

A Tight Analysis of the $(1 + 1)$ -EA for the Single Source Shortest Path Problem

Benjamin Doerr
Max-Planck-Institut für Informatik
Stuhlsatzenhausweg 85
66123 Saarbrücken, Germany

Edda Happ
Max-Planck-Institut für Informatik
Stuhlsatzenhausweg 85
66123 Saarbrücken, Germany

Christian Klein
Max-Planck-Institut für Informatik
Stuhlsatzenhausweg 85
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 a tight bound of $\Theta(n^2 \max\{\log(n), \ell\})$ on the optimization time, where ℓ is the maximum number of edges of a shortest path with minimum number of edges from the source to another vertex. Using various tools from probability theory we show that these bounds not only hold in expectation, but also with high probability. We are optimistic that these tools can also be used to analyze the run-time of evolutionary algorithms for other problems.

I. INTRODUCTION

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 rather artificial problems like maximizing 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 [7], [15], [16]. These problems already lead to quite some insight on how evolutionary algorithms work and on how to analyze them.

More recently, the focus moves on to classical problems from computer science. One of the first papers in this direction is [14], where evolutionary algorithms for sorting and shortest path problems are proposed and analyzed. Other results on evolutionary algorithms for combinatorial optimization problems include a series of papers on the Eulerian cycle problem [4]–[6], [11], minimum spanning trees [12], [13] or maximal matchings [8].

For such closer to the real world problems, the analysis of evolutionary algorithms becomes even more interesting. One simple reason for this is that there we typically have the choice between several natural variation operators, representations of the individuals, and fitness functions.

To avoid misunderstandings, let us stress that the focus of such 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 building blocks like particular representations or variation operators work, and, finally, to increase the understanding of how to analyze evolutionary algorithms.

Surprisingly, even for extremely simple evolutionary algorithms on simple problems a tight analysis of their run-time behavior can be very complicated. A classical example is the (tight) $O(n \log(n))$ bound for the optimization time (i.e., number of fitness evaluations necessary to find the optimum) of a simple $(1 + 1)$ evolutionary algorithm maximizing a monotone linear function on $\{0, 1\}^n$. Here, direct methods resulted in a complicated seven pages proof [7], and only drift analysis [9] leads to reasonable proofs.

In this paper, we analyze the $(1 + 1)$ evolutionary algorithm proposed by Scharnow, Tinnefeld and Wegener in [14] for the problem of finding in a graph with edge weights a shortest path from a single node to all other vertices (see below for a precise definition of the problem). Scharnow et. al. show an upper bound of $O(n^3)$ for the expected optimization time on n -vertex graphs. This bound is tight for certain graphs and weights. The proof of the upper bound reveals an in fact stronger upper bound of $O(n^2 \sum_{i=1}^n \log(n_i))$, where n_i is the number of vertices for which the shortest path to the source with the minimum number of edges consists of exactly i edges. In particular, this yields a bound of $O(n^2 \ell \log(n))$, where ℓ is the maximum number of edges on such a shortest path connecting the source and some vertex.

In this work, we give a tight analysis of the proposed algorithm and show that the expected optimization time is bounded by $O(n^2 \max\{\log(n), \ell\})$. In fact, this bound holds not only in expectation, but is also fulfilled with high probability. This bound is tight for all ℓ , that is, 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 $\Omega(n^2 \max\{\log(n), \ell\})$ with high probability.

The proof of the upper bound uses a nice strong concentration behavior on the way long shortest paths are found, which

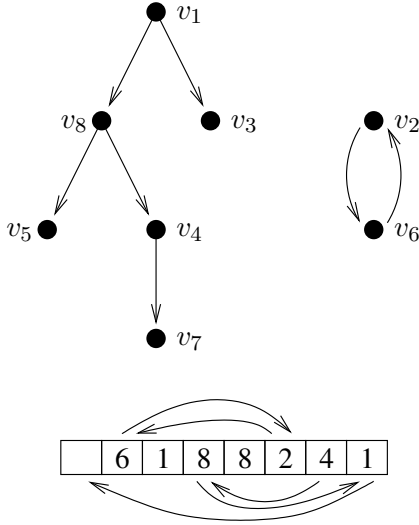


Fig. 1. The graph representation and the vector representation of an individual not representing a tree. The fitness of v_7 is $d_{74} + d_{48} + d_{81}$, as represented by the arrows below the vector representation. The fitness of v_2 is ∞ , since there is no path from v_2 to v_1 , as shown by the arrows above the vector representation.

will probably see more applications in the near future.

