

Tight Analysis of the $(\mathbf{1} + \mathbf{1})$ -EA for the Single Source Shortest Path Problem

Benjamin Doerr Edda Happ
Christian Klein

Max Planck Institute for Computer Science,
Campus E1 4
66123 Saarbrücken, Germany

Abstract

We conduct a rigorous analysis of the $(1+1)$ evolutionary algorithm for the single source shortest path problem proposed by Scharnow, Tinnefeld and Wegener (Journal of Mathematical Modelling and Algorithms, 2004). We prove that with high probability the optimization time is $O(n^2 \max\{\ell, \log(n)\})$, where ℓ is the smallest integer such that any vertex can be reached from the source via a shortest path having at most ℓ edges.

This bound is tight. For all values of n and ℓ we provide a graph with edge weights such that, with high probability, the optimization time is of order $\Omega(n^2 \max\{\ell, \log(n)\})$.

To obtain such sharp bounds, we develop a new technique that overcomes the coupon collector behavior of previously used arguments. Also, we exhibit a simple Chernoff type inequality for sums of independent geometrically distributed random variables, and one for sequences of random variables that are not independent, but show a desired behavior independent of the outcomes of the previous random variables. We are optimistic that these tools find further applications in the analysis of evolutionary algorithms.

1 Introduction

Rigorous run-time analyses of evolutionary algorithms for classical algorithmic problems became a hot topic in the last years. The aim is to obtain a theoretically founded understanding of the basic principles of evolutionary

computation. Currently, we are in the situation that we have seen many highly successful applications of evolutionary algorithms, but hardly can explain why they are so successful in practice. Such an understanding, however, would be very helpful in the future design of such algorithms.

Theoretical run-time analyses started on artificial problems like maximizing simple pseudo-boolean functions $f : \{0, 1\}^n \rightarrow \mathbb{R}$, e.g., the number of ones ($\text{ONEMAX}(x) := \sum_{i=1}^n x_i$), the number of leading ones ($\text{LO}(x) := \max\{i \in \mathbb{N}_0 \mid \forall j \leq i : x_j = 1\}$) or monotone linear functions and polynomials [Weg01, DJW02, WW05]. Regarding such artificial problems already lead to some insight on how evolutionary algorithms work and on how to analyze them.

More recently, the focus moved on to classical problems from computer science. In one of the first papers in this direction, [STW04] proposed and analyzed evolutionary algorithms for sorting and shortest path problems. Other results on evolutionary algorithms for combinatorial optimization problems include papers on the problem of computing Eulerian cycles [Neu04, DHN06, DKS07, DJ07], minimum spanning trees [NW04, NW05], maximal matchings [GW03] and partitions [Wit05].

For such problems, the design and analysis of evolutionary algorithms becomes more interesting. One simple reason is that we typically have the choice between several natural representations of individuals, fitness functions and variation operators.

To avoid misunderstandings, let us stress that the focus of this line of research is not to find superior algorithms for the particular underlying optimization problem. Since these are classical and important problems, they have been investigated thoroughly and hence very good custom-tailored algorithms already exist. The focus rather is to analyze how such problems can be tackled with generic approaches, to understand how their components like particular representations or variation operators work, and, finally, to develop methods to analyze evolutionary algorithms. We refer to the recent books [NW10, AD11] for more information on theoretical analyses of bio-inspired randomized search heuristics.

Surprisingly, even for extremely simple evolutionary algorithms on simple problems a tight analysis of their run-time behavior can be very complicated. Note that we only aim at determining its order of magnitude, that is, we ignore constant factors and lower order terms. Also, we only regard the optimization time, that is, the number of fitness evaluations needed to find the desired solution.

A classical example for the difficulties one faces when analyzing evolutionary algorithms is the $O(n \log(n))$ bound for the optimization time of a simple $(1 + 1)$ evolutionary algorithm maximizing a monotone linear func-

tion on $\{0, 1\}^n$. Here, classical methods from the analysis of randomized algorithms lead to a highly technical proof [DJW02]. Subsequent efforts put into this problem resulted in the so-called drift analysis becoming a major tool in the run-time analysis of evolutionary algorithms (see [HY04] for the first use and [Jäg08, DJW10, DG10] for recent simplifications).

Naturally, analyses are even harder for problems that carry a richer structure. Hence it is not surprising that the analysis of the single source shortest path evolutionary algorithm [STW04], the first paper performing rigorous run-time analyses for combinatorial optimization problems, is tight only for certain instances. As we shall see in this work, for many graphs the optimization time needed to solve the single source shortest path problem is better than what is proven in [STW04].

[STW04] proposed a natural $(1 + 1)$ evolutionary algorithm for the problem of finding shortest paths from a single node (“source”) to all other vertices in a graph with edge weights (see below for a precise definition of the problem). They show an upper bound of $O(n^3)$ for the expected optimization time on n -vertex graphs. This bound is tight if (and only if, as we shall see) the graph and edge weights are such that there is a vertex such that all shortest paths to the source contain $\Omega(n)$ edges.

The proof given by [STW04] reveals an in fact stronger upper bound of $O(n^2 \sum_{i=1}^n \log(n_i + 1))$, where n_i is the number of vertices for which a shortest path to the source with the minimum number of edges consists of exactly i edges. Since $\sum_{i=1}^n (n_i + 1) = 2n$, this yields a bound of $O(n^2 \ell \log(\frac{2n}{\ell}))$, where ℓ is the smallest integer such that any vertex can be reached from the source via a shortest path having at most ℓ edges.

In this work, we give a tight analysis of the proposed algorithm. This leads to an improved upper bound for the expected optimization time of $O(n^2 \max\{\ell, \log(n)\})$. In addition, we show that this bound not only holds in expectation, but is fulfilled with high probability. That is, for any constant $c > 0$ the implicit constants in the optimization time bound can be chosen such that after that many iterations, the optimum is found with probability at least $1 - n^{-c}$. The bound on the optimization time is tight for all ℓ . For all values of ℓ we present a problem instance such that all shortest paths have length at most ℓ , but the optimization time is $\Omega(n^2 \max\{\ell, \log(n)\})$ with high probability.

