Rigorous Analyses of Fitness-Proportional Selection for Optimizing Linear Functions

Edda Happ Daniel Johannsen Christian Klein Frank Neumann
Max-Planck-Institut für Informatik
Campus E 1 4
66123 Saarbrücken, Germany

ABSTRACT

Rigorous runtime analyses of evolutionary algorithms (EAs) mainly investigate algorithms that use elitist selection methods. Two algorithms commonly studied are Randomized Local Search (RLS) and the (1+1) EA and it is well known that both optimize any linear pseudo-Boolean function on n bits within an expected number of $O(n \log n)$ fitness evaluations. In this paper, we analyze variants of these algorithms that use fitness proportional selection.

A well-known method in analyzing the local changes in the solutions of RLS is a reduction to the gambler's ruin problem. We extend this method in order to analyze the global changes imposed by the (1+1) EA. By applying this new technique we show that with high probability using fitness proportional selection leads to an exponential optimization time for any linear pseudo-Boolean function with non-zero weights. Even worse, all solutions of the algorithms during an exponential number of fitness evaluations differ with high probability in linearly many bits from the optimal solution.

Our theoretical studies are complemented by experimental investigations which confirm the asymptotic results on realistic input sizes.

Categories and Subject Descriptors: F.2 [Theory of Computation]: Analysis of Algorithms and Problem Complexity

General Terms: Theory, Algorithms

Keywords: Running time analysis, Selection, Theory

1. INTRODUCTION

Selection methods play an important role when designing successful evolutionary algorithms. This can be observed in several applications. In this paper, we examine the use of fitness-proportional selection (also known as roulette wheel selection) which has been introduced in the context of genetic algorithms (see e.g. [5]). Here, each individual of the population is selected with a probability that depends on its fitness in correlation to the fitness of the other individuals.

Our goal is to examine the mentioned method by rigorous runtime analyses. Such analyses have been mainly carried out for evolutionary algorithms using elitist selection strategies, that is, only the currently best solutions are used in the next generation. In the case that the parent and offspring population are considered for selecting the individuals of the next generation, the best solution found so far cannot get lost and unimodal functions attaining not too many fitness values are easy to optimize [1].

Rigorous runtime analyses for evolutionary algorithms using elitist selection strategies are available for pseudo-Boolean functions [10, 13, 14] as well as for several well-known combinatorial optimization problems [4, 11, 15]. The most widely studied algorithm for such theoretical investigations is the (1+1) EA which works with a population of size one and produces one offspring in each iteration. Often also a simplified algorithm called Randomized Local Search (RLS) is considered which flips exactly one randomly chosen bit in each mutation step. Both algorithms optimize all linear pseudo-Boolean functions in an expected number of $O(n \log n)$ iterations.

Our aim is to analyze the properties of fitness proportional selection with respect to the runtime behavior. Therefore, we replace the elitist selection procedure of RLS and the (1+1) EA by fitness proportional selection and call the algorithms RLS_F and EA_F respectively. This implies that the fitness of the next solution may be smaller than the fitness of the current one. The objective of our investigations is to study how this influences the runtime behavior. Initial investigations into this direction have been carried out by He and Yao [8] who have shown that RLS_F needs in expectation an exponential number of iterations to reach an optimal search point for the function OneMax. First, we generalize the investigations in [8]. In particular, we show that with high probability RLS_F and EA_F on OneMax are even unable to improve the initial solution within a polynomial number of steps by any constant factor greater than one. Later on, we show that both algorithms are actually inefficient on general linear pseudo-Boolean functions with a linear number of non-zero weights. These theoretical results are complemented with experimental investigations that show that the asymptotic behavior can also be observed for realistic input sizes.

In our analyses, we make use of results on the gambler's ruin problem [2] which has already been considered for the analysis of evolutionary computation methods before [6, 12].

These results can be directly applied to algorithms working with mutation operators that flip one single bit in each iteration. However, a generalization to more general mutation operators is not straight-forward. Another powerful tool for analyzing evolutionary algorithms is drift analysis [7]. We study its relation to the gambler's ruin problem in greater detail and point out that drift analysis can be used to generalize the results of the gambler's ruin process to mutation steps that flip more than a single bit. This correlation may be of independent interest as we expect it to be useful for further analyses of evolutionary computation methods in the future.

The outline of the paper is as follows. In Section 2, we introduce the algorithms that are subject to our analyses. Section 3 points out the correlation between the gambler's ruin problem and drift analysis. In Section 4, we show that RLS_F and the EA_F are unable to get close to the optimal solution of the function OneMax. These negative results are extended to the whole class of linear functions with a linear number of non-zero weights in Section 5. Our experimental results are shown in Section 6 and finally we finish with some conclusions.

2. ALGORITHMS

To examine the effect of fitness proportional selection, we consider two basic algorithms that have already been studied with respect to elitist strategies. Our algorithms work with a population of size one and produce in every iteration one single offspring by mutation. Both have the objective to optimize a non-negative pseudo-boolean function $f: \{0,1\}^n \rightarrow \mathbb{R}_0^+$. The first one is called *Fitness-proportional Randomized Local Search* (RLS_F) and was investigated before in [8]. In every mutation step RLS_F flips one single bit and can be defined as follows.

Algorithm 1. *RLS_F*

1. *Choose $x \in \{0,1\}^n$ uniformly at random.*
2. *Repeat*
   (a) *Create x' by flipping one randomly chosen bit of x.*
   (b) *Replace x by x' with probability $\frac{f(x')}{f(x)+f(x')}$.*

Considering mutation-based evolutionary algorithms it is more common to allow to flip more than one bit in a single step. Our algorithm called *EA_F* differs from the previous one by flipping each bit with probability $1/n$.

Algorithm 2. *EA_F*

1. *Choose $x \in \{0,1\}^n$ uniformly at random.*
2. *Repeat*
   (a) *Create x' by flipping each bit of x with probability $1/n$.*
   (b) *Replace x by x' with probability $\frac{f(x')}{f(x)+f(x')}$.*

We compare the algorithms introduced above with the variants using elitist selection procedures. RLS and the (1+1) EA differ from RLS_F and EA_F, respectively, by using the following selection mechanism in step 2.(b) of the algorithms.