II. THE SINGLE SOURCE SHORTEST PATH PROBLEM

For a directed graph $G = (V, E)$ ($|V| = n$) with edge weights $w : E \rightarrow \mathbb{N}$ the single source shortest path problem (SSSP) is the problem of finding for each $v \in V$ the shortest path from a given source $s = v_1 \in V$ to v . Dijkstra's famous algorithm [3] solves the problem in time $O(n^2)$.

If we allow the weight $w(e) = \infty$ for not existing edges $e \notin E$ we can consider the complete graph K_n on n vertices. The problem instance is given by the distance matrix $D = (d_{ij})_{1 \leq i, j \leq n}$ of the graph, where $d_{ij} = w(v_i, v_j) \in \mathbb{N} \cup \{\infty\}$. Note that the below described algorithm also works in the case of undirected graphs. For each undirected edge $e = \{v_i, v_j\}$ simply set $d_{ij} = d_{ji} = w(e)$.

A well known way of representing solutions to the SSSP problem is to give for each vertex v_i its predecessor $p(v_i)$ on a shortest path from s to v_i . Such a solution will form a tree rooted at s .

In this paper, we analyze the simple $(1+1)$ evolutionary algorithm $((1+1)$ -EA) for the SSSP problem introduced in [14]. It represents the candidate solutions as vectors of predecessors $I = (p(v_2), \dots, p(v_n)) \in \{1, \dots, n\}^{n-1}$ and thereby implicitly as trees or tree-like structures. This representation itself does not assure that an individual forms a tree, see Figure 1 for an example. A fitness-optimal individual, however, will always represent a tree.

At the beginning, the $(1+1)$ -EA generates an initial individual I by assigning to each vertex v a predecessor $p(v)$ 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 I 's fitness. 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 is infeasible for more complex representations, like the one we use here, this behavior must be simulated. To do this, a number S is chosen at random according to a Poisson distribution¹ $\text{Pois}(\lambda = 1)$ with parameter $\lambda = 1$. A mutation step consists of sequentially applying $S + 1$ single mutations to I .

A single mutation of the vector I consists of randomly choosing a vertex v_i with $i \in [2..n]$ and setting its predecessor $p(v_i)$ to a vertex v_j chosen uniformly at random with $j \in [1..n]$ and $j \neq i$. Obviously, there are $(n-1)^2$ possible ways to choose a vertex and its predecessor and thus to do a single mutation on individual I .

As fitness function, we use the multi-criteria fitness function described in [14]. For an individual I , it is defined as $f(I) := (f_2(I), \dots, f_n(I))$ with

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

Here, $w(P(s, v_i))$ is the cost of the path P from s to v_i implied by I . If this path is $P = (s, v_2, \dots, v_i)$ then $w(P(s, v_i)) = d_{12} + d_{23} + \dots + d_{i-1i}$. See Figure 1 for an example.

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_i , the $(1+1)$ -EA does not accept mutations that would cause s to be connected to v_i using a suboptimal path. Pseudo-code for the $(1+1)$ -EA for the SSSP problem is given in Figure 2.

The performance of evolutionary algorithms is usually measured in terms of *optimization time* which is the number of evaluations of the fitness function the algorithm executes until it finds an optimal solution. The time needed for the creation of new individuals by mutation or crossover and for the execution of the fitness function is usually disregarded.

III. THE UPPER BOUND

In this section we show that the optimization time of the $(1+1)$ -EA for the SSSP problem is upper bounded by $O(n^2 \max\{\log(n), \ell\})$ with high probability. Here, ℓ is the maximum number of edges of all shortest paths with a minimum number of edges (see definition below) and "with high probability" means that an event happens with probability n^{-c} for an arbitrary but fixed constant c .

To prove this theorem we need the following classical result from probability theory (cf. [1]) and a number of definitions and lemmas.

Theorem 1 (Chernoff Bound): Let X_i be mutually independent variables with $\Pr[X_i = 1] = p$ and $\Pr[X_i = 0] = 1 - p$.

¹The Poisson distribution with $\lambda = 1$ is used because it is the limit of the Binomial distribution for n trials with probability $\frac{1}{n}$ each.

(1 + 1)-EA FOR SSSP

Initialization:

1 $I \leftarrow (p(v_2), \dots, p(v_n)),$
 $p(v_i) \in V \setminus v_i$ 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 $j \in [1..n], j \neq i$ u. a. r.

9 $I^k \leftarrow I^{k-1} : p^{k-1}(v_i) \leftarrow v_j$

Selection:

10 **if** $f_i(I^k) \leq f_i(I)$ for all $i \in [2..n]$

11 **then** $I \leftarrow I^k$

12 **forever**

Fig. 2. An (1 + 1)-EA for the SSSP problem.