To prove strong bounds like this, we develop several tools that might see further applications in the future. To prove the upper bound, we closely analyze how nodes become connected to the source via shortest paths. The speed of growth of such shortest paths (note that they do not have to be unique) displays a strong concentration behavior. Although we use a union bound argument over all paths needed, it is still strong enough to obtain bounds

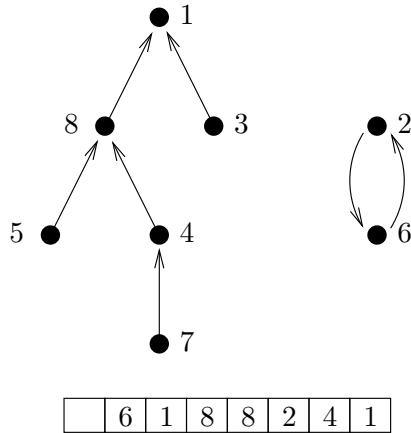


Figure 1: The graph representation and the vector representation of an individual not representing a tree. The fitness of vertex 7 is $d_{18} + d_{84} + d_{47}$, as represented by the arrows below the vector representation. The fitness of vertex 2 is ∞ , since there is no path from vertex 2 to vertex 1, as shown by the arrows above the vector representation.

that hold with high probability. To show the lower bound, we prove a Chernoff type strong concentration bound for sums of geometrically distributed random variables. To the best of our knowledge, such bounds have not been published so far in a mathematics or computer science journal. For both the upper and the lower bound, we encounter a situation where a sequence of random variables is not independent, but has the property that each member of the sequence has a desired property for all possible outcomes of the previous random variables. For such situations, we prove Chernoff type concentration bounds. We are optimistic that all these tools see more applications in the near future.

2 A (1+1)-EA for the Single Source Shortest Path Problem

Let $G = (V, E)$, $V = [1..n] := \{1, \dots, n\}$, $E \subseteq V^2$ be a directed graph with edge weights $w : E \rightarrow \mathbb{N}$. Given a vertex $s \in V$ called “source”, the single source shortest path problem (SSSP) is the problem of finding a shortest path from s to all other vertices $v \in V$. A simple implementation of Dijkstra’s famous algorithm [Dij59] solves the problem in time $O(n^2)$.

We can easily restrict ourselves to the complete (bi-directed) graph K_n by allowing the weight $w(e) = \infty$ for previously not existing edges e . A

problem instance then is given by the distance matrix $D = (d_{ij})_{1 \leq i, j \leq n}$ of the graph, where $d_{ij} = w((i, j)) \in \mathbb{N} \cup \{\infty\}$. Note that the algorithm described below also works in the case of undirected graphs. For each undirected edge $e = \{i, j\}$ simply set $d_{ij} = d_{ji} = w(e)$.

It is easy to see that we can choose shortest paths from s to any other vertex in such a way that the union of these paths forms a tree. Hence we may represent solutions to the SSSP problem by giving for each vertex $i \in V$ its predecessor $p(i)$ on a shortest path from s to i .

In this paper, we analyze the $(1+1)$ evolutionary algorithm $((1+1)$ -EA) for the SSSP problem introduced by [STW04]. In the remainder of this section, we describe this algorithm assuming $s = 1$.

We represent the candidate solutions as vectors of predecessors $I = (p(2), \dots, p(n)) \in \{1, \dots, n\}^{n-1}$. Note that this representation does not imply that an individual forms a tree. See Figure 1 for an example.

We use a multi-criteria fitness function. For an individual I , it is defined as $f(I) := (f_2(I), \dots, f_n(I))$ with

$$f_i(I) := \begin{cases} \infty & \text{if } I \text{ does not connect } s \text{ to } i, \\ w(P(s, i)) & \text{otherwise.} \end{cases}$$

Here, $w(P(s, i))$ is the cost of the path P from s to i defined by I . If this path is $P = (s = v_1, v_2, \dots, v_j = i)$ for $v_1, \dots, v_j \in V$ then $w(P(s, i)) = d_{v_1 v_2} + \dots + d_{v_{j-1} v_j}$. See again Figure 1 for an example.

At the beginning, the $(1+1)$ -EA generates an initial individual I by assigning to each vertex $v \in V \setminus \{s\}$ a predecessor $p(v) \in V \setminus \{v\}$ uniformly at random. In the following mutation step, I is modified to generate a new individual I' . Then, a selection step is done replacing the individual I by I' if the fitness of I' is not worse than the one of I . Mutation and selection are repeated as long as desired.

On bit-strings of length n , the mutation step of the classical $(1+1)$ -EA independently flips each bit with probability $\frac{1}{n}$. As this does not make sense for more complex representations like the one we use here, this behavior must be simulated suitably. The number of elementary mutations (“bit flips”) applied in the classical case can be approximated using a Poisson distribution $\text{Pois}(\lambda = 1)$ with parameter $\lambda = 1$. The Poisson distribution with $\lambda = 1$ is the limit of the Binomial distribution for n trials with probability $\frac{1}{n}$ each. For general $\lambda > 0$ and $k \in \mathbb{N}$, the Poisson distribution with parameter λ is given by the probability mass function

$$f(k; \lambda) = \frac{\lambda^k e^{-\lambda}}{k!},$$

```

(1 + 1)-EA FOR SSSP
  Initialization:
1   $I \leftarrow (p(2), \dots, p(n)),$ 
    $p(i) \in V \setminus \{i\}$  chosen u. a. r.
2  repeat
   Mutation:
3   Pick  $S$  according to  $\text{Pois}(\lambda = 1)$ 
4    $I^0 \leftarrow I$ 
5   for  $k = 1$  to  $S + 1$ 
6     do
7       Choose  $i \in [2..n]$  u. a. r.
8       Choose  $j \in [1..n] \setminus \{i\}$  u. a. r.
9       Generate  $I^k$  from  $I^{k-1}$  by setting  $p(i)$  to  $j$ .
   Selection:
10  if  $f_i(I^{S+1}) \leq f_i(I)$  for all  $i \in [2..n]$ 
11    then  $I \leftarrow I^{S+1}$ 
12 forever

```

Figure 2: The (1 + 1)-EA for the SSSP problem.

which gives the probability that a $\text{Pois}(\lambda)$ -distributed random variable is equal to k . The number of elementary mutations applied in one step is then given as $S + 1$, where S is distributed according to $\text{Pois}(\lambda = 1)$.

An elementary mutation of the vector I consists of randomly choosing a vertex v with $v \in V \setminus \{s\}$ and setting its predecessor $p(v)$ to a vertex chosen uniformly at random from $V \setminus \{v\}$. Obviously, there are $(n - 1)^2$ possible ways to choose a vertex and its predecessor and thus to apply an elementary mutation to an individual I .

The selection step accepts I' if $f(I') \leq f(I)$, which is the case if $f_i(I') \leq f_i(I)$ for all $2 \leq i \leq n$. Therefore, once we have found an optimal path for a vertex v , the (1 + 1)-EA does not accept mutations that would cause s to be connected to v using a suboptimal path.

The pseudo-code for the (1 + 1)-EA solving the SSSP problem is given in Figure 2.

3 The Upper Bound

In this section we show that the optimization time of the $(1 + 1)$ -EA for the SSSP problem is $O(n^2 \max\{\ell, \log(n)\})$ with high probability. Here, ℓ is the edge radius of s defined as follows.

Definition 1 (Edge radius). *Let G be a weighted graph and s a vertex of G . Then the edge radius $\ell_G(s)$ of s in G is the minimum number r such that for all vertices $v \in V \setminus \{s\}$, there is a shortest path from s to v having at most r edges. In other words,*

$$\ell_G(s) := \max_{v \in V} \min\{\ell(P) \mid P \text{ is a shortest path from } s \text{ to } v\},$$

where $\ell(P)$ is the number of edges of the path P .

