standard methods. We analyze the search heuristics as infinite processes where a' always replaces a (even if a is optimal and a' is not) and are interested in the first time of hitting an optimal point. These Markov chains are ergodic and the only stationary distribution is the uniform distribution on $\{0,1\}^n$. We compress the state space to $D = \{0,\dots,d\}$ and search point a is replaced by $a_1 + \cdots + a_d$. The above result implies that the stationary distribution π on D has the values $\pi(i) = \binom{d}{i}/2^d$. Let Y be the Markov chain on D resulting from one of the considered algorithms and let $\tau_{i,j}$ be the minimal point of time $t \geq 1$ such that $Y(t) = j$ when starting in i. The fundamental theorem on ergodic Markov chains (see Motwani and Raghavan (1995)) states that $E(\tau_{i,i}) = 1/\pi(i) = 2^d/\binom{d}{i}$. We are interested in $E(\tau_{i,d})$. Since all considered algorithms often only flip a single bit, we estimate $E(\tau_{i,d})$ for all i by the sum of all $E(\tau_{j,j+1})$, $0 \leq j \leq d-1$.

Let $P(i,j)$ be the transition probabilities of Y. Then by the law of total probability

$$E(\tau_{j+1,j+1}) = 1 + \sum_{\substack{k=0\\k\neq j+1}}^{d} P(j+1,k)\cdot E(\tau_{k,j+1}) \geq P(j+1,j)\cdot E(\tau_{j,j+1})$$

or

$$E(\tau_{j,j+1}) \leq \frac{2^d}{\binom{d}{j+1}\cdot P(j+1,j)}.$$

Altogether,

$$E(\tau_{i,d}) \leq 2^d \cdot \sum_{j=0}^{d-1} \frac{1}{\binom{d}{j+1}\cdot P(j+1,j)}.$$

Hence, we need lower bounds on $P(j+1,j)$. This parameter depends on the considered algorithm. For RLS_p and $p \leq 1/n$, we switch from $j+1$ to j if the first flipping bit is one of the $j+1$ ones among the first d positions and no other bit flips. This leads to the lower bound $\frac{j+1}{n}(1-p)^{d-1} \geq \frac{j+1}{n}(1-1/n)^{d-1} \geq \frac{j+1}{en}$. For the (1+1) EA the probability that exactly one bit flips and this is one of the $j+1$ ones among the first d positions equals $(j+1)\frac{1}{n}(1-1/n)^{n-1} \geq \frac{j+1}{en}$. These bounds also hold if we consider the algorithms under the conditions that we omit steps with more than two flipping bits. Now

$$\binom{d}{j+1}\cdot\frac{j+1}{en} = \binom{d-1}{j}\cdot\frac{d}{en}$$

and, obviously, the sum of all $1/\binom{d-1}{j}$, $0 \leq j \leq d-1$, is bounded above by a constant c^*. Altogether,

$$E(\tau_{i,d}) \leq e\cdot c^*\cdot n\cdot 2^d/d = O((n/d)\cdot 2^d).$$

Theorem 1 *The algorithms RLS_p, $p \leq 1/n$, and (1+1) EA in their pure form and under the condition of omitting all steps flipping more than two bits optimize monomials of degree d in an expected time of $\Theta((n/d)\cdot 2^d)$. The upper bounds also hold if the initialization is replaced by the deterministic choice of any $a \in \{0,1\}^n$.*

Proof: The upper bound has been proven by the preceding arguments. The lower bound is easy to prove. Droste, Jansen, Tinnefeld, and Wegener (2002) have shown a lower bound of $2^{d-1} + 1/2$ for each black-box algorithm optimizing monomials of degree d. This bound even holds if each search point is counted only once. We do not define the notion black-box algorithm here. We only state the fact that this bound holds for all considered algorithms which indeed are black-box algorithms. We apply this bound for the compressed search space $\{0,1\}^d$ consisting of the first d bits. Hence, it is sufficient to prove that for each compressed search point it takes an expected time of $\Omega(n/d)$ before another compressed search point is found. The probability that the (1+1) EA flips at least one of the first d bits is bounded above by d/n. The corresponding probability for RLS_p is bounded above by $d/n + ((n-d)/n) \cdot d \cdot p \leq 2d/n$. The probabilities are even smaller if steps flipping more than two bits are omitted. This proves the lower bound. $\square$

Theorem 1 has been proved before by Garnier, Kallel, and Schoenauer (1999) for the special case $d = n$ and the algorithms RLS_0 and (1+1) EA.

4 On the Analysis of Random Local Search

The random local search algorithm RLS flips one bit per step. If the considered polynomial f contains a monomial m on the variable set $m(X)$ and if a activates m, i. e., all bits of m equal 1, then m will never be deactivated. This property simplifies the analysis of RLS and we will obtain the best possible upper bound $O((n/d) \cdot \log(n/d+1) \cdot 2^d)$ on the expected optimization time of RLS on monotone polynomials. We prove the upper bound before presenting in Section 5 an example that the bound is best possible.