Algorithm 3. *Selection for RLS and (1+1) EA*

- *Replace x by x' if and only if $f(x') \geq f(x)$.*

When analyzing the runtime behavior of evolutionary algorithms we consider the *optimization time* of the algorithm which equals the number of fitness evaluations to reach an optimal search point. The expected optimization time refers to the expectation of this value.

3. GAMBLER'S RUIN AND DRIFT ANALYSIS

The gambler's ruin problem and its analysis has been used before in the analysis of evolutionary algorithms. However, a direct application of these results has up to now only been shown for local variants such as RLS. The aim of this section is to recall the results on the gambler's ruin problem and to generalize them to global mutation operators. To achieve this goal, we make use of drift analysis and show how this method can be put into the framework of a gambler's ruin process.

In the classical gambler's ruin problem (see [2]) the following process is studied. Let x be the amount of dollars a gambler owns at the beginning of a series of bets. Every bet the gambler wins one dollar with probability p and looses one dollar with probability $q = 1 - p$. In this setting we are interested in the probability that the gambler wins, that is, that his capital reaches an amount of $z > x$ dollars before it attains the amount of zero dollars.

The following variant of this problem can be easily obtained from the statements in [2].

Theorem 1 (Gambler's Ruin). *Let p be the probability of winning one dollar and $q = 1 - p$ be the probability of loosing one dollar in a single bet and let $\delta = q/p$. Starting with x dollars, the probability of reaching $z > x$ dollars before attaining zero dollars is*

$$p_x = \frac{\delta^x - 1}{\delta^z - 1}.$$

Note, that $p_x \leq \delta^{-(z-x)}$ holds if $\delta > 1$. This result on the gambler's ruin process is well suited to prove lower bounds on the runtime behavior of randomized search heuristics like RLS and RLS_F where in every step only a single bit is flipped. The following theorem makes this precise by stating two conditions that if simultaneously satisfied imply an exponential lower bound on the optimization time of a randomized search heuristic.

Theorem 2 (Local Gambler's Ruin). *Let $X_0, X_1, X_2, \ldots$ with $X_{t+1} - X_t \in \{-1, 0, 1\}$ for all $t \geq 0$ be random variables describing a Markov process over the state space $\mathbb{N}_0$. For constants $a, b \in \mathbb{R}$ with $0 \leq a < b \leq 1$ let the random variable T denote the earliest point in time $t \geq 0$ that satisfies $X_t \leq an$.*

If there exists a constant $\delta > 1$ such that the two conditions

(a) *$P[X_0 \geq bn] = 1 - 2^{-\Omega(n)}$,*

(b) *$P[X_{t+1} - X_t = 1 \mid X_t] \geq \delta \cdot P[X_{t+1} - X_t = -1 \mid X_t]$ for all $t \geq 0$ and $an < X_t < bn$*

hold then $T \geq \delta^{1/3 \cdot (b-a) \cdot n}$ with probability $1 - 2^{-\Omega(n)}$.

PROOF. For $t \geq 0$ let X_t, $0 \leq a < b \leq 1$, and $\delta > 1$ be defined as above and suppose that conditions (a) and (b) hold.

If the statement follows for $X_0 \geq bn$, then it also follows if $X_0 \geq bn$ with probability $1-2^{-\Omega(n)}$. Hence, by condition (a) we suppose that $X_0 \geq bn$.

Let $p = 1/(1+\delta)$ and $q = \delta/(1+\delta)$. Then it holds by condition (b) that $P[X_{t+1} - X_t = -1 \mid X_t] \leq p$ as well as $P[X_{t+1} - X_t = -1 \mid X_t] \geq q$ for all $t \geq 0$ and X_t such that $an < X_t < bn$. Thus, if we replace $P[X_{t+1} - X_t = -1 \mid X_t]$ by p and $P[X_{t+1} - X_t = -1 \mid X_t]$ by q for all $t \geq 0$ and X_t such that $an < X_t < bn$ we do not increase the probability that $T \geq B$ for any $B \geq 0$.

We apply the Gambler's Ruin Theorem as stated above with $z = \lfloor (b-a)\cdot n \rfloor$ and $x = \lfloor z/2 \rfloor$ and choose n sufficiently large such that $z - x \geq 4/9 \cdot (b-a) \cdot n$. Since $\delta = q/p > 1$, the probability p_x of reaching z before attaining zero is at most $\delta^{-(z-x)}$.

Thus, starting with $X_0 \geq bn$, the probability that X_t reaches a value of at most an after passing the value $\lfloor bn \rfloor - x$ is at most $\delta^{-4/9\cdot(b-a)\cdot n}$. Given less then $\delta^{1/3\cdot(b-a)\cdot n}$ such tries, the probability to succeed is still at most $\delta^{-1/9\cdot(b-a)\cdot n}$. Hence, $T \geq \delta^{1/3\cdot(b-a)\cdot n}$ with probability $1 - 2^{-\Omega(n)}$. □

The previous theorem is generalized by the Global Gambler's Theorem below which on its own is a specialization of a variant of the Drift Theorem found in [3].

THEOREM 3 (DRIFT THEOREM). *Let $X_0, X_1, X_2, \ldots$ be the random variables describing a Markov process over a state space S and $g\colon S \to \mathbb{R}_0^+$ a function mapping each state to a non-negative real number. Pick two real numbers $a(n)$ and $b(n)$ depending on a parameter n such that $0 \leq a(n) < b(n)$ holds. Let T be the random variable denoting the earliest point in time $t \geq 0$ such that $g(X_t) \leq a(n)$ holds. If there are constants $\lambda > 0$ and $D \geq 1$ and a polynomial $p(n) > 0$ such that the four conditions*

1. $g(X_0) \geq b(n)$,
2. $b(n) - a(n) = \Omega(n)$,
3. $\mathrm{E}[e^{-\lambda\cdot(g(X_{t+1})-g(X_t))} \mid X_t] \leq 1 - 1/p(n)$ *for all* $t \geq 0$ *and* X_t *such that* $a(n) < g(X_t) < b(n)$,
4. $\mathrm{E}[e^{-\lambda\cdot(g(X_{t+1})-b(n))} \mid X_t] \leq D$ *for all* $t \geq 0$ *and* X_t *such that* $g(X_t) \geq b(n)$

hold then for all time bounds $B \geq 0$

$$P\left[T \leq B\right] \leq e^{\lambda\cdot(a(n)-b(n))} \cdot B \cdot D \cdot p(n)$$

holds.