“With high probability” means that an event happens with probability at least $1 - n^{-c}$, where c is an arbitrary constant. The “optimization time” of an evolutionary algorithm is the number of evaluations of the fitness function the algorithm executes until it finds an optimal solution. This is the usual complexity measure used when analyzing evolutionary algorithms, whereas the time needed for the creation of new individuals by mutation or crossover and for the execution of the fitness function is usually disregarded.

The key observation in the proof is that the actual speed of growth of the shortest path tree deviates only little from the expected speed. This phenomenon called “strong concentration” can be quantized through so-called Chernoff bounds if the random variable of interest can be written as sum of independent random variables. We shall use the following version from [AS08].

Theorem 2 (Chernoff bound, lower tail). *Let X_1, \dots, X_t be mutually independent random variables with $\Pr[X_i = 1] = p$ and $\Pr[X_i = 0] = 1 - p$ for all i . Let $X := \sum_{i=1}^t X_i$. Then for all $\delta \in (0, 1]$,*

$$\Pr[X < (1 - \delta)pn] < \exp\left(-\frac{\delta^2 pn}{2}\right).$$

In our proofs, however, we will also encounter sums of moderately correlated variables. They will form a sequence of random variables such that each member has certain properties no matter what is the outcome of previous ones. The following simple, though not completely trivial lemma shows how to apply Chernoff bounds also in this setting.

Lemma 3 (Chernoff bound, lower tail, moderate independence). *Let X_1, \dots, X_t be arbitrary binary random variables. Let X_1^*, \dots, X_t^* be binary random variables that are mutually independent and such that for all i , X_i^* is independent of X_1, \dots, X_{i-1} . Assume that for all i and all $x_1, \dots, x_{i-1} \in \{0, 1\}$,*

$$\Pr[X_i = 1 | X_1 = x_1, \dots, X_{i-1} = x_{i-1}] \geq \Pr[X_i^* = 1].$$

Then for all $k \geq 0$, we have

$$\Pr \left[\sum_{i=1}^t X_i < k \right] \leq \Pr \left[\sum_{i=1}^t X_i^* < k \right],$$

and the latter term can be bounded by Chernoff bounds for lower tails.

Proof. For $j \in [0..t]$, let $P_j := \Pr[\sum_{i=1}^j X_i + \sum_{i=j+1}^t X_i^* \geq k]$. Let $\mathcal{X}_k^t :=$

$\{(x_1, \dots, x_t) \in \{0, 1\}^t \mid \sum_{i=1}^t x_i = k\}$. Then for all $j \in [0..t-1]$, we compute

$$\begin{aligned}
P_{j+1} &= \Pr \left[\sum_{i=1}^{j+1} X_i + \sum_{i=j+2}^t X_i^* \geq k \right] \\
&= \Pr \left[\sum_{i=1}^j X_i + \sum_{i=j+2}^t X_i^* \geq k \right] + \\
&\quad \sum_{(x_1, \dots, x_j, x_{j+2}, \dots, x_t) \in \mathcal{X}_{k-1}^{t-1}} \Pr[X_1 = x_1, \dots, X_j = x_j, X_{j+1} = 1] \prod_{i=j+2}^t \Pr[X_i^* = x_i] \\
&= \Pr \left[\sum_{i=1}^j X_i + \sum_{i=j+2}^t X_i^* \geq k \right] + \\
&\quad \sum_{(x_1, \dots, x_j, x_{j+2}, \dots, x_t) \in \mathcal{X}_{k-1}^{t-1}} \left(\Pr[X_1 = x_1, \dots, X_j = x_j] \right. \\
&\quad \left. \Pr[X_{j+1} = 1 \mid X_1 = x_1, \dots, X_j = x_j] \prod_{i=j+2}^t \Pr[X_i^* = x_i] \right) \\
&\geq \Pr \left[\sum_{i=1}^j X_i + \sum_{i=j+2}^t X_i^* \geq k \right] + \\
&\quad \sum_{(x_1, \dots, x_j, x_{j+2}, \dots, x_t) \in \mathcal{X}_{k-1}^{t-1}} \left(\Pr[X_1 = x_1, \dots, X_j = x_j] \right. \\
&\quad \left. \Pr[X_{j+1}^* = 1] \prod_{i=j+2}^t \Pr[X_i^* = x_i] \right) \\
&= \Pr \left[\sum_{i=1}^j X_i + \sum_{i=j+2}^t X_i^* \geq k \right] + \Pr \left[\sum_{i=1}^j X_i + \sum_{i=j+2}^t X_i^* = k-1 \wedge X_{j+1}^* = 1 \right] \\
&= \Pr \left[\sum_{i=1}^j X_i + \sum_{i=j+1}^t X_i^* \geq k \right] \\
&= P_j.
\end{aligned}$$

Thus, we have $\Pr[\sum_{i=1}^t X_i \geq k] = P_t \geq P_{t-1} \geq \dots \geq P_1 \geq P_0 = \Pr[\sum_{i=1}^t X_i^* \geq k]$, and the claim follows. \square

We are now ready to prove the upper bound.

Theorem 4. *The optimization time of the $(1 + 1)$ -EA is $O(n^2 \max\{\ell, \log(n)\})$ with high probability (meaning with probability at least $1 - n^{-c}$ for any arbitrary constant c).*

This follows immediately from the following lemma.

Lemma 5. *Let $\ell := \ell_G(s)$ be the edge radius of s in G , $\ell^* = \max\{\ell, \log(n)\}$, $c > 0$ be a constant, $\eta \geq 8(c + 1) > 8$ and $t = e\eta(n - 1)^2\ell^*$. Then the optimization time of the $(1 + 1)$ -EA is less than t with probability at least $1 - n^{-c}$.*

Proof. Let $v \in V \setminus \{s\}$. We first analyze the number of iterations needed to find a shortest path from s to v . Let $P := (v_1, \dots, v_{\ell'+1})$ be a shortest path from $s = v_1$ to $v = v_{\ell'+1}$. By definition of the edge radius ℓ , we can pick P so that it has $\ell' \leq \ell$ edges.

For an individual I , let

$$\ell(P, I) := \max\{i - 1 \mid \text{there is a shortest path from } s \text{ to } v_i \text{ in } I\}.$$

We call a mutation step an *improvement* in P if it increases $\ell(P, I)$. Note that $\ell(P, I)$ never decreases due to the multi-criteria formulation of the fitness function.

Let t' be the number of steps the $(1 + 1)$ -EA needs to reach an $\ell(P, I)$ value of ℓ' , the maximum possible value, indicating that a shortest path from s to v has been found. Define the random variables X_i for $1 \leq i \leq t'$ by $X_i := 1$ if the i -th mutation step consists just of a single elementary mutation (happens with probability $\frac{1}{e}$), and this is an improvement in P , and $X_i := 0$ otherwise.