It is crucial for our analysis how the monomials overlap. Intuitively, many overlapping monomials simplify the optimization. Consider two monomials m_1 and m_2 of degree d which overlap in i variables. If one of the monomials is activated, the other monomial has i variables which are fixed on their right value and it is sufficient to "optimize the truncated monomial of degree $d-i$." Nevertheless, we start with a result where we only investigate a set of non-overlapping monomials.

Lemma 7 *Let f be a monotone polynomial whose degree is bounded by d and let M be a set of pairwise non-overlapping monomials of f. The expected time until RLS has activated all monomials in M is bounded by $O((n/d) \cdot \log(n/d+1) \cdot 2^d)$.*

Proof: If a monomial outside M is activated, some variables are fixed at the value 1. This can only speed up the activation of the monomials in M. Hence, it is sufficient to consider the case where f contains only the monomials in M.

Let c^* be the constant from Section 3 such that each monomial of degree i is activated by RLS in an expected time bounded by $c^* \cdot (n/i) \cdot 2^i$. Let c^{**} be another appropriate constant which is chosen below depending on c^*. For $s := 2c^{**} \cdot (n/d) \cdot \lceil \log(n/d) + 1 \rceil \cdot 2^d$ we claim that the probability that RLS activates all monomials in M within s steps is at least 1/2. This will prove the lemma.

Since $(n/i) \cdot (\log(n/i) + 1) \cdot 2^i$ is monotone increasing with respect to i if $i \geq 2$, we can choose the constant c^{**} such that the considered time interval of s steps contains $\lceil \log(n/i) \rceil + 1$ phases of length $2c^* \cdot (n/i) \cdot 2^i$ each. Hence, the probability of not activating a monomial of degree d within s steps is bounded by $(1/2)^{\lceil \log(n/i) \rceil + 1} \leq i/(2n)$. Let c_i be the number of monomials in M of degree i. Then we can conclude that the probability of not activating at least one of the monomials in M within s steps is bounded above by the sum of all $c_i \cdot i/(2n)$. Since the monomials in M are pairwise non-overlapping, the sum of all $c_i \cdot i$ is bounded above by n and the failure probability within s steps is bounded above by 1/2. This proves the claim and, therefore, the lemma. □

Theorem 2 *The expected optimization time of RLS on a monotone polynomial of degree* d *is bounded by* $O((n/d) \cdot \log(n/d + 1) \cdot 2^d)$.

Proof: The idea is to apply Lemma 7 repeatedly for carefully chosen sets M_i of monomials. Let $f_0 = f$ and M_0 be a maximal set of pairwise non-overlapping monomials of f_0. After having chosen $M_0, \ldots, M_{i-1}$ we consider the truncation f_i of f defined as the subfunction of f obtained by assigning the value 1 to all variables contained in one of the monomials in $M_0 \cup \cdots \cup M_{i-1}$. If f_i is the constant 1, the procedure is finished. Otherwise, M_i is a maximal set of pairwise non-overlapping monomials of f_i. Obviously, $f_j = 1$ for some $j \leq d$.

The expected optimization time of RLS on f is bounded above by the sum of all T_i where T_i is the expected time until all monomials of M_i are activated. Since we always choose a maximal set of pairwise non-overlapping monomials, in each iteration at least one variable of each monomial is replaced by the constant 1. Hence, the degree of all monomials in M_i is bounded above by $d - i$ and, by Lemma 7, $T_i \leq c \cdot \frac{n}{d-i} \cdot \log(\frac{n}{d-i} + 1) \cdot 2^{d-i}$ for some constant c. Finally, by simple calculations, $T_0 + \cdots + T_{d-1}$ is bounded by $O((n/d) \cdot \log(n/d + 1) \cdot 2^d)$ and this proves the theorem. □

5 A Worst-Case Example

In order to prove that the bound of Theorem 2 is asymptotically tight, we look for a monotone degree-d polynomial which is difficult for RLS (and other randomized search heuristics). The proof of the upper bound shows that the worst case seems to be the case of many non-overlapping monomials. W. l. o. g. we only consider the case $n = kd$ for some integer k. The set of variables can be partitioned into k blocks of size d each. We investigate the function which counts the number of blocks where all bits have the value 1. This function can be written as

$$\mathrm{RR}_d(x) = \sum_{i=0}^{k-1} x_{id+1} \cdots x_{id+d}$$

and, therefore, is a polynomial with k non-overlapping monomials of degree d. This function is well known in the community of genetic algorithms. It got the name *royal road function* since Mitchell, Forrest, and Holland (1992) conjectured that these functions are royal roads to prove the usefulness of the crossover operator. However, as Mitchell, Holland, and Forrest (1994) have observed, the royal road functions do not gain from crossover. Functions where it can be proved that crossover reduces the expected optimization time from exponential (for all mutation-based evolutionary algorithms) to polynomial have been found only recently (Jansen and Wegener (2001, 2002)).