Next, we consider the Drift Theorem from a different point of view. The following Global Gambler's Ruin Theorem can handle mutation steps flipping more than one bit in a similar way as the Local Gambler's Ruin Theorem handles one-bit flips. The reader may consider the described behavior as a gambler's ruin process where the gambler is allowed to win or loose $j \geq 1$ dollars with a certain probability in every step. Similar as in the Local Gambler's Ruin Theorem, we consider the ratio between the probabilities of winning and losing j dollars for any fixed choice of j.

THEOREM 4 (GLOBAL GAMBLER'S RUIN). *Let $X_0, X_1, X_2, \ldots$ be random variables describing a Markov process over the state space $\mathbb{N}_0$. For constants $a, b \in \mathbb{R}$ with $0 \leq a < b \leq 1$ let the random variable T denote the earliest point in time $t \geq 0$ that satisfies $X_t \leq an$.*

If there exist constants $\delta > 1$ and $C > 0$ such that the three conditions

(a) $P[X_0 \geq bn] = 1 - 2^{-\Omega(n)}$,

(b) $P[X_{t+1} - X_t = j \mid X_t] \geq \delta^j \cdot P[X_{t+1} - X_t = -j \mid X_t]$ *for all* $j \geq 1$, $t \geq 0$, *and* $an < X_t < bn$,

(c) $\sum_{j\geq 1} \delta^j \cdot P[X_{t+1} - X_t = -j \mid X_t] \leq C$ *for all* $t \geq 0$ *and* $X_t \geq bn$

hold then $T \geq \delta^{1/3\cdot(b-a)\cdot n}$ with probability $1 - 2^{-\Omega(n)}$.

PROOF. For $t \geq 0$ let X_t , $0 \leq a < b \leq 1$, and $\delta > 1$ be defined as above, suppose conditions (a) – (c) hold, and let $\Delta_t = X_{t+1} - X_t$.

We apply the Drift Theorem with the same random variables, $g = \mathrm{id}$, $a(n) = an$, $b(n) = bn$, $\lambda = \ln(\delta)/2$, $D = C+1$, and $p(n) = (1+\delta^{-1})/(1-\delta^{-1/2})^2$, i. e., $S = \mathbb{N}_0^+$, $g(X) = X$, $e^\lambda = \delta^{1/2} > 1$, and $p(n)$ is a strictly positive constant. We denote $\mathrm{E}(e^{-\lambda\cdot(g(X_{t+1})-g(X_t))} \mid X_t)$ by μ_t then

$$\mu_t = \sum_{j\in\mathbb{Z}} \delta^{-j/2} \cdot P[\Delta_t = j \mid X_t]\,.$$

We check conditions 1. – 4. of the Drift Theorem.

1. If the statement follows for $X_0 \geq bn$, then it also follows if $X_0 \geq bn$ with probability $1 - 2^{-\Omega(n)}$. Hence, condition 1 is satisfied by condition (a).
2. Clearly, $b(n) - a(n) = (b-a)n = \Omega(n)$.
3. Let $t \geq 0$ and X_t such that $a(n) < g(X_t) < b(n)$, that is, $an < X_t < bn$. We carefully replace $P[\Delta_t = j \mid X_t]$ for all $j \in \mathbb{Z}$ such that μ_t increases. This is done in three steps.

   First, we suppose that $P[\Delta_t = 0 \mid X_t] < 1$ for all $an < X_t < bn$. We can safely do so, because if X_t takes a value such that $P[\Delta_t = 0 \mid X_t] = 1$ then $P[T \geq B] = 1$ for every $B \geq 0$ since the process never leaves this state.

   Second, we ignore whenever $X_t = X_{t+1}$ for some $t \geq 0$ in the definition of T. Clearly, this never increases T. Formally, we replace $P[\Delta_t = 0 \mid X_t]$ by zero and $P[\Delta_t = j \mid X_t]$ by $P[\Delta_t = j \mid X_t]/(1 - P[\Delta_t = 0 \mid X_t])$ for all $j \neq 0$. Thus,

$$\mu_t \leq \sum_{j\neq 0} \delta^{-j/2} \cdot \frac{P[\Delta_t = j \mid X_t]}{1 - P[\Delta_t = 0 \mid X_t]}\,.$$

   Third, since $\delta > 1$, the right hand-side of this inequality never decreases if we increase $P[\Delta_t = -j \mid X_t]$ by some amount and decrease $P[\Delta_t = 0 \mid X_t]$ by the same amount for any $j \geq 1$. Thus, condition (b) implies that μ_t does not decrease if we replace $P[\Delta_t = -j \mid X_t]$ by $\delta^{-j} P[\Delta_t = j \mid X_t]$ for every $j \geq 1$ and $P[\Delta_t = 0 \mid X_t]$ by $1 - \sum_{j\geq 1}(1+\delta^{-j}) \cdot P[\Delta_t = j \mid X_t]$. Hence,

$$\mu_t \leq 1 - \frac{(1-\delta^{-1/2})^2}{1 - P[\Delta_t = 0 \mid X_t]} \cdot \sum_{j\geq 1} P[\Delta_t = j \mid X_t]\,.$$

Now, again invoking condition (b),

$$P[\Delta_t = j \mid X_t] \geq \frac{P[\Delta_t = j \mid X_t] + P[\Delta_t = -j \mid X_t]}{1+\delta^{-1}}$$

and since $P[\Delta_t = 0 \mid X_t] = 1 - \sum_{j\neq 0} P[\Delta_t = j \mid X_t]$ we have $\mu_t \leq 1 - 1/p(n)$.