Then regardless of the values of X_j , $j < i$, we have $\Pr[X_i = 1] \geq p := \frac{1}{e(n-1)^2}$, since an improvement in particular is obtained if the i -th mutation step picks the vertex $v_{\ell(P,I)+1}$ as base vertex (happens with probability $\frac{1}{n-1}$) and resets its predecessor pointer to $v_{\ell(P,I)}$ (which again happens with probability $\frac{1}{n-1}$).

For $i > t'$ define the random variables X_i by $\Pr[X_i = 1] := p$ and $\Pr[X_i = 0] := 1 - p$ independently from all other random variables.

Let $X := \sum_{i=1}^t X_i$. Note that if no path from s to v has been found within t iterations, then $X < \ell'$.

Let X_i^* for $i \in [1..t]$ be mutually independent random binary variables with $\Pr[X_i^* = 1] = p$ and $\Pr[X_i^* = 0] = 1 - p$. Let $X^* := \sum_{i=1}^t X_i^*$. Then by construction, $\Pr[X_i = 1 \mid X_1 = x_1, \dots, X_{i-1} = x_{i-1}] \geq \Pr[X_i^* = 1]$ for all i and all $x_1, \dots, x_{i-1} \in \{0, 1\}$.

By applying Lemma 3, the probability of not finding a shortest path from s to v in t iterations is at most

$$\begin{aligned} \Pr \left[\begin{array}{l} \text{no shortest path from} \\ s \text{ to } v \text{ found in time } t \end{array} \right] &\leq \Pr[X < \ell'] \\ &\leq \Pr[X < \ell^*] \\ &\leq \Pr[X^* < \ell^*]. \end{aligned}$$

For $\delta := \frac{(\eta-1)}{\eta}$ we have

$$(1 - \delta)\mathbb{E}[X^*] = \left(1 - \frac{(\eta-1)}{\eta}\right) \frac{1}{e(n-1)^2} e\eta(n-1)^2 \ell^* = \ell^*.$$

Hence by Theorem 2, we can bound the probability of not finding a shortest path from s to v in time $t = e\eta(n-1)^2 \ell^*$ by

$$\begin{aligned} \Pr[X^* < \ell^*] &= \Pr[X^* < (1 - \delta)\mathbb{E}[X^*]] \\ &\leq \exp\left(-\frac{\delta^2 \mathbb{E}[X^*]}{2}\right) \\ &\leq \exp\left(-\frac{(\eta-1)^2 \ell^*}{2\eta}\right) \\ &\leq \exp\left(-\frac{(\frac{\eta}{2})^2 \ell^*}{2\eta}\right) \\ &= \exp\left(-\frac{\eta}{8} \ell^*\right), \end{aligned}$$

valid for $\eta \geq 2$. Using a union bound argument, we get

$$\begin{aligned} \Pr \left[\begin{array}{l} \text{not for all } v \in V \setminus \{s\} \text{ a shor-} \\ \text{test path from } s \text{ to} \\ v \text{ found in time } t \end{array} \right] &\leq \sum_{v \in V \setminus \{s\}} \Pr \left[\begin{array}{l} \text{no shortest path} \\ \text{from } s \text{ to } v \\ \text{found in time } t \end{array} \right] \\ &\leq n \exp\left(-\frac{\eta}{8} \ell^*\right) \\ &\leq n \exp\left(-\frac{\eta}{8} \log(n)\right) \\ &\leq n^{1-\frac{\eta}{8}} \\ &\leq n^{-c}. \end{aligned}$$

□

Note that we did not optimize the value of η .

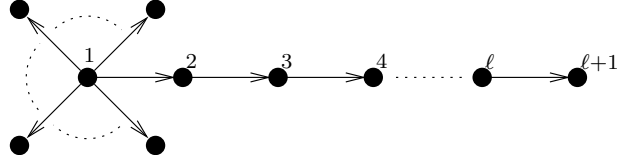


Figure 3: The SSSP tree of $G_{n,\ell}$ with source $s = 1$.

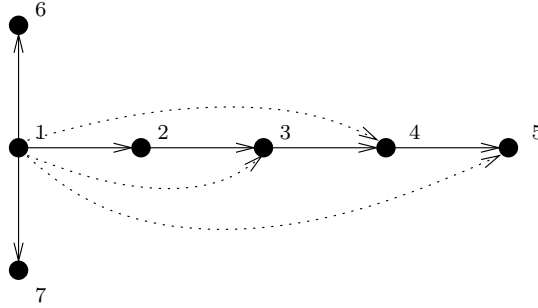


Figure 4: The weighted graph $G_{7,4}$. The solid edges have weight one and form the shortest path tree. All other edges have weight 14. The dotted edges are the second shortest paths from 1 to i for $i \in \{3, 4, 5\}$.

4 The Lower Bound

In this section we show a lower bound matching the upper bound presented in the previous section. More precisely, for any $n \in \mathbb{N}$ and $\ell \in [1..n - 1]$ we define a weighted graph $G_{n,\ell}$ on n vertices such that for a suitable vertex s the edge radius is $\ell_{G_{n,\ell}}(s) = \ell$ and the algorithm has an optimization time of at least $\Omega(n^2 \max\{\ell, \log(n)\})$ with high probability.

4.1 The Graph Class $G_{n,\ell}$

Let $n \in \mathbb{N}$ and $V = [1..n]$. For all $\ell \in [1..n - 1]$ we define the weighted graph $G_{n,\ell} = (V, E)$ such that the source of the SSSP tree to be computed is $s = 1$ and $\ell_{G_{n,\ell}}(s) = \ell$. We show that the optimization time of the $(1 + 1)$ -EA is $\Omega(n^2 \max\{\ell, \log(n)\})$ with high probability.

We will set the weights in such a way that $(1, 2, \dots, \ell, \ell + 1)$ is the unique shortest path from $s = 1$ to $\ell + 1$. For all other vertices k with $k > \ell + 1$, the edge (s, k) shall be the unique shortest path from s to k . Figure 3 shows the SSSP tree of $G_{n,\ell}$. For simplicity, we assign the weight 1 to all edges in the SSSP tree.

To guarantee that the optimization time depends linearly on ℓ , the edges

on the shortest path should be added one by one. To this purpose, each vertex $i \in [2..\ell + 1]$ should be connected to vertex s with a sufficiently cheap edge ensuring that, as long as vertex $i - 1$ is not correctly connected, it is cheaper to connect s and i directly than to connect s to i via $i - 1$. These requirements are fulfilled by $G_{n,\ell} := ([1..n], \{(i, j) | u, v \in [1..n], u \neq v\})$ with edge weights

$$w(i, j) := \begin{cases} 1, & \text{if } j = i + 1 \leq \ell + 1, \\ 1, & \text{if } i = 1 \wedge j > \ell + 1, \\ 2n, & \text{otherwise.} \end{cases}$$

The graph $G_{7,4}$ is shown in Figure 4. Note that $G_{n,1}$ is the graph with edge weight 1 for each edge $(s, i), i \in [2..n]$ and $2n$ for all other edges.