The royal road functions are indeed the most difficult monotone polynomials of degree d for RLS as will be shown in the following theorem.

Theorem 3 *The probability that RLS has optimized the degree-d royal road function within* $(n/d) \cdot \log(n/d) \cdot 2^d/32$ *steps is bounded above by* $o(1)$ *if* $d = o(n)$. *The expected optimization time of RLS on royal road functions equals* $\Theta((n/d) \cdot \log(n/d + 1) \cdot 2^d)$.

Proof: The upper bound on the expected optimization time is contained in Theorem 2.

First, we consider the simple cases, namely $d = \Theta(n)$ and $d = O(1)$. For $d = \Theta(n)$, the lower bound equals $\Omega(2^d)$ and follows from the results of Garnier, Kallel, and Schoenauer (1999) as well as more generally from Droste, Jansen, Tinnefeld, and Wegener (2002). For $d = O(1)$, we lower bound the optimization time by the time until each bit initialized as 0 has flipped at least once. Then the bound follows from the Coupon Collector's Theorem (see Motwani and Raghavan (1995)).

In the following we assume that $d = \omega(1)$ and $d = o(n)$. We have to investigate the first $k \cdot (\log k) \cdot 2^d/32$ steps. We are interested in the probability that each monomial has been activated (all bits are set to 1) within this period. RLS never deactivates a monomial since it flips one bit per step and does not accept worsenings.

Considering a single monomial, the substring containing the corresponding bits is initialized randomly. The probability that the substring is affected by a step of RLS equals $d/n = 1/k$. These steps are called *essential* with respect to the considered monomial. The monomials are "close to independent," more precisely the number of steps essential for one monomial depends on the number of essential steps for the other monomials. However, for d not too small, the monomials have almost the same number of essential steps. If the number of essential steps for a monomial is fixed, then the event whether it is activated is independent of the steps concerning the other monomials. Garnier, Kallel, and Schoenauer (1999) have shown that for $d \to \infty$ the probability that a monomial is activated within $t2^d$ essential steps is approximately $1 - e^{-t}$. For d large enough, the success probability within 2^{d-1} essential steps is therefore bounded by $1/2$. Besides, it has been shown that the random number of steps τ until activating m has for $d \to \infty$ a memoryless distribution, i. e., $\mathrm{Prob}(\tau \geq t) = \mathrm{Prob}(\tau \geq t+t' \mid \tau \geq t')$ for all $t, t' \geq 0$. More precisely, the difference $\mathrm{Prob}(\tau \geq t) - \mathrm{Prob}(\tau \geq t + t' \mid \tau \geq t')$ is bounded by $O(1/d)$ independently of t and t'. If d is large enough, this implies that even the conditional success probability $\mathrm{Prob}(\tau \leq 2^{d-1} + t' \mid \tau \geq t')$ is bounded above by $1/2$ for all values of t'. These results will be applied later on.

Since $d \geq 3$, we can apply Chernoff bounds to prove that the probability of starting with less than $k/2$ passive monomials is exponentially small with respect to k and therefore $o(1)$. In the following, we concentrate our considerations to $k/2$ monomials which are assumed to start as passive monomials. The $k \cdot (\log k) \cdot 2^d/32$ steps are partitioned to $(\log k)/4$ phases of length $k \cdot 2^d/8$ each.

Let p_i be the random number of passive monomials after the i-th phase. By our assumptions, $p_0 \geq k/2$. We claim that the event that $p_i < p_{i-1}/8$ for some $i \leq (\log k)/4$ is exponentially small with respect to $k^{1/4}$. If this event does not occur, the number of passive monomials at the end of all $(\log k)/4$ phases is at least

$$p_0 \cdot \left(\frac{1}{8}\right)^{(\log k)/4} \geq \frac{1}{2} \cdot k \cdot k^{-3/4} \geq \frac{k^{1/4}}{2}.$$

For proving the claim we can assume by the last calculation that we have at least $p_{i-1} \geq k^{1/4}/2$ passive monomials at the end of the $(i-1)$-th phase. The expected number of steps in one phase which are essential for one of the passive monomials equals $p_{i-1} \cdot 2^d/8$. By Chernoff bounds, the probability of more than $p_{i-1} \cdot 2^d/4$ of these steps is exponentially small with respect to p_{i-1} and, therefore, with respect to $k^{1/4}$. Hence, the probability that this happens in at least one of the $(\log k)/4$ phases is still exponentially small with respect to $k^{1/4}$. We assume that such an event does not occur. In each phase we have at most $p_{i-1} \cdot 2^d/4$ essential steps for all p_{i-1} passive monomials. By the pigeon-hole principle there are at most $p_{i-1}/2$ passive monomials with at least $2^d/2$ essential steps each. Pessimistically, we assume that these monomials are activated in the i-th phase. For each other monomial the probability of getting activated is at most $1/2$. Hence, by Chernoff bounds, the probability of activating more than $3/4$ of these monomials is exponentially small with respect to $p_{i-1} \geq k^{1/4}/2$. Altogether, the failure probability in one phase is exponentially small with respect to $k^{1/4}$ and this also holds for all phases together. □

Essentially, the same bounds hold for RLS_p, $p \leq 1/n$, and the (1+1) EA when optimizing royal road functions. The upper bound on the expected optimization time of the (1+1) EA has been proved by Mitchell, Holland, and Forrest (1994). The same methods work for RLS_p.