4. Let $t \geq 0$ and X_t such that $g(X_t) \geq b(n)$, that is, $X_t \geq bn$. Then μ_t increases if we reduce $P[\Delta_t = j]$ to zero for all $j \geq 1$ in combination with replacing $P[\Delta_t = 0]$ by one. Thus,

$$\mu_t \leq 1 + \sum_{j\geq 1} \delta^j P[\Delta_t = -j \mid X_t] \overset{(c)}{\leq} D\,.$$

Since conditions 1.–4. of the Drift Theorem hold,

$$P[T \leq \delta^{1/3\cdot(b-a)\cdot n}] \leq \frac{(C+1)(1-\delta^{-1/2})^2}{1+\delta^{-1}} \cdot \delta^{-1/6\cdot(b-a)\cdot n}\,.$$

This is bounded from above by $2^{-\Omega(n)}$, hence the statement of this theorem follows. □

4. ANALYSES FOR ONEMAX

The function OneMax which counts the number of one-bits in a bit string, OneMax: $\{0,1\}^n \to \mathbb{N}_0$

$$\text{OneMax}(x) = \sum_{i=1}^{n} x_i\,,$$

is the first one for which rigorous results with respect to the optimization time behavior of the (1+1) EA have been shown. An upper bound of $O(n \log n)$ can be found in [10] and a matching lower bound can easily be obtained by using results on the coupon collectors problem [9]. The $\Theta(n \log n)$ bound also holds for RLS as the analyses for the (1+1) EA rely on mutation steps flipping in each iteration one single bit. The goal of this section is to point out that the variants RLS_F and EA_F are unable to optimize OneMax.

We start by analyzing the behavior of RLS_F on OneMax. It has already been shown in [8] that the expected optimization time of RLS_F on OneMax is exponential. Actually, we show a much stronger result: With high probability the initial solution of roughly $n/2$ one–bits is improved only sublinearly in n for an exponential number of steps.

Proposition 5. *Let $0 < \alpha < 1/2$, be a constant and let $X \leq \alpha n$ be the number of zero-bits of the current search point and X' the number of zero-bits of the next search point of the EA_F. Then for sufficiently large n*

1. $P[X' - X = 1 \mid X] \geq \frac{1}{4}$,
2. $P[X' - X = -1 \mid X] \leq \frac{1}{4} \cdot \left(\frac{1}{2} + \alpha\right) < \frac{1}{4}$.

Proof. Let α, X, and X' be given as in the statement.

1. Since there are $n - X \geq (1-\alpha)\cdot n$ one-bits in the current solution, the probability that one of them is flipped by RLS_F is at most $1-\alpha$. This new solution is accepted with probability

$$\frac{n-X-1}{2(n-X)-1} \geq \frac{1}{2}\cdot\left(1 - \frac{1}{2(1-\alpha)n-1}\right)\,.$$

Since $1-\alpha > 1/2$ and $\lim_{n\to\infty} 1/(2(1+\alpha)n+1) = 0$, $P[X' - X = 1 \mid X] \geq \frac{1}{4}$ for sufficiently large n.

2. The probability that one of the X zero-bits of the current solution is flipped is at most α. This new solution is accepted with probability

$$\frac{n-X+1}{2(n-X)+1} \leq \frac{1}{2}\cdot\left(1 + \frac{1}{2(1-\alpha)n+1}\right)\,.$$

Thus, since $\alpha < 1/2$ and $\lim_{n\to\infty} 1/(2(1+\alpha)n+1) = 0$, $P[X' - X = 1 \mid X] \leq \frac{1}{4}\cdot\left(\frac{1}{2}+\alpha\right) < \frac{1}{4}$ for sufficiently large n.

□

Using the previous proposition we are able to prove the following theorem.

Theorem 6. *Let $\epsilon > 0$ be a constant. Then there exists asecond constant $\gamma > 1$ such that with probability $1 - 2^{-\Omega(n)}$ in all solutions produced in the first γ^n steps of RLS_F at most $(1+\epsilon)\cdot n/2$ bits are set correctly.*

Proof. Because of monotonicity we can suppose $\epsilon < 1/2$. For $t \geq 0$ let X_t be the number of zero-bits of the solution x after t iterations. We apply Theorem 2 with parameters $a = \frac{1}{2}\cdot(1-\epsilon)$, $b = \frac{1}{2}\cdot(1-\frac{1}{2}\cdot\epsilon)$, $\delta = 1+\epsilon/4$, and sufficiently large n.

(a) Since one-bits and zero-bits occur with the same probability in the initial solution, Chernoff bounds imply

$$P[X_0 \geq bn] = 1 - e^{-\Omega(n)}\,.$$

(b) Let $t \geq 0$ and $X_t < bn$. Then by Proposition 5 with $\alpha = b$

$$\frac{P[X_{t+1} - X_t = 1 \mid X_t]}{P[X_{t+1} - X_t = -1 \mid X_t]} = \frac{2}{1+2\alpha} = 1 + \frac{\epsilon}{4+\epsilon} \geq \delta\,.$$

The statement follows. □

The EA_F can flip more than one bit in a single mutation step. To handle such mutations we consider the probability that an offspring of the current solution x increases (or decreases) the number of one-bits by j and is accepted.

Proposition 7. *Let $0 < \alpha \leq 1$ and let $X \leq \alpha n$ be the number of zero-bits of the current search point and X' the number of zero-bits of the next search point of the EA_F. Then for all $j \in \{1,\ldots,X\}$ and sufficiently large n*

1. $P[X' - X = j \mid X] \geq (2\cdot(1-\alpha))^j \cdot P[X' - X = -j \mid X]$ *if* $\alpha < 1/2$,
2. $P[X' - X = -j \mid X] \leq \frac{1}{j!}$.

Proof. Let $X \leq \alpha n$, $1 \leq j \leq X$, and X' be defined as above.