4.2 Proving a Lower Bound

We now give a lower bound on the number of steps needed by the $(1 + 1)$ -EA depending on n and ℓ . To prove that $\Omega(n^2 \max\{\ell, \log(n)\})$ is a lower bound on the optimization time of the $(1 + 1)$ -EA, we first prove that $\Omega(n^2 \log(n))$ is a lower bound on the optimization time for *all* graphs that have a unique SSSP tree.

Lemma 6. *Let $G = (V, E)$ be a graph on n vertices and $s \in V$ a vertex such that the SSSP tree of G with source s is unique. Then the optimization time of the $(1 + 1)$ -EA is $\Omega(n^2 \log(n))$ with high probability.*

For the proof, we need the following elementary lemma on the Poisson distribution.

Lemma 7. *The expected number of elementary mutations done in k iterations is exactly $2k$. The probability that more than $3k$ are done, is less than $(e/4)^k$.*

Proof. By the definition of the $(1 + 1)$ -EA, the number of elementary mutations done in a single iteration is distributed as $X + 1$, where X has distribution $\text{Pois}(1)$. By elementary properties of the Poisson distribution, we see that the number of elementary mutations done in k iterations, has distribution $\text{Pois}(k) + k$. In consequence, its expectation is $2k$ and the probability that the Poisson part exceeds its expectation by a factor of two, is at most $(e/4)^k$, see, e.g., Theorem A.1.15 of [AS08]. \square

Proof of Lemma 6. Since the shortest path tree is unique, the $(1 + 1)$ -EA is done if and only if for each vertex $v \in V \setminus \{s\}$, the predecessor pointer $p(v)$ is set to the predecessor in the unique shortest path tree. We reformulate this process as a coupon collector process.

In this proof, we say that v is *fine* after iteration t , if $p(v)$ points to the desired predecessor now, or if did so earlier. Note that if the predecessor is not already connected to s via a shortest path, v might change its predecessor to a different vertex than the one given by the shortest path tree. Clearly, the time until all vertices are fine is a lower bound for the optimization time.

After the initialization, each vertex is fine with probability exactly $\frac{1}{n-1}$. In consequence, with probability $1 - \exp(-\Theta(n))$, less than $n/2$ vertices are fine. We call the remaining $k \geq n/2$ vertices *interesting*.

Assume that, after initialization, we run the $(1+1)$ -EA for $T = \frac{1}{12}(n-1)^2 \ln(n/2)$ iterations. By Lemma 7, we see that within this number of iterations, we do at most $3T$ elementary mutations (apart from a super-exponentially small failure probability).

The probability that an elementary mutation chooses one of the interesting vertices and sets its predecessor to the desired one, is $(k/(n-1))(1/(n-1))$. In consequence, with probability $1 - \exp(-\Theta(n \log n))$, a sequence of $3T$ elementary mutations produces at most $6Ti/(n-1)^2$ such good events. Viewing each of these good events as the action of buying one out of i different coupons, the coupon collector theorem tells us that with probability $1 - \exp(-\Theta(n))$, these $6Ti/(n-1)^2 \leq \frac{1}{2}i \ln(i)$ trials do not suffice to obtain all i coupons. □

The above lemma shows our lower bound for $\ell = O(\log(n))$. To complete our claim, we have to prove that for larger ℓ the optimization time linearly depends on ℓ . To this aim, we need a number of tools, including a Chernoff type inequality for geometrically distributed random variables that seems to be new.

Theorem 8. (*Chernoff bounds, upper tail*) *Let $X_i, i \in [1..n]$, be independent random variables, $X := \sum_{i=1}^n X_i$, and $0 < p < 1$ and $\delta > 0$ be constants. Then the following holds.*

a) *If $\Pr[X_i = 1] = p$ and $\Pr[X_i = 0] = 1 - p$ for all $i \in [1..n]$, then*

$$\Pr[X \geq (1 + \delta)pn] \leq \exp\left(-\frac{\min\{\delta, \delta^2\}pn}{3}\right).$$

b) *If the X_i are geometrically distributed random variables with $\Pr[X_i = j] = (1 - p)^{j-1}p$ for all $j \in \mathbb{N}$, then*

$$\Pr[X \geq (1 + \delta)\mathbb{E}[X]] \leq \exp\left(-\frac{\delta^2}{2} \frac{(n-1)}{1+\delta}\right).$$

Proof. Part a) is a classical Chernoff bound, cf. [AS08]. To prove part b), let Y_1, Y_2, \dots be an infinite sequence of independent, identically distributed biased coin tosses (binary random variables) such that Y_i is one with probability $\Pr[Y_i = 1] = p$ and zero with probability $\Pr[Y_i = 0] = 1 - p$. Note that the random variable “smallest j such that $Y_j = 1$ ” has the same distribution as each X_i . In consequence, X has the same distribution as “smallest j such that exactly n of the variables Y_1, \dots, Y_j are one”. In other words, $\Pr(X \geq j) = \Pr(\sum_{i=1}^{j-1} Y_i \leq n - 1)$ for all $j \in \mathbb{N}$. This manipulation reduces our problem to the analysis of independent Bernoulli trials and enables us to use classical Chernoff bounds.

The expected value of each X_i is $\mathbb{E}[X_i] = \frac{1}{p}$, thus $\mathbb{E}[X] = \frac{n}{p}$. Let $Y := \sum_{i=1}^{\lceil (1+\delta)\mathbb{E}[X]-1 \rceil} Y_i$. By the above,

$$\Pr[X \geq (1 + \delta)\mathbb{E}[X]] = \Pr[Y \leq n - 1].$$

The expected value of Y is bounded by

$$\mathbb{E}[Y] = \lceil (1 + \delta)\mathbb{E}[X] - 1 \rceil p \geq (1 + \delta)n - p > (1 + \delta)(n - 1).$$

Now let $\delta' := 1 - \frac{n-1}{\mathbb{E}[Y]}$. Then $0 < \delta' \leq 1$ and $\Pr[Y \leq n - 1] = \Pr[Y \leq (1 - \delta')\mathbb{E}[Y]]$. Hence we can apply Theorem 2 to get

$$\begin{aligned} \Pr[X \geq (1 + \delta)\mathbb{E}[X]] &= \Pr[Y \leq (1 - \delta')\mathbb{E}[Y]] \\ &\leq \exp\left(-\frac{1}{2}\mathbb{E}[Y]\left(1 - \frac{n-1}{\mathbb{E}[Y]}\right)^2\right) \\ &\leq \exp\left(-\frac{1}{2}\mathbb{E}[Y]\left(1 - \frac{1}{1+\delta}\right)^2\right) \\ &\leq \exp\left(-\frac{1}{2}(n-1)(1+\delta)\left(\frac{\delta}{1+\delta}\right)^2\right). \end{aligned}$$

□

The following lemma is an analogue of Lemma 3, but now for upper tails. In addition, we allow the variables to take as values arbitrary non-negative integers.