Theorem 4 *For each $\varepsilon > 0$, the probability that the (1+1) EA is on a royal road function by a factor of $(1+\varepsilon)$ faster than RLS is $O(1/n)$. The same holds for RLS_p, $p \leq 1/n$, and the factor $(2+\varepsilon)$ in comparison to RLS.*

Proof: We prove the result on the (1+1) EA by replacing the (1+1) EA by a faster algorithm $(1+1)^*$ EA and comparing the faster algorithm with RLS. A step of the $(1+1)^*$ EA works as follows. First, the number k of flipping bits is chosen according to the same distribution as for the (1+1) EA. Then the $(1+1)^*$ EA flips a random subset of k bits. This can be realized as follows. In each step, one random bit is flipped until one obtains a point of Hamming distance k to the given one. Now the new search point of the $(1+1)^*$ EA is obtained as follows. The selection procedure of RLS is applied after each step. This implies by the properties of the royal road functions that we obtain a search point a^* compared to the search point a of the (1+1) EA such that $a \leq a^*$ according to the componentwise partial order. This implies that the $(1+1)^*$ EA reaches the optimal string 1^n not later than the (1+1) EA.

However, the $(1+1)^*$ EA chooses flipping bits as RLS and it uses the same selection procedure. The difference is that sometimes many steps of RLS are required to simulate a single step of the $(1+1)^*$ EA. The probability that the (1+1) EA tries to flip more than δn bits, $\delta > 0$ a constant, in one step is $2^{-\Omega(n \log n)}$ and, therefore, still $2^{-\Omega(n \log n)}$ if we consider the first $2^{O(n \log^{1/2} n)}$ steps. If this event happens, this is included in the failure probability. Moreover, the probability of an optimization time which is at least $2^{cn \log^{1/2} n}$, $c > 0$ a constant, is by Markov bounds and the upper bound $O(2^n)$ for all monotone polynomials exponentially small. Otherwise, we simulate steps with $k \leq \delta n$ flipping bits. Let a be the starting point of the simulation of one step. Hence, all considered search points have a Hamming distance of at most δn from a. The probability of increasing the Hamming distance with the next flipping bit is at least $1-\delta$. Hence, among t steps we expect $(1-\delta)t$ distance-increasing steps and δt distance-decreasing steps leading to an overhead of $(1-2\delta)t$ distance-increasing steps. Let t be the random number of steps of RLS. If $t = \Omega(n)$ (this is the lower bound for all royal road functions, which holds with probability exponentially close to 1), we have an exponentially small failure probability if we want to ensure an overhead of $(1-3\delta)t$ distance-increasing steps. The expected number of flipping bits of the (1+1) EA in one step equals 1 and the variance is a little less than 1. Now we can apply the inequality of Chebyshev to obtain a bound of $O(1/t)$ on the probability that there are more than $(1+\gamma)t$ flipping bits in t steps for some fixed constant $\gamma > 0$. If all the failures do not happen, we have to simulate at most $(1+\gamma)t$ flipping bits and $(1+\gamma) \cdot t/(1-3\delta)$ steps of RLS are sufficient to simulate them. Choosing γ and δ such that $(1+\gamma)/(1-3\delta) = 1/(1+\varepsilon)$, we have proved the claim.

The statement on RLS_p follows in the same way taking into account that the expected number of flipping bits per step is upper bounded by $2 - 1/n$ and the variance of this random number is less than 1. □

6 On the Analysis of RLS_p

The analysis of an algorithm which often flips three or more bits is much more complicated. Active monomials can be deactivated since passive monomials with a larger total weight are activated. Moreover, even the analysis until a monomial is activated is more difficult. Steps where two bits of the monomial flip from 0 to 1 and only one bit flips from 1 to 0 may not be accepted since the bit flipping to 0 causes a decrease of the fitness. Hence, we do not obtain simple Markov chains like in the case of RLS or in the case of single monomials.

However, the probability of three and more flipping bits contained in the same monomial strongly depends on the degree of the monomial. Theorems 3 and 4 have shown that we can obtain polynomial upper bounds only if $d = O(\log n)$. Therefore, it make sense to concentrate on this case. If the current search point is not optimal, there is a passive monomial which has to be activated. The best bound we can hope for is of order $(n/d) \cdot 2^d$. We can prove this bound if p is small enough.