1. Let $\alpha < 1/2$. The probability that $X' = X + j$ is given by the probability p_j that the new solution has $X + j$ zero-bits times the probability q_j that the new solution is accepted. Similarly, the probability that $X' = X - j$ is given by the probability p_{-j} that the new solution has $X - j$ zero-bits times the probability q_{-j} that this new solution is accepted.

   Now, p_j is at least the probability that $j + k$ one-bits and k zero-bits in the current solution are flipped for some $0 \le k \le X-j \le \min\{X, n-X-j\}$ and hence at least

$$\sum_{k=0}^{X-j} \binom{n-X}{k+j} \cdot \binom{X}{k} \cdot \left(\frac{1}{n}\right)^{j+2k} \cdot \left(1-\frac{1}{n}\right)^{n-j-2k},$$

while p_{-j} is exactly the probability that $j+k$ zero-bits and k one-bits in the current solution are flipped for some $0 \le k \le X-j$ and hence

$$\sum_{k=0}^{X-j} \binom{X}{k+j} \cdot \binom{n-X}{k} \cdot \left(\frac{1}{n}\right)^{j+2k} \cdot \left(1-\frac{1}{n}\right)^{n-j-2k}.$$

Then $p_j \ge \left(\frac{1-\alpha}{\alpha}\right)^j \cdot p_{-j}$ since for $X \le \alpha n < n/2$

$$\binom{n-X}{k+j} \cdot \binom{X}{k} \ge \left(\frac{n-X}{X}\right)^j \cdot \binom{X}{k+j} \cdot \binom{n-X}{k}.$$

Next, $q_j = \frac{(n-X)-j}{2(n-X)-j}$ and $q_{-j} = \frac{(n-X)+j}{2(n-X)+j}$. Thus,

$$\frac{q_j}{q_{-j}} = 1 - \frac{2(n-X)j}{2(n-X)^2 + (n-X)j - j^2}$$

Since $j \le n-X$, $n-X \ge (1-\alpha)n$, and $\alpha < 1/2$ the following three inequalities hold for sufficiently large n

$$\frac{q_j}{q_{-j}} \ge 1 - \frac{j}{(1-\alpha)n} \ge e^{-\frac{2j}{(1-\alpha)n}} \ge (2\alpha)^j.$$

It follows that $P[X'-X=j \mid X] = p_j \cdot q_j$ is at least $(2\cdot(1-\alpha))^j \cdot p_{-j} \cdot q_{-j} = (2\cdot(1-\alpha))^j \cdot P[X'-X=-j \mid X]$.

2. The probability that $X' = X-j$ is at most the probability that j of the X zero-bits of the current solution are flipped which is $\binom{X}{j} \cdot \frac{1}{n^j} \le \frac{1}{j!}$.

□

Using the previous proposition, we are able to apply the Global Gambler's Ruin Theorem from Section 3 to show that also the EA$_F$ is with high probability not able to get close to the optimal solution within an exponential number of steps.

THEOREM 8. *Let $\epsilon > 0$ be a constant. Then there exists a constant $\gamma > 1$ such that with probability $1 - 2^{-\Omega(n)}$ in all solutions produced in the first γ^n steps of the EA$_F$ at most $(1+\epsilon) \cdot n/2$ bits are set correctly.*

PROOF. For $t \ge 0$ let X_t be the number of zero-bits of the solution x after t iterations, then $X_0, X_1, X_2, \ldots$ is a Markov process. We apply Theorem 4 with parameters $a = \frac{1}{2}\cdot(1-\epsilon)$, $b = \frac{1}{2} \cdot (1-\frac{\epsilon}{2})$, $\delta = 1 + \frac{\epsilon}{2}$, $C = e^\delta$, and n sufficiently large.

(*a*) $P[X_0 \ge bn] = 1 - e^{-\Omega(n)}$ by the Chernoff bounds.

(*b*) $P[X_{t+1} - X_t = j \mid X_t] \ge \delta^j \cdot P[X_{t+1} - X_t = -j \mid X_t]$ hold for all $t \ge 0$, $X_t < bn$, and $j \in \{1, \ldots, X_t\}$ by Proposition 7.1 with $\alpha = b$.

(*c*) $\sum_{j \ge 1} \delta^j P[X_{t+1} - X_t = -j \mid X_t] \le \sum_{j \ge 1} \frac{\delta^j}{j!} \le e^\delta = C$ holds for all $t \ge 0$ and $X_t \ge bn$ by Proposition 7.2.

The statement follows. □

5. LINEAR FUNCTIONS

A pseudo-boolean function $f\colon \{0,1\}^n \to \mathbb{R}$ is *linear* if there are $\tau, w_1, \ldots, w_n \in \mathbb{R}$ such that for every $x \in \{0,1\}^n$

$$f(x) = \tau + \sum_{i=1}^n w_i x_i.$$

The analyses of the optimization times of RLS and the (1+1) EA optimizing linear functions can be found in [1]. Both algorithms find an optimal solution in an expected optimization time of $O(n \log n)$. A matching lower bound of $\Omega(n \log n)$ holds if the number of non-zero weights is linear with respect to the number of bits. In contrast, we show that with high probability the optimization time of RLS$_F$ and the EA$_F$ is exponential in this case.

If a weight w_i is zero, then the corresponding bit x_i does not contribute to $f(x)$. Therefore, we only consider functions that have non-zero weights. Similar to the $\Omega(n \log n)$ for RLS and the (1+1) EA, all our results also hold for functions with a linear number of non-zero weights.

The investigation of RLS and the (1+1) EA in [1] restricts its analysis to linear functions with non-negative weight since both algorithms treat one-bits and zero-bits symmetrically. For every bit x_i with negative weight w_i it is possible to replace x_i by $1 - x_i$, w_i by $|w_i|$ and τ by $\tau - |w_i|$. Thus, all weights of the resulting function are non-negative. Furthermore, the resulting function has the same values as the original function, only that zero and one are swapped for all bits that originally had negative weights. Since RLS$_F$ and the EA$_F$ as well treat zero-bits and one-bits symmetrically, from now on we suppose that the weights $w_1, \ldots, w_n$ of a linear function are positive. The behavior of RLS and the (1+1) EA is invariant under change of τ, thus in the analysis of these algorithms τ is set to zero.