Lemma 9 (Chernoff bound, upper tail, moderate independence). *Let X_1, \dots, X_t be arbitrary random variables taking non-negative integers as values. Let X_1^*, \dots, X_t^* be non-negative random variables that are mutually independent and such that for all i , X_i^* is independent of X_1, \dots, X_{i-1} . Assume that for all i and all x_1, \dots, x_{i-1} , we have for all $m > 0$,*

$$\Pr[X_i = m | X_1 = x_1, \dots, X_{i-1} = x_{i-1}] \leq \Pr[X_i^* = m].$$

Then for all $k \geq 0$, we have

$$\Pr\left[\sum_{i=1}^t X_i \geq k\right] \leq \Pr\left[\sum_{i=1}^t X_i^* \geq k\right].$$

Proof. Since $\Pr[X_i = m | X_1 = x_1, \dots, X_{i-1} = x_{i-1}] \leq \Pr[X_i^* = m]$ for all x_1, \dots, x_{i-1} and $m > 0$, it follows that

$$\Pr[X_i \geq m | X_1 = x_1, \dots, X_{i-1} = x_{i-1}] \leq \Pr[X_i^* \geq m]$$

holds for all non-negative m . Define $P_j := \Pr[\sum_{i=1}^j X_i + \sum_{i=j+1}^t X_i^* \geq k]$ for $j \in [0..t]$ and $\mathcal{X}_k^t := \{(x_1, \dots, x_t) \in \mathbb{Z}_{\geq 0}^t \mid \sum_{i=1}^t x_i = k\}$. Then we get

$$\begin{aligned} P_{j+1} &= \Pr \left[\sum_{i=1}^{j+1} X_i + \sum_{i=j+2}^t X_i^* \geq k \right] \\ &= \Pr \left[\sum_{i=1}^j X_i + \sum_{i=j+2}^t X_i^* \geq k \right] + \\ &\quad \sum_{m=1}^k \left(\Pr \left[\sum_{i=1}^j X_i + \sum_{i=j+2}^t X_i^* = k - m \right] \cdot \Pr[X_{j+1} \geq m] \right) \\ &= \Pr \left[\sum_{i=1}^j X_i + \sum_{i=j+2}^t X_i^* \geq k \right] + \\ &\quad \sum_{m=1}^k \sum_{(x_1, \dots, x_j, x_{j+2}, \dots, x_t) \in \mathcal{X}_{k-m}^{t-1}} \Pr [X_1 = x_1, \dots, X_j = x_j] \cdot \\ &\quad \Pr [X_{j+1} \geq m | X_1 = x_1, \dots, X_j = x_j] \cdot \prod_{i=j+2}^t \Pr [X_i^* = x_i] \\ &\leq \Pr \left[\sum_{i=1}^j X_i + \sum_{i=j+2}^t X_i^* \geq k \right] \\ &\quad + \sum_{m=1}^k \Pr \left[\sum_{i=1}^j X_i + \sum_{i=j+2}^t X_i^* = k - m \right] \cdot \Pr [X_{j+1}^* \geq m] \\ &= \Pr \left[\sum_{i=1}^j X_i + \sum_{i=j+1}^t X_i^* \geq k \right] \\ &= P_j. \end{aligned}$$

Thus, we have

$$\Pr \left[\sum_{i=1}^t X_i \geq k \right] = P_t \geq P_{t-1} \geq \dots \geq P_1 \geq P_0 = \Pr \left[\sum_{i=1}^t X_i^* \geq k \right].$$

□

Now we can prove the following lemma to complete the proof of our lower bound.

Lemma 10. *Let $n \in \mathbb{N}$ and $\ell = \omega(\log(n))$. Then the optimization time of the $(1+1)$ -EA on $G_{n,\ell}$ is $\Omega(n^2\ell)$ with high probability.*

Proof. The idea of this proof is similar to the one used in [DJW02] for the proof of the lower bound on the runtime of the $(1+1)$ -EA on the leading ones function LO. To prove the claim, we analyze how long it takes until the individual I contains the path $P := (s = 1, 2, \dots, \ell, \ell + 1)$. To this aim, we analyze how the length $L(I)$ of the longest subpath of P starting in s that is contained in I grows. Note that this length $L(I)$ never decreases, since for each vertex on P this subpath is the unique shortest path to s . In the following, we show that with high probability (i) $L(I)$ initially is constant, (ii) in $\Theta(n^2\ell)$ iterations $L(I)$ increases at most $O(\ell)$ times, and (iii) the total increase in these $O(\ell)$ relevant iterations (plus the initial constant length) is less than ℓ .

The probability that in the initial individual some vertex $i \in [2..\ell + 1]$ is already linked to $i - 1$ is exactly $\frac{1}{n-1}$. Hence the probability that $L(I) \geq c$ is $O(n^{-c})$ for all constants $c \in \mathbb{N}$, that is, with high probability $L(I)$ is initially constant.

Let t^* be the time step in which $L(I)$ increases to the maximal possible value of ℓ . For $i \in [1..t^*]$, we define a binary random variable X_i by $X_i = 1$ if $L(I)$ increases in step i . To increase $L(I)$, one of the $S + 1$ elementary mutations in the current step has to connect vertex $L(I)+2$ to vertex $L(I)+1$. The probability that an elementary mutation succeeds in doing so is $\frac{1}{(n-1)^2}$. By Lemma 7, the probability that one iteration does so is at most $\frac{2}{(n-1)^2}$. Hence $\Pr[X_i = 1] \leq p := \frac{2}{(n-1)^2}$. For $i > t^*$ define X_i by $\Pr[X_i = 1] := p$ and $\Pr[X_i = 0] := 1 - p$, independent of all other random variables.

Let $t := \eta(n-1)^2\ell$, where η is a constant to be chosen later. Let X_1^*, \dots, X_t^* be mutually independent random variables with $\Pr[X_i^* = 1] := p$ and $\Pr[X_i^* = 0] := 1 - p$ for all i . Let $X^* := \sum_{i=1}^t X_i^*$. Then $\Pr[X_i^* = 1] \geq \Pr[X_i = 1 | X_1 = x_1, \dots, X_{i-1} = x_{i-1}]$ for all $x_1, \dots, x_{i-1} \in \{0, 1\}$. Also,

$$\mathbb{E}[X^*] = pt = \eta(n-1)^2\ell \frac{2}{(n-1)^2} = 2\eta\ell.$$

Hence, applying Lemma 9 and using the first Chernoff bound from Theorem 8

with $\delta = 1$, we compute

$$\begin{aligned}
\Pr \left[\sum_{i=1}^t X_i \geq 4\eta\ell \right] &\leq \Pr[X^* \geq 2\mathbb{E}[X^*]] \\
&\leq \exp \left(-\frac{\mathbb{E}[X^*]}{3} \right) \\
&= \exp \left(-\frac{2\eta\ell}{3} \right) \\
&= \exp \left(-\frac{2\eta}{3} \log(n) \frac{\ell}{\log(n)} \right) \\
&= n^{-\frac{2\eta}{3} \frac{\ell}{\log(n)}} \\
&= n^{-\omega(1)}.
\end{aligned}$$

In the last lines we used that since $\ell = \omega(\log(n))$ we have $\frac{\eta\ell}{\log n} = \omega(1)$ for any constant η . Since $\sum_{i=1}^t X_i$ is an upper bound on the number of improvements in the first t iterations, this means that with high probability the $(1 + 1)$ -EA did at most $t' := 4\eta\ell$ improvements during these t iterations.