Lemma 8 *Let f be a monotone polynomial of degree $d \le c \log n$ and let m be one if its monomials. There is a constant $\alpha > 0$ such that RLS_p with $p = \min\{1/n, \alpha/(n^{c/2} \log n)\}$ activates m in an expected time of $O((n/d) \cdot 2^d)$ steps.*

Proof: The idea is to prove that RLS_p activates m with a constant probability $\varepsilon > 0$ within a phase of $c' \cdot (n/d) \cdot 2^d$ steps for some constant c'. Since our analysis does not depend on the starting point, this implies an upper bound $c' \cdot (n/d) \cdot 2^d/\varepsilon$ on the expected time to activate m.

We bound the probability of the following so-called failure events:

- there is a step in the phase where at least three bits of m flip,
- under the condition that the first type of failure does not happen, the algorithm does not create within a phase a search point where m is active,
- the first search point which activates m is not accepted.

If none of these failures happens, we activate m within the considered phase. The first and third type of failure can be handled by standard techniques. For the second type of failure we compare RLS_p with steps of at most two flipping m-bits on the function f with the same algorithm on the single monomial m. For this purpose we apply the techniques developed in Section 2.

In the following we assume w. l. o. g. that m contains $x_1, \dots, x_d$. The analysis is easier if m has smaller degree.

The first type of failure happens if the first flipping and two further flipping bits belong to m or if three further flipping bits belong to m. Hence, the failure probability for one step can be bounded above by

$$\frac{d}{n}\binom{d-1}{2}p^2 + \frac{n-d}{n}\binom{d}{3}p^3 \le d^3 \cdot \frac{p^2}{n}.$$

The failure probability for all steps of a phase is bounded above by

$$c' \cdot \frac{n}{d} \cdot 2^d \cdot d^3 \cdot \frac{p^2}{n} = c' \cdot 2^d \cdot d^2 \cdot p^2 \le c' \cdot n^c \cdot c^2 \cdot (\log^2 n) \cdot \alpha^2 \cdot n^{-c} \cdot \log^{-2} n.$$

Choosing α small enough, this is bounded by an arbitrary positive constant.

For the third type of failure it is necessary that at least one of the so-called suffix bits $x_{d+1}, \dots, x_n$ flips. However, m is activated during this step. Therefore, we work under the condition that exactly all 0-bits of m, w. l. o. g. $x_1, \dots, x_j$, $j \ge 1$, have flipped. We estimate the probability q that the failure does not happen. Let A_i be the event that bit i flips. Then

$$\begin{aligned} q &= \text{Prob}(\overline{A}_{d+1} \cap \dots \cap \overline{A}_n \mid A_1 \cap \dots \cap A_j \cap \overline{A}_{j+1} \cap \dots \cap \overline{A}_d) \\ &= \frac{\text{Prob}(A_1 \cap \dots \cap A_j \cap \overline{A}_{j+1} \cap \dots \cap \overline{A}_d \cap \overline{A}_{d+1} \cap \dots \cap \overline{A}_n)}{\text{Prob}(A_1 \cap \dots \cap A_j \cap \overline{A}_{j+1} \cap \dots \cap \overline{A}_d)} \end{aligned}$$

The numerator equals

$$\frac{j}{n} \cdot p^{j-1}(1-p)^{n-j} \ge \frac{j}{n} \cdot p^{j-1} \cdot e^{-1}$$

and the denominator equals

$$\frac{j}{n} \cdot p^{j-1}(1-p)^{d-j} + \frac{n-d}{n} \cdot p^j(1-p)^{d-j} \le 2 \cdot \frac{j}{n} \cdot p^{j-1}.$$

Hence, the probability that this failure does not happen is at least $1/(2e)$. This holds also under the condition that the first two types of failure do not happen.

Finally, we apply Lemma 5. The Markov chain Y_0 equals RLS_p^*, namely RLS_p on the monomial m, where the condition holds that no step flips more than two bits of m. The Markov chain Y_1 equals RLS_p^* on the monotone polynomial f which again equals RLS_p under the condition that no step flips more than two bits. Both Markov chains are investigated on the compressed state space $D = \{0, \dots, d\}$. This implies that Y_1 is not time-homogeneous. The transition probabilities at time step t depend on the random current search point X_t. Nevertheless, we denote the transition probabilities by $P_1(i,j)$ and consider only properties which hold for arbitrary search points X_t with the right number of ones among the m-bits.

We prove this lemma by applying Lemma 5 and proving the conditions of Lemma 5, more precisely:

- Y_1 has a relative advantage to Y_0 for c-values such that $c_{\min} \geq 1/(2e)$,
- Y_0 has a $(2e-1)$-advantage,
- $E(\tau_0^i) = O((n/d) \cdot 2^d)$ for all i.