This is not true for the corresponding algorithms using fitness-based selection. Still, since RLS$_F$ and the EA$_F$ are only defined on non-negative functions, we may suppose that τ is non-negative as well. Clearly, if $\tau, w_1, \ldots, w_n$ are non-negative then the function $f(x) = \tau + \sum_{i=1}^n w_i x_i$ is maximized if all bits are set to one.

It is easy to see that it is not possible to guarantee a bad approximation ratio based on the the solutions' fitnesses. Consider functions where one weight is substantially larger than the others. Both, RLS$_F$ and the EA$_F$ need expected linear time to set the corresponding bit to one, thus achieving an arbitrarily good approximation. Therefore, we take the number of ones obtained by the algorithms as a quality measure. Our aim is to prove lower bounds on the optimization time behavior until a certain number of ones has been achieved.

The following lemma will be useful to prove lower bounds on the optimization time of RLS$_F$ and the EA$_F$ on the class of linear functions.

LEMMA 9. *Let $\epsilon > 0$ and $f : \{0,1\} \to \mathbb{R}_0^+$ be a linear function with non-negative weights and let $x \in \{0,1\}^n$ be a point with k one-bits. Then at least $(1-\epsilon)k$ one-bits of x have weight at most $f(x)/\epsilon k$.*

PROOF. Assume that strictly less than $(1-\epsilon)k$ one-bits of x have weight at most $f(x)/\epsilon k$. Then there are at least ϵk one-bits of x with weight strictly more than $f(x)/\epsilon k$. But then $f(x) > \epsilon k \cdot f(x)/\epsilon k = f(x)$ which clearly is a contradiction. □

We first consider RLS$_F$. Using the Local Gambler's Ruin Theorem from Section 3, we are going to show that the optimization time of this algorithm is exponential with high probability. To show this result, we bound the probabilities of increasing or decreasing the number of ones in the current solution in a similar way as done for the function OneMax.

Proposition 10. *Let $0 < \alpha < 1/3$ and $f : \{0,1\} \to \mathbb{R}_0^+$ be linear with strictly positive weights. Let $X \leq \alpha n$ be the number of zero-bits of the current and X' of the next search point of RLS$_F$. Then for sufficiently large n*

1. $P[X' - X = 1 \mid X] \geq \frac{1}{3}$,
2. $P[X' - X = -1 \mid X] \leq \alpha$.

Proof. Let $X \leq \alpha n$ and X' with $\alpha < 1/3$ be defined as above.

1. Let $\epsilon = (1 - 3\alpha)/6(1 - \alpha)$. By Lemma 9 the current search point has at least $(1 - \epsilon)(1 - \alpha)n$ one-bits each of weight at most $f(x)/\epsilon(1 - \alpha)n$. Thus, one of these bits is flipped with probability at least $1/6 \cdot (5 - 3\alpha)$ and the next search point is accepted with probability at least
$$\frac{f(x) - f(x)/\epsilon(1-\alpha)n}{2f(x) - f(x)/\epsilon(1-\alpha)n} = \frac{(1-3\alpha)n - 6}{2(1-3\alpha)n - 6}\,.$$
The statement follows for $n \geq 18(1 - \alpha)/(1 - 3\alpha)^2$.
2. The probability that $X' = X - 1$ is at most $X/n \leq \alpha$ which is the probability that one of the zero-bits of the current search point x is flipped.

□

Using the previous proposition we are able to prove the following theorem.

Theorem 11. *Let $f : \{0,1\} \to \mathbb{R}_0^+$ be linear with non-zero weights. Then for every $\epsilon > 0$ there exists a constant $\gamma > 1$ such that with probability $1 - 2^{-\Omega(n)}$ in all solutions produced by RLS$_F$ in the first γ^n steps at most $(2/3 + \epsilon)n$ bits are set correctly.*

Proof. As discussed before, we suppose that all weights of f are strictly positive and hence that in an optimal solution of f all bits have the value one. For $t \geq 0$ let X_t be the number of zero-bits of the solution x after t iterations.

We apply Theorem 2 with $a = 1/3 - \epsilon$, $b = 1/3 - \epsilon/2$, and $\delta = 1/3b$ with $\delta \geq e^{3\epsilon/2} > 1$.

(*a*) Due to the Chernoff bounds $P[X_0 \geq bn] = 1 - 2^{-\Omega(n)}$, since in the initial search point one-bits and zero-bits occur with the same probability.

(*b*) Let $t \geq 0$ and X_t such that $an < X_t < bn$. Then $P[X_{t+1} - X_t = 1 \mid X_t] \geq \delta \cdot P[X_{t+1} - X_t = -1 \mid X_t]$ by Proposition 10 with $\alpha = 1/3 - \epsilon/2$.

The statement follows. □

For the EA$_F$ it is necessary to handle mutation steps that flip more than a single bit. The aim is to apply the Global Gambler's Ruin Theorem from Section 3. Therefore, we bound the probabilities of increasing or decreasing the number of one-bits by j for each fixed j.

Proposition 12. *Let $0 < \alpha \leq 1$ and let $f : \{0,1\} \to \mathbb{R}_0^+$ be linear with strictly positive weights. Let $X \leq \alpha n$ be the number of zero-bits of the current and X' of the next search point of EA$_F$. Let $1 \leq j \leq X$, then*

1. $P[X' - X = j \mid X] \geq \frac{(1-3\alpha)^j}{(3ej)^j}$ *for* $\alpha < 1/3$,
2. $P[X' - X = -j \mid X] \leq \frac{\alpha^j}{j!} \leq \frac{(\alpha e)^j}{j^j}$.

Proof. Let $X \leq \alpha n$ and X' with $0 < \alpha \leq 1$ be defined as above and let $j \in \{1, \ldots, X\}$.