Let $X := \sum_{i=1}^t X_i$. Then for all $a > 0$,

$$\Pr[X < \mathbb{E}[X] - a] \leq \exp\left(-\frac{a^2}{2pt}\right).$$

Definition 2 (Edge Radius): The edge radius $\ell_G(s)$ of a vertex s in a weighted graph G is the maximum number of edges of any shortest path with minimum number of edges from s to a vertex v in G .

$$\ell_G(s) = \max_{v \in V} \left\{ \min_{P \in \mathcal{P}_v} \{\ell(P)\} \right\}$$

with $\mathcal{P}_v := \{P | P \text{ is a shortest path from } s \text{ to } v\}$ and $\ell(P)$ being the number of edges of path P .

For convenience we use the abbreviation $\ell := \ell_G(s)$.

Lemma 3: If $\ell \geq \log(n)$, the optimization time needed by the (1 + 1)-EA to find all shortest paths is with high probability less than $t = \lambda(n - 1)^2 \ell$ for a sufficiently large constant λ .

Proof: Because of the multi-criteria fitness function, the (1 + 1)-EA cannot replace any path in the individual I by a longer path. Thus, any successful mutation step that would apply more than one mutation can be simulated by a number of successful mutation steps applying a single mutation. Since the probability for such a single mutation step is constant, for the upper bound analysis we can assume that only single mutation steps are applied.

The (1 + 1)-EA has to find the $n - 1$ shortest paths from the source s to every vertex v . There may be many different possible shortest paths for a vertex v . For each v we fix one shortest path P and show that all the mutation steps needed to create this path will be executed in expected time $t = \lambda(n - 1)^2 \ell$ for a constant $\lambda > 0$. However, meanwhile optimal sub-paths of P may be exchanged for different optimal sub-paths so that the resulting path might not be P .

Pick a vertex v and a shortest path $P := (v^1, \dots, v^{\ell+1})$ from $s = v^1$ to $v = v^{\ell+1}$. Note that by the definition of the edge radius ℓ we can pick P so that it has $\ell' \leq \ell$ edges. We call a single mutation step the j -th improvement of P if before the mutation the individual I contains a shortest path from s to v^j for some $1 \leq j \leq \ell'$ and after the mutation it contains a shortest path from s to v^{j+1} . Note that I might contain a shortest path from s to v^{j+1} before the j -th improvement. If the (1 + 1)-EA has performed the ℓ' -th improvement we say it has followed P . Obviously, a shortest path from s to $v = v^{\ell'+1}$ has been found by then.

Let t' be the number of steps the (1 + 1)-EA needs to follow P . Define the random variables X_i for $1 \leq i \leq t'$ as $X_i = 1$ if the i -th mutation step is an improvement of P and $X_i = 0$ otherwise. Then $\Pr[X_i = 1] \geq p = \frac{1}{(n-1)^2}$, since it suffices if the i -th mutation step picks the correct vertex with probability $\frac{1}{n-1}$ and sets it to its correct predecessor with probability $\frac{1}{n-1}$. For $i > t'$ define $\Pr[X_i = 1] = p$ and $\Pr[X_i = 0] = 1 - p$. Since we only consider single mutation steps, the X_i are mutually independent. Hence, the expected value of $X := \sum_{i=1}^t X_i$ is $\mathbb{E}[X] \geq pt$.

If the (1 + 1)-EA has not found an optimal path from s to v , it obviously has not followed P and thus $X < \ell$. Thus, the probability of not finding a shortest path from s to v in time t can be bounded by

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

For $a = (\lambda - 1)\ell$ and $t = \lambda(n - 1)^2 \ell$ we have

$$\begin{aligned} \mathbb{E}[X] - a &\geq pt - a \\ &= \frac{1}{(n-1)^2} \lambda(n-1)^2 \ell - (\lambda-1)\ell \\ &= \ell. \end{aligned}$$

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

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

for $\lambda \geq 2$.

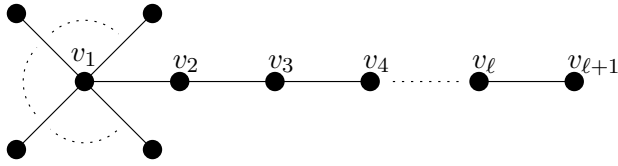


Fig. 3. The SSSP tree of $G_{n,\ell}$ with source v_1 .