The last claim follows from Theorem 1 which shows that the upper bound $O((n/d) \cdot 2^d)$ holds for all starting points. Lemma 2 implies that Y_0 has a $(2e-1)$-advantage if $d \leq (n-1)/(4e+1)$ which holds for large enough n. We are left with the first claim. According to Definition 3 and the fact that at most two m-bits flip, we have to consider $c(i,j) = P_1(i,j)/P_0(i,j)$ for $j \in \{i-2, i-1, i+1, i+2\}$ and to prove that

- $1/(2e) \leq c(i,i+1) \leq 1$,
- $c(i,i+2) \geq c(i,i-1)$,
- $c(i,i-1) \leq c(i,i+1)$, and
- $c(i,i-2) \leq c(i,i+1)$ (or even $c(i,i-2) \leq c(i,i-1)$).

Since RLS_p^* on m accepts each new string as long as the optimum is not found, $P_1(i,i+1) \leq P_0(i,i+1)$ and $c(i,i+1) \leq 1$. Since RLS_p^* on f accepts a step where one m-bit flips from 0 to 1 and not bit outside m flips, we can apply our calculations on the third failure probability to prove that $c(i,i+1) = P_1(i,i+1)/P_0(i,i+1) \geq 1/(2e)$.

The compare the c-values, consider a search point $a = (b,c)$ with i ones in m, i. e., in b. Assuming that a is the current search point, $P_0(i,j)$ is the probability that RLS_p^* produces a search point with j ones in the prefix. Moreover, $P_1(i,j)$ is the probability of producing a search point with j prefix ones and of accepting it for the fitness function f. Hence, $c(i,j)$ is the conditional probability of accepting a string with j prefix ones if it is produced. Hence, we can prove the inequalities for each fixed new suffix c'. Let b_ℓ, $1 \leq \ell \leq d-j$, be the string obtained from b by flipping the ℓ-th 0-bit. Under the considered conditions $c(i,i+1) = k/(d-j)$, where k is the number of indices ℓ such that RLS_p^* accepts (b_ℓ, c') on f. Let $b_{\ell,\ell'}$, $1 \leq \ell < \ell' \leq d-j$, be the string obtained from b by flipping the ℓ-th and the ℓ'-th 0-bit. Let k^* the the number of pairs (ℓ, ℓ') such that RLS_p^* accepts $(b_{\ell,\ell'}, c')$. Then $c(i,i+2) = k^*/\binom{d-i}{2}$. If (b_ℓ, c') is accepted, each (b_{ℓ,ℓ^*}, c') or $(b_{\ell^*,\ell}, c')$ is accepted. Hence, $c(i,i+2) \geq c(i,i+1)$ since it is more likely to hit one of the k positions when choosing two of them than when only choosing one. A dual argument proves $c(i,i-2) \le c(i,i-1)$. For the inequality $c(i,i-1) \le c(i,i+1)$ we distinguish two cases. If (b,c') is accepted, $c(i,i+1)=1$ since flipping a 0-bit can only increase the f-value. Hence, $c(i,i-1) \le 1 = c(i,i+1)$. If (b,c') is not accepted, $c(i,i-1)=0$ since flipping a 1-bit can only decrease the f-value. Hence, $c(i,i-1) = 0 \le c(i,i+1)$. Altogether, we have proved the lemma. □

When optimizing a monotone polynomial with RLS$_p$, it is not enough to activate each of its monomials once. Activated monomials can get lost while other monomials are activated. We obtain an upper bound of $O((n^2/d)\cdot 2^d)$ on the expected optimization time if p is small enough. This is not far from the lower bound $\Omega((n/d)\cdot\log(n/d+1)\cdot 2^d)$.

Theorem 5 *The expected optimization time of RLS$_p$ on a monotone polynomial f of degree $d \le c\log n$ is bounded above by $O((n^2/d)\cdot 2^d)$ if*

$$0 < p \le \min\bigl\{1/(2dn) - \gamma,\ \alpha/(n^{c/2}\cdot\log n)\bigr\}$$

for the constant α from Lemma 8 and each constant $\gamma > 0$.

Proof: In order to measure the progress of the optimization process, the f-value of the current search point is not the right choice. An f-value of v can be due to a single monomial of degree 1 or to many monomials of large degree. Here we choose the "pseudo-fitness" function $E_f(a)$ counting the number of *essential ones* of a (with respect to f). A 1-entry of a is called essential if it is contained in an activated monomial of f. All other 1-entries may flip to 0 without decreasing the f-value and the optimization process is not influenced by inessential ones until they get essential ones. An essential 1-entry can become inessential, but this implies that in the same step some monomial is activated. Otherwise, the f-value would decrease and the new search point would not be accepted.

If we have not found the optimum, there is a passive monomial of f. The number of essential ones does not change until a passive monomial is activated. Such *steps* are called *essential*. By Lemma 8, it is sufficient to prove a bound of $O(n)$ on the number of essential steps.