Finally, we analyze which value of $L(I)$ results from these t' improvements. Clearly, each improvement increases $L(I)$ by at least one. However, there are two ways how additional vertices i can be connected to the longest subpath of P starting in s . (i) There may be an elementary mutation in the iteration that causes the improvement that changes the pointer of i to its predecessor $i - 1$ in P , or (ii), i may coincidentally be connected to $i - 1$, which becomes part of the subpath by an event of type (i) or (ii). We shall argue that both events happen only with a probability of at most $\frac{1}{2}$.

Suppose first that i enlarges the path of interest through an elementary mutation. For this to happen (among other things), the following has to occur. Among the possible more than one elementary mutations in the current step that connect i to some other vertex, the last one has to connect i to its predecessor in P . By definition of the mutation operator, this happens with a probability of $\frac{1}{n-1} \leq \frac{1}{2}$.

Now suppose that the predecessor $i - 1$ of i becomes part of the subpath of interest. We argue that the probability that i is coincidentally pointing to $i - 1$ is at most the probability that it is pointing to s , and in consequence, at most $\frac{1}{2}$. Clearly, this holds for the initially chosen individual as here all vertices different from i have equal probability of being the target of i 's pointer. Consider an iteration that does not result in making $i - 1$ part of the subpath of interest, and fix a sequence of elementary mutations to be conducted in this iteration. Assume that at the start of the iteration vertex i

is pointing to vertex $j \in [1..t]$ for which the path from s to j in I has length $w_s(s, j)$. If at the end of the operation the path $w_e(s, i - 1)$ from s to $i - 1$ in I is at most as long as $w_s(s, j)$, then the acceptance of the iteration does not depend on whether i 's pointer is changed to $i - 1$ or s , since both events would not increase the length of the path from i to s in I . On the other hand, if at the end of the operation $w_e(s, i - 1) > w_s(s, j)$, only a change of i 's pointer to s , but not to $i - 1$, would be accepted. Thus, any iteration that does not make $i - 1$ part of the path rather increases the probability that i points to s compared to the one of pointing to $i - 1$. In consequence, the probability that i coincidentally points at $i - 1$ in the iteration in which $i - 1$ becomes part of the subpath of interest, is at most $\frac{1}{2}$.

Summarizing, an additional vertex may either already be coincidentally connected correctly (with probability at most $\frac{1}{2}$), or, if this is not the case (which happens with probability at least $\frac{1}{2}$), it may become connected by an elementary mutation (with probability at most $\frac{1}{2}$). Hence, the probability that an additional vertex becomes connected is at most $\frac{3}{4}$.

Let t'' be the number of improvement steps the $(1 + 1)$ -EA performs until it finds the optimal solution. Recall that with high probability, $t'' \leq t' = 4\eta\ell$. For $i \in [1..t'']$ let Y_i be the random variable describing the total number of vertices which are added in the i -th improvement step. By the above arguments, independent of the outcome of previous random choices, we have

$$\Pr[Y_i = m] \leq \left(\frac{3}{4}\right)^{m-1} \frac{1}{4}$$

for all $m \geq 1$.

If $t'' < t' = 4\eta\ell$, let Y_i for $i \in [t'' + 1..t']$ be defined by $\Pr[Y_i = m] := \left(\frac{3}{4}\right)^{m-1} \frac{1}{4}$ independent from all other Y_j . Define Y_i^* for $i \in [1..t']$ to be mutually independent random variables that are geometrically distributed with parameter $q = \frac{1}{4}$, that is, $\Pr[Y_i^* = m] := \left(\frac{3}{4}\right)^{m-1} \frac{1}{4}$ for all $m \geq 1$. The expected value of Y_i^* is $\mathbb{E}[Y_i^*] = q^{-1} = 4$. Let $Y := \sum_{i=1}^{t'} Y_i$ and $Y^* := \sum_{i=1}^{t'} Y_i^*$. Then $\mathbb{E}[Y^*] = 4t'$.

Applying Lemma 9 and the second bound in Theorem 8 with $\delta = 1$ and

assuming $t' \geq 2$ we get

$$\begin{aligned}
\Pr[Y \geq 8t'] &\leq \Pr[Y^* \geq 8t'] \\
&= \Pr[Y^* \geq 32\eta\ell] \\
&\leq \exp\left(-\frac{(t'-1)}{4}\right) \\
&\leq \exp\left(-\frac{t'}{2 \cdot 4}\right) \\
&= \exp\left(-\frac{\eta\ell}{2}\right) \\
&= \exp\left(-\frac{\eta}{2} \log(n) \frac{\ell}{\log(n)}\right) \\
&= n^{-\frac{\eta}{2} \frac{\ell}{\log(n)}} \\
&= n^{-\omega(1)}.
\end{aligned}$$

Thus, with high probability during up to $t' = 4\eta\ell$ improvements at most $32\eta\ell$ additional vertices become part of the shortest path. Choosing $\eta = \frac{1}{64}$, we see that with high probability, $L(I)$ is at most $c + 32\eta\ell = c + \frac{1}{2}\ell < \ell$ after $t = \eta(n-1)^2\ell$ iterations, that is, the path P has not been found in this time. \square

Combining Lemma 6 and Lemma 10 yields the following theorem.

Theorem 11 (Lower Bound). *For all values of n and ℓ , the optimization time of the $(1+1)$ -EA is $\Omega(n^2 \max\{\ell, \log(n)\})$ with high probability.*

5 Conclusions

In this work, we gave a tight runtime analysis of the $(1+1)$ -EA for the SSSP problem [STW04]. This includes improving the upper bound of $O(n^2\ell \log(\frac{n}{\ell}))$ implicit in a proof to $O(n^2 \max\{\ell, \log(n)\})$, and a carefully selected lower bound example for all values of n and ℓ .

At least as important as the precise bounds for this particular problem are the methods developed in this paper. Past arguments suggested a coupon-collector like behavior in finding the shortest paths. Those, however, cannot be employed to obtain such sharp bounds. Indeed, our analysis shows that the true behavior is different. Namely, the different shortest paths grow at comparable speeds that are strongly concentrated around their expected values.