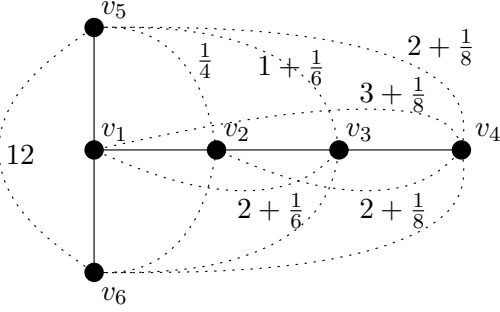


Fig. 4. The graph $G_{6,3}$. The solid edges have weight one and form the shortest path tree.

Using the above calculations we get

$$\begin{aligned}
\Pr \left[\begin{array}{l} \text{not for all } v_i \text{ a short-} \\ \text{test path from } s \text{ to} \\ v_i \text{ found in time } t \end{array} \right] &\leq \sum_{i=2}^n \Pr \left[\begin{array}{l} \text{no shortest path} \\ \text{from } s \text{ to } v_i \\ \text{found in time } t \end{array} \right] \\
&\leq \sum_{i=2}^n \exp \left(-\frac{\lambda}{8} \ell \right) \\
&\leq n \exp \left(-\frac{\lambda}{8} \log(n) \right) \\
&\leq n^{1-\frac{\lambda}{8}}.
\end{aligned}$$

Hence, by choosing λ appropriately, we can achieve a failure probability of $O(n^{-c})$ for any c . Note that we did not optimize for λ . ■

Corollary 4: If $\ell < \log(n)$ the optimization time needed by the $(1+1)$ -EA to find all shortest paths is less than $t = \lambda'(n-1)^2 \log(n)$ with high probability for a sufficiently large constant λ' .

Proof: Consider the graph G' , which is G with a shortest path of length $\log(n)$ added. Then by Lemma 3 the $(1+1)$ -EA needs time t to find a shortest paths from s to all vertices in G' with high probability. But surely the $(1+1)$ -EA cannot optimize G' faster than G . ■

Theorem 5: The optimization time of the $(1+1)$ -EA is $O(n^2 \max\{\log(n), \ell\})$ with high probability.

Proof: The proof follows directly from Lemma 3 and Corollary 4. ■

IV. THE LOWER BOUND

In this section we show a lower bound matching the upper bound presented in the previous section.

A. A Worst Case Graph Class

Let $V = \{v_1, \dots, v_n\}$ be a set of n vertices. For all $\ell \in [1..n-1]$ we define a graph $G_{n,\ell} = (V, E)$ such that the source of the SSSP tree to be computed is $s = v_1$ and $\ell_G(s) = \ell$. We show that the expected optimization time of the $(1+1)$ -EA is $\Omega(n^2 \max\{\log(n), \ell\})$.

We will set the weights in such a way that $(s, v_2, \dots, v_\ell, v_{\ell+1})$ is the unique shortest path from s to $v_{\ell+1}$. For all other vertices v_k with $k > \ell + 1$, the edge (s, v_k) shall be the unique shortest path. 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 expected optimization time linearly depends on ℓ , we must choose the right weights for the edges not in the SSSP tree. Each vertex $v_i, i \in [2..l+1]$ should be connected to both s and $v_k, k > \ell + 1$ with edges ensuring that, as long as v_{i-1} is not correctly connected, it is cheaper to connect v_i using those edges instead of connecting it to v_{i-1} using the edge of weight 1. This ensures that most of the edges on the shortest path are added one after the other. The formal definition of the edge weights $w(v_i, v_j), i, j \in [1..n], i < j$ is as follows.

$$w(v_i, v_j) := \begin{cases} 1, & \text{if } j = i + 1 \leq \ell + 1, \\ 1, & \text{if } i = 1 \wedge j > \ell + 1, \\ j - i + \frac{1}{2^j}, & \text{if } 1 \leq i < j \leq \ell + 1, \\ i - 2 + \frac{1}{2^i}, & \text{if } i \leq \ell + 1 \wedge j > \ell + 1, \\ 2n, & \text{if } i, j > \ell + 1. \end{cases}$$

The graph $G_{6,3}$ is shown in Figure 4. Note that $G_{n,1}$ is the graph with edge weight 1 for each edge $(s, v_i), i \in [2..n]$ and $2n$ for all other edges.

B. A Lower Bound

We now give a lower bound on the expected number of steps needed by the $(1+1)$ -EA depending on n and ℓ . To prove that $\Omega(n^2 \max\{\log(n), \ell\})$ is a lower bound on the expected optimization time of the $(1+1)$ -EA, we first prove that $\Omega(n^2 \log(n))$ is a lower bound on the expected optimization time. Observe that this bound holds 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 number of steps needed by the $(1+1)$ -EA to compute the SSSP tree of G with source s is $\Omega(n^2 \log(n))$ with high probability.