The proof of this bound is an application of Lemma 6. Such an approach is sometimes called *drift analysis* (see Hajek (1982), Sasaki and Hajek (1988), and He and Yao (2001)). Let X_i be the number of essential ones after the i-th essential step, i. e., X_0 is the number of essential ones after initialization. Let $D_0 = X_0$ and $D_i = X_i - X_{i-1}$ for $i \ge 1$. Then we are interested in τ, the minimal i where $D_0 + D_1 + \cdots + D_i = n$. Some conditions of Lemma 6 are verified easily. We have $|D_i| \le n$ and $E(\tau) < \infty$ since we have a probability of at least p^n to create the optimal string in each step. If we can prove that $E(D_i \mid \tau \ge i) \ge \varepsilon$ for some $\varepsilon > 0$, Lemma 6 implies that $E(\tau) = O(n)$.

At least one monomial is activated in an essential step. This implies that at least one bit turns from inessential into essential. We have to bound the expected number of bits turning from essential into inessential. Since the assumption that the new search point is accepted only decreases this number, we consider the number of flipping ones under the condition that a 0-bit is flipped. A flipping one may turn other essential bits into inessential.

Let Y be the random number of additional bits flipped by RLS_p under the assumption that a specified bit (activating a monomial) has flipped. The specified bit can flip since it is chosen immediately by RLS_p as a flipping bit (probability $1/n$) or since it is not chosen immediately and flips nevertheless (probability $p \cdot (1-1/n)$. In the first case Y is binomially distributed with respect to $n-1$ and p and has an expected value of $(n-1) \cdot p$. In the second case, $Y = Z + 1$ where Z is binomially distributed with respect to $n-2$ and p and the expected value of Y equals $1 + (n-2) \cdot p$. Altogether,

$$
\begin{aligned}
E(Y) &= \left[(n-1) \cdot p \cdot \frac{1}{n} + (1 + (n-2) \cdot p) \cdot p \cdot \left(1 - \frac{1}{n}\right)\right] \Big/ \left(\frac{1}{n} + p \cdot \left(1 - \frac{1}{n}\right)\right) \\
&= [p + (1 + (n-2) \cdot p) \cdot p] \Big/ \left(\frac{1}{n-1} + p\right) \\
&\leq (2p + p^2 n) \Big/ \left(\frac{1}{n} + p\right) \leq \frac{1-\varepsilon}{d}
\end{aligned}
$$

for some $\varepsilon > 0$ since $p \leq 1/(2dn) - \gamma$ for some constant $\gamma > 0$.

Finally, we are interested in the expected number of bits turning from essential into inessential if $Y = i$. In the worst case all these i bits are flipping from 1 to 0. Since we do not take into account that some new search point is not accepted, each subset of size i of the essential ones has the same probability of being the flipping ones. Let A contain the positions of the essential ones. Let L be the random number of $j \in A$ turning into inessential if a bit at a random position $k \in A$ flips to 0. We claim that $E(L) \leq d$. We denote by L_k the number of $j \in A$ turning into inessential if bit k flips. If the claim holds and we flip i of these positions one after the other, we lose on average not more than $i \cdot d$ essential ones. Since the average number of flipping essential ones is bounded above by $(1-\varepsilon)/d$, the expected number of bits turning from essential into inessential is bounded above by $1-\varepsilon$. Since at least one bit turns from inessential into essential, this implies $E(D_i \mid \tau \geq i) \geq \varepsilon$ and the theorem.

We still have to prove the claim $E(L) \leq d$. We consider a matrix of size $|A| \times |A|$. The (k,j)-entry equals 1 if bit j turns into inessential if bit k flips. Otherwise, the entry equals 0. The sum of row k equals L_k and we are interested in the average number of ones per row which equals (by the book-keeping method) the average number of ones per column. In column j we have ones for all k whose flip makes bit j inessential. The column sum is bounded by d since for each (k,j)-entry equal to 1 the k-th variable has to be in each activated monomial containing the j-th variable. If each column sum is bounded by d, so is the average column sum. □

7 On the Analysis of the (1+1) EA

The bounds in the last section hold only if p is small enough. In particular, Theorem 5 cannot be transferred to the (1+1) EA. However, a result corresponding to Lemma 8 can be proved.

Lemma 9 *Let f be a monotone polynomial of degree d and let m be one of its monomials. There is a constant a such that the* (1+1) *EA activates m in an expected number of $O((n/d) \cdot 2^d)$ steps if $d \leq 2\log n - 2\log\log n - a$.*

Proof: The proof follows the same structure as the proof of Lemma 8. We only describe where we need different arguments.

First, the probability of at least three flipping m-bits is bounded above by $\binom{d}{3} \cdot (1/n)^3 \leq d^3 n^{-3}/6$ for the (1+1) EA. The probability that this happens in a phase of length $c \cdot (n/d) \cdot 2^d$ is bounded by $(c/6) \cdot d^2 \cdot 2^d/n^2$ which can be made smaller than 1 by choosing a large enough.

Second, the probability that no bit outside m flips is at least $(1-1/n)^{n-1} \geq 1/e$. Also Lemma 8 can be applied with a value $c_{\min} \geq 1/e$ and an $(e-1)$-advantage of Y_0. It is again possible to apply Theorem 1. Instead of Lemma 2, Lemma 1 is applied. Here it is sufficient that $d \leq (n-1)/e$.