While our work is very satisfying from the methodological point of view, some particular questions for the SSSP problem remain open. The most challenging one from a broader perspective is whether the multi-criteria fitness function is necessary. Recall that we accept a newly created individual only if for no vertex the distance to the source is increased. A natural (single-criteria) alternative would be to consider the average distance. [STW04] argue that the multi-criteria fitness function is necessary for the algorithm to run properly. However, their counterexample only works if vertices not connected to the source are assumed to have an infinite distance to the source. In this case, changing the number of ∞ -distance vertices does not change the average distance, and hence the EA finds itself on a large plateau of constant fitness. A simple way to overcome this (and the one you would choose naturally in an implementation) would be to replace the infinite distance of such vertices by a large, but finite number. We strongly believe that in this case, taking the average distance as fitness would result in the same run-time behavior. However, the analysis seems to be much harder, due to the fact that the individual can develop in less conservative ways.

This problem resembles the one of maximizing linear functions $f : \{0, 1\}^n \rightarrow \mathbb{R}, x \mapsto \sum_{i=1}^n a_i x_i$ with positive coefficients a_i . Viewing f as the multi-criteria fitness function $x \mapsto (a_1 x_1, \dots, a_n x_n)$, a simple coupon collector argument shows an optimization time of $\Theta(n \log(n))$. That the same bound also holds for the fitness function f itself is the result of a highly complex analysis by [DJW02]. Attempts to simplify this result later led to the invention of the drift analysis method in evolutionary computation (cf. [HY04]). With this development in mind, it seems likely that it is very difficult to prove that a single-criteria EA can solve the SSSP efficiently.

Note added in proof: Some progress on this problem has been made in [BBD⁺09]. There the authors showed that the single-criteria formulation of the SSSP problem can be solved in polynomial time. However, the run-time bounds obtained are not tight. Hence a full understanding of this problem is still missing.

References

- [AD11] Anne Auger and Benjamin Doerr, editors. *Theory of Randomized Search Heuristics*. World Scientific, 2011.
- [AS08] N. Alon and J. H. Spencer. *The Probabilistic Method*. Wiley, 3rd edition, 2008.
- [BBD⁺09] Surender Baswana, Somenath Biswas, Benjamin Doerr, Tobias

- Friedrich, Piyush P. Kurur, and Frank Neumann. Computing single source shortest paths using single-objective fitness. In *FOGA '09: Proceedings of the Tenth ACM SIGEVO Workshop on Foundations of Genetic Algorithms*, pages 59–66, New York, NY, USA, 2009. ACM.
- [DG10] Benjamin Doerr and Leslie Goldberg. Drift analysis with tail bounds. In *Proceedings of Parallel Problem Solving from Nature (PPSN XI), Part I*, LNCS 6238, pages 174–183. Springer, 2010.
- [DHN06] Benjamin Doerr, Nils Hebbinghaus, and Frank Neumann. Speeding up evolutionary algorithms through restricted mutation operators. In *Proceedings of the 9th International Conference on Parallel Problem Solving From Nature (PPSN)*, volume 4193 of *Lecture Notes in Computer Science*, pages 978–987. Springer, 2006.
- [Dij59] Edsger. W. Dijkstra. A note on two problems in connexion with graphs. In *Numerische Mathematik*, volume 1, pages 269–271. Mathematisch Centrum, Amsterdam, The Netherlands, 1959.
- [DJ07] Benjamin Doerr and Daniel Johannsen. Adjacency list matchings — an ideal genotype for cycle covers. In *Proceedings of the 2007 Conference on Genetic and Evolutionary Computation (GECCO)*, pages 1203–1210. ACM Press, 2007.
- [DJW02] Stefan Droste, Thomas Jansen, and Ingo Wegener. On the analysis of the (1+1) evolutionary algorithm. *Theoretical Computer Science*, 276(1–2):51–81, 2002.
- [DJW10] Benjamin Doerr, Daniel Johannsen, and Carola Winzen. Multiplicative drift analysis. In *Proceedings of Genetic and Evolutionary Computation Conference (GECCO 2010)*, pages 1449–1456. ACM, 2010.
- [DKS07] Benjamin Doerr, Christian Klein, and Tobias Storch. Faster evolutionary algorithms by superior graph representation. In *Proceedings of the First IEEE Symposium on Foundations of Computational Intelligence (FOCI)*, pages 245–250. IEEE Press, 2007.
- [GW03] Oliver Giel and Ingo Wegener. Evolutionary algorithms and the maximum matching problem. In *Proceedings of the 20th Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 2607 of *Lecture Notes in Computer Science*, pages 415–426. Springer, 2003.

- [HY04] Jun He and Xin Yao. A study of drift analysis for estimating computation time of evolutionary algorithms. *Natural Computing*, 3(1):21–35, 2004.
- [Jäg08] Jens Jägersküpper. A blend of Markov-chain and drift analysis. In *Proceedings of PPSN X*, LNCS 5199, pages 41–51. Springer, 2008.
- [Neu04] Frank Neumann. Expected runtimes of evolutionary algorithms for the Eulerian cycle problem. In *Proceedings of the 2004 IEEE Congress on Evolutionary Computation (CEC)*, pages 904–910. IEEE Press, 2004.
- [NW04] Frank Neumann and Ingo Wegener. Randomized local search, evolutionary algorithms, and the minimum spanning tree problem. In *Proceedings of the 2004 Conference on Genetic and Evolutionary Computation (GECCO)*, volume 3102 of *Lecture Notes in Computer Science*, pages 713–724. Springer, 2004.
- [NW05] Frank Neumann and Ingo Wegener. Minimum spanning trees made easier via multi-objective optimization. In *Proceedings of the 2005 Conference on Genetic and Evolutionary Computation (GECCO)*, pages 763–769. ACM Press, 2005.
- [NW10] Frank Neumann and Carsten Witt. *Bioinspired Computation in Combinatorial Optimization – Algorithms and Their Computational Complexity*. Springer, 2010.
- [STW04] Jens Scharnow, Karsten Tinnefeld, and Ingo Wegener. The analysis of evolutionary algorithms on sorting and shortest paths problems. *Journal of Mathematical Modelling and Algorithms*, 3(4):349–366, 2004.
- [Weg01] Ingo Wegener. Theoretical aspects of evolutionary algorithms. In *Proceedings of the 28th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 2076 of *Lecture Notes in Computer Science*, pages 64–78. Springer, 2001.
- [Wit05] Carsten Witt. Worst-case and average-case approximations by simple randomized search heuristics. In Volker Diekert and Bruno Durand, editors, *Proceedings of the 22nd Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 3404 of *Lecture Notes in Computer Science*, pages 44–56. Springer, 2005.

- [WW05] Ingo Wegener and Carsten Witt. On the optimization of monotone polynomials by simple randomized search heuristics. *Combinatorics, Probability and Computing*, 14(1–2):225–247, 2005.