1. Let $\alpha < 1/3$ and $\epsilon = 2\alpha/(1 - \alpha)$. By Lemma **??** the current search point has $k \geq (1 - 3\alpha)n$ one-bits with weight at most $f(x)/2\alpha n$. Thus, the probability that exactly j of these bits are flipped is at least
$$\binom{k}{j} \cdot \frac{1}{n^j} \cdot \left(1 - \frac{1}{n}\right)^{n-j} \geq \frac{1}{e} \cdot \left(\frac{1-3\alpha}{j}\right)^j$$
and the next search point is accepted with probability at least
$$\frac{f(x) - j \cdot f(x)/2\alpha n}{2f(x) - j \cdot f(x)/2\alpha n} = \frac{2\alpha n - j}{4\alpha n - j} \overset{j \leq \alpha n}{\geq} 1/3\,.$$
2. The probability that $X' = X - j$ is at most the probability that j of the zero-bits of the current search point are flipped which is
$$\binom{X}{j} \frac{1}{n^j} \leq \frac{\alpha^j}{j!} \leq \frac{\alpha^j e^j}{j^j}\,.$$

□

Using this proposition, we are able to show the following lower bound on the optimization time of the EA$_F$ on every non-negative linear function with non-zero weights.

Theorem 13. *Let $f : \{0,1\} \to \mathbb{R}_0^+$ be linear with non-zero weights. Then there exists a constant $\gamma > 1$ such that with probability $1 - 2^{-\Omega(n)}$ in all solutions produced by the EA$_F$ in the first γ^n steps at most $0.97n$ bits are set correctly.*

Proof. As discussed before, we suppose that all weights of f are strictly positive and hence that in an optimal solution of f all bits have the value one. For $t \geq 0$ let X_t be the number of zero-bits of the solution x after t iterations.

We apply Theorem 4 with parameters $a = 0.03$, $b = 0.035$, $\delta = (1 - 3b)/3e^2b > 1.15$, and $C = e^\delta$.

(*a*) Due to the Chernoff bounds $P[X_0 \geq bn] = 1 - 2^{-\Omega(n)}$, since in the initial search point one-bits and zero-bits occur with the same probability.

(*b*) Let $t \geq 0$ and X_t such that $an < X_t < bn$. Then $P[X_{t+1} - X_t = j \mid X_t] \geq \delta \cdot P[X_{t+1} - X_t = -j \mid X_t]$ by Proposition 12 with $\alpha = b$.

(*c*) Let $t \geq 0$ and X_t such that $X_t \geq bn$. Then for $\alpha = 1$ $\sum_{j \geq 1} \delta^j \cdot P[X_{t+1} - X_t = -j \mid X_t] \leq \sum_{j \geq 1} \delta^j/j! \leq C$ by Proposition 12.1.

The statement follows. □

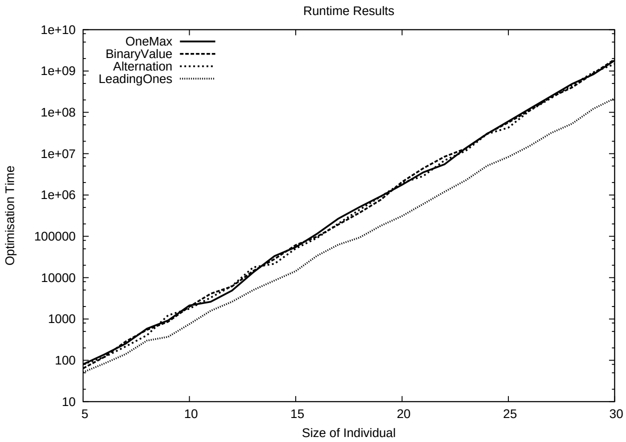

Figure 1. The expected optimization time of the EA_F on various fitness functions depending on the bit size of the individuals. Observe that we use a logarithmic scale. For every size of individuals, we conducted 50 runs.

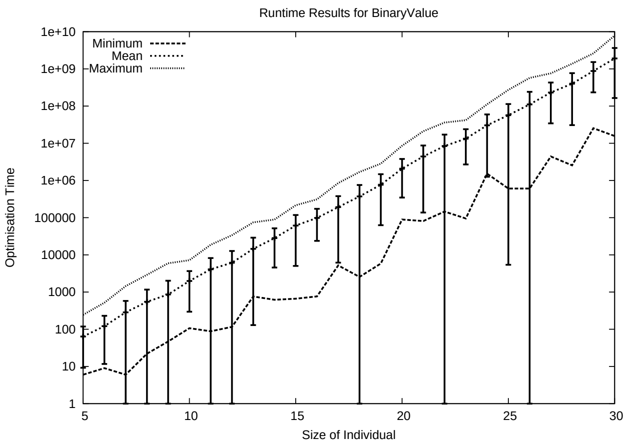

Figure 2. A closer look at the optimization time of the EA_F on BinaryValue. The minimum, maximum, and mean runtime (over 50 runs) is shown.

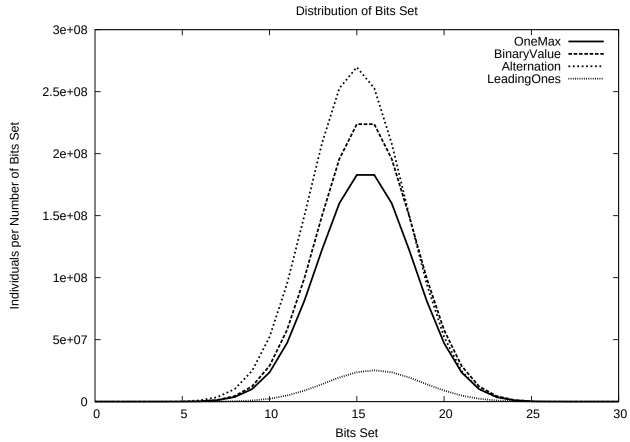

Figure 3. The frequency of individuals depending on the number of correctly set bits of the individuals during a run of the EA_F. The data was obtained by taking the mean values of 25 runs on 30-bit individuals.