Finally, the argument that Y_1 has a relative advantage to Y_0 for c-values such that $c_{\min} \geq 1/e$ can be used in the same way here. □

Since the expected number of flipping bits is too large, we only obtain an upper bound on the expected optimization time which depends on the number of f-monomials with non-zero weights.

Theorem 6 *The expected optimization time of the* (1+1) *EA on a monotone polynomial with N monomials and degree $d \leq 2\log n - 2\log\log n - a$ for the constant a from Lemma 9 is bounded above by $O(N \cdot (n/d) \cdot 2^d)$.*

Proof: Here we use the method of measuring the progress by fitness layers. Let the positive weights of the N monomials be sorted, i. e., $w_1 \geq \cdots \geq w_N > 0$. We partition the search space $\{0,1\}^n$ into $N+1$ layers $L_0, \ldots, L_N$, where

$$L_i = \{a \mid w_1 + \cdots + w_i \leq f(a) < w_1 + \cdots + w_{i+1}\}$$

for $i < N$ and L_N contains all optimal search points with f-value $w_1 + \cdots + w_N$. The search process leaves each layer L_i, $i < N$, at most once. Hence, it is sufficient to prove a bound of $O((n/d) \cdot 2^d)$ on the expected time to leave L_i, $i < N$. Let $a \in L_i$. Then there exists some $j \leq i+1$ such that the monomial m_j corresponding to w_j is passive. By Lemma 9, the expected time until m_j is activated is bounded by $O((n/d) \cdot 2^d)$. Either we have left L_i before or some monomial is deactivated during the step activating m_j or we leave L_i with the step activating m_j. Since the probability that no bit outside m_j flips in the step activating m_j is at least e^{-1}, we have a constant probability to leave L_i within $O((n/d) \cdot 2^d)$ steps. Otherwise, we stay in L_i and can repeat the arguments for a next phase with another passive monomial m_k, $k \leq i+1$. The expected number of phases is bounded by e and the theorem is proved. □

Conclusions

We have analyzed randomized search heuristics like random local search and a simple evolutionary algorithm on monotone polynomials. The conjecture is that all these algorithms optimize monotone polynomials of degree d in an expected number of $O((n/d) \cdot \log(n/d+1) \cdot 2^d)$ steps. It has been shown that some functions need that amount of time. Moreover, for random local search the bound has been verified. If the expected number of flipping bits per step is limited, a little weaker bound is proved. However, for the evolutionary algorithm only a bound depending on the number of monomials with non-zero weights has been obtained. Although there is room for improvements, the bounds and methods are a step to understand how randomized search heuristics work on simple problems.

References

Droste, S., Jansen, T., Tinnefeld, K., and Wegener, I. (2002). A new framework for the valuation of algorithms for black-box optimization. In *Proc. of Foundations of Genetic Algorithms 7 (FOGA 2002)*, 197–214. Final version of the proceedings to appear in 2003.

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

Feller, W. (1971). *An Introduction to Probability Theory and its Applications.* Wiley, New York.

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

Hajek, B. (1982). Hitting-time and occupation-time bounds implied by drift analysis with applications. *Advances in Applied Probability*, **14**, 502–525.

He, J. and Yao, X. (2001). Drift analysis and average time complexity of evolutionary algorithms. *Artificial Intelligence*, **127**, 57–85.

Jansen, T. and Wegener, I. (2001). Real royal road functions – where crossover provably is essential. In *Proc. of GECCO 2001, Genetic and Evolutionary Computation Conference*, 375–382.

Jansen, T. and Wegener, I. (2002). The analysis of evolutionary algorithms – a proof that crossover really can help. *Algorithmica*, **34**, 47–66.

Mitchell, M., Forrest, S., and Holland, J. H. (1992). The royal road for genetic algorithms: Fitness landscapes and GA performance. In Varela, F. J. and Bourgine, P. (eds.), *Towards a Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life*, 245–254. MIT Press, Paris.

Mitchell, M., Holland, J. H., and Forrest, S. (1994). When will a genetic algorithm outperform hill climbing. In Cowan, J. D., Tesauro, G., and Alspector, J. (eds.), *Advances in Neural Information Processing Systems*, vol. 6, 51–58. Morgan Kaufmann.

Motwani, R. and Raghavan, P. (1995). *Randomized Algorithms*. Cambridge University Press.

Sasaki, G. H. and Hajek, B. (1988). The time complexity of maximum matching by simulated annealing. *Journal of the ACM*, **35**, 387–403.

Wegener, I. (2001). Theoretical aspects of evolutionary algorithms (invited paper). In *Proceedings of ICALP 2001*, no. 2076 in LNCS, 64–78.

Wegener, I. and Witt, C. (2002). On the analysis of a simple evolutionary algorithm on quadratic pseudo-boolean functions. To appear in Journal of Discrete Algorithms.