Proof: Some arguments of the proof are similar to the proof for the coupon collector's problem (cf. the book by Motwani and Raghavan [10]). We only prove the bound on the expected optimization time here, and omit the proof for the high probability, as it is very technical and does not differ much from the proof in the above mentioned book.

As each vertex of G is incident with $n-1$ edges, due to the uniqueness of the SSSP tree the probability that a vertex is connected correctly after the initialization step is $\Theta(\frac{1}{n})$. Thus, in expectation, only a constant number of vertices is connected correctly at the beginning.

Let T denote the random variable describing the number of steps needed by an $(1+1)$ -EA to find the shortest path tree with source s . The expected value of T can then be bounded by

$$\mathbb{E}[T] = \sum_{i=1}^{\infty} i \Pr[T = i] \geq t \Pr[T \geq t]$$

for any fixed $t \geq 1$. The probability $\Pr[T \geq t]$ that the $(1+1)$ -EA needs at least t steps, however, is the same as the probability that the $(1+1)$ -EA will not finish after $t-1$ steps. Hence, if we can bound this probability from below for a given t , we get a lower bound on the expected optimization time.

For this bound, consider a vertex v not yet connected correctly. To construct the shortest path tree, v has to be chosen at least once together with its correct predecessor v' to connect it using the right edge. The probability that this happens in a single mutation is $\frac{1}{n-1} \cdot \frac{1}{n-1} = \frac{1}{(n-1)^2}$.

By the definition of the Poisson distribution, the probability that during a single step exactly $s+1$ mutations are performed is $\frac{1}{e \cdot s!}$. Since each of the $s+1$ mutations are done independently, the probability that a fixed mutation is performed in a certain step is at most $\frac{s+1}{(n-1)^2}$. Hence the probability that a fixed mutation is performed in one step is at most

$$\sum_{s=0}^{\infty} \frac{1}{e \cdot s!} \cdot \frac{s+1}{(n-1)^2} = \frac{2}{(n-1)^2}$$

since

$$\begin{aligned} \sum_{s=0}^{\infty} \frac{s+1}{s!} &= \sum_{s=1}^{\infty} \frac{s}{s!} + \sum_{s=0}^{\infty} \frac{1}{s!} \\ &= \sum_{s=1}^{\infty} \frac{1}{(s-1)!} + e \\ &= 2e. \end{aligned}$$

By this, the probability that a fixed mutation is never chosen during $t-1$ steps is at least $(1 - \frac{2}{(n-1)^2})^{t-1}$. Hence it follows that this fixed mutation is tried at least once in $t-1$ steps with probability at most $1 - (1 - \frac{2}{(n-1)^2})^{t-1}$. Since there are $n-1$ correct mutations, the probability that all of them are tried is at most $(1 - (1 - \frac{2}{(n-1)^2})^{t-1})^{n-1}$. But this means that with probability at least $1 - (1 - (1 - \frac{2}{(n-1)^2})^{t-1})^{n-1}$ at least one of the combinations is never chosen during the $t-1$ steps. We now choose $t := 1 + ((n-1)^2 - 1) \log(n-1)$ to obtain

$$\begin{aligned} &1 - \left(1 - \left(1 - \frac{1}{(n-1)^2}\right)^{((n-1)^2-1) \log(n-1)}\right)^{n-1} \\ &\geq 1 - e^{-1}. \end{aligned}$$

Hence, for this value of t we have $\Pr[T \geq t] \geq 1 - e^{-1}$. Thus the expected number of steps needed is bounded by

$$\begin{aligned} \mathbb{E}[T] &\geq (1 - e^{-1}) \cdot (1 + ((n-1)^2 - 1) \log(n-1)) \\ &= \Omega(n^2 \log(n)). \end{aligned}$$

■

By the above lemma, our lower bound is tight as long as $\ell \in O(\log(n))$. To complete our claim, however, we need to prove that for larger ℓ the expected optimization time linearly depends on ℓ . For this we need the following inequalities.

Theorem 7: Let $X_i, i \in [1..n]$ be independent random variables and let $X := \sum_{i=1}^n X_i$. Then the following holds.

- a) If $\Pr[X_i = 1] = p$ and $\Pr[X_i = 0] = 1-p$ and $\delta \in [0, 1]$, then

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

This is a classical Chernoff bound (cf. [1]).

- b) Let $w > 0$. If $\Pr[X_i > wL] \leq 2^{-L}$ for all $L \in \mathbb{N}$, then

$$\Pr[X \geq 3\delta n w] \leq \exp(-