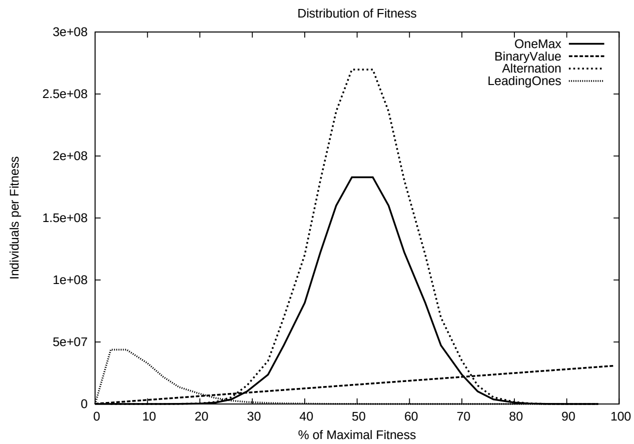

Figure 4. The frequency of individuals depending on the fitness of the individuals during a run of the EA_F. The data was obtained by taking the mean values of 25 runs on 30-bit individuals.

6. EXPERIMENTAL RESULTS

We conduct experiments to emphasize the result of this paper. For this we study the function OneMax as defined in Section 4 and the following additional three pseudo-Boolean functions.

$$\textsc{BinaryValue}(x) \ := \ \sum_{i=1}^{n} 2^i x_i\,.$$

$$\textsc{Alternation}(x) \ := \ \lceil n \rceil + \sum_{i=1}^{n} (-1)^i x_i\,.$$

$$\textsc{LeadingOnes}(x) \ := \ \sum_{i=1}^{n} \prod_{j=1}^{i} x_j\,.$$

All these functions have been studied before in theoretical as well as in practical works. The functions OneMax, BinaryValue, and Alternation are linear, thus the expected optimization time of the elitist (1+1) EA is of order $\Theta(n \log n)$, see [1]. In contrast to this, the function LeadingOnes, which counts the number of ones in the bitstring x up to the first zero, is not linear. It is well known that the elitist (1+1) EA has expected optimization time $\Theta(n^2)$ on this function [1].

We run the EA_F on the fitness functions mentioned above for individuals of sizes from 5 to 30. For each size, 50 runs are conducted. In Figure 1, the average optimization time of all four functions is shown.

Our experiments show that the standard deviation from the average is quite high. To illustrate this, Figure 2 takes a closer look at the function BinaryValue. It shows the minimum, maximum, and mean of the optimization times needed by the 50 runs for each bit-size from 5 to 30. We see that these differ by orders of magnitude. For individuals consisting of 26 bits, for example, the fastest run has an optimization time of 605,563 generations compared to 570,330,004 generations for the slowest. The standard deviation for each bit-size is depicted by the bars around the mean optimization time. That these bars seem to reach further down than up is caused by the logarithmic scale used.

Interestingly, it seems that for LeadingOnes the optimization time is lower (albeit still exponential) than for the linear functions. We can only speculate that this is caused by the fact that flipping one of the first bits to zero becomes more and more costly as the number of leading one-bits approaches the total number of bits.

Our proofs make use of the fact that individuals are likely to get stuck with about half of their bits set and cannot proceed from there. To illustrate this effect, we conduct 25 runs with 30-bit individuals for each of the four functions mentioned above. During these runs, we keep track of how many individuals with a certain number of correctly set bits are created. Figure 3 shows the results of this experiment and the distribution is indeed as expected.

We also analyze how many individuals of certain fitness are generated for the four functions. The results are shown in Figure 4. Although we know that in most individuals only about half the bits are set correctly, only for OneMax and Alternation the average individual as well has only about half the maximal possible fitness. For BinaryValue the number of individuals created for each fitness value raises linearly. This is caused by the fact that the highest bits are also the most expensive ones. Hence flipping them to zero is more unlikely than flipping a lower bit to zero. Thus, the one-bits tend to cluster in the higher part of the individual, leading to more individuals of a high fitness value. For LeadingOnes, nearly all individuals generated have low fitness, and as soon as a certain fitness is reached, it seems that the optimal individual is found quite fast. Indeed, the number of individuals having a certain fitness value falls off exponentially fast towards the end. This may again be caused by the fact that once a large number of leading ones has been generated, it is expensive to accept solutions with a small number of leading ones. The more leading ones exist, the higher the price of accepting such solutions.

*

7. CONCLUSIONS

We have carried out rigorous runtime analyses for using fitness proportional selection in evolutionary algorithms. Our results point out that switching from elitist selection to fitness proportional selection increases the runtime of simple evolutionary algorithms drastically on all linear pseudo-Boolean functions with a linear number of non-zero weights. Our experimental investigations for some popular functions complement our asymptotic results and show that the proven behavior may also be observed for small instance sizes.

Due to the negative results for the use of fitness proportional selection presented in this paper, the question arises whether larger populations can help to overcome the drawbacks pointed out. Often it is argued that fitness proportional selection helps in the optimization process as it ensures a diverse population. We think it would be a challenging and interesting step to analyze population-based EAs using fitness proportional selection or other non-elitist selection strategies in the future.

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] W. Feller. *An Introduction to Probability Theory and Its Applications*, volume 1. Wiley, 3rd edition, 1968.

[3] O. Giel and P. K. Lehre. On the effect of populations in evolutionary multi-objective optimization. In *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 651–658, New York, NY, USA, 2006. ACM.

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

[5] D. E. Goldberg. *Genetic Algorithms in Search Optimization and Machine Learning*. Addison-Wesley, 1989.

[6] G. R. Harik, E. Cantú-Paz, D. E. Goldberg, and B. L. Miller. The gambler's ruin problem, genetic algorithms, and the sizing of populations. *Evolutionary Computation*, 7(3):231–253, 1999.

[7] J. He and X. Yao. Drift analysis and average time complexity of evolutionary algorithms. *Artificial Intelligence*, 127:57–85, 2001.

[8] J. He and X. Yao. Towards an analytic framework for analysing the computation time of evolutionary algorithms. *Artif. Intell.*, 145(1-2):59–97, 2003.

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

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

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

[12] F. Neumann and C. Witt. Ant colony optimization and the minimum spanning tree problem. In *Electronic Colloquium on Computational Complexity (ECCC)*, 2006. Report No. 143.

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

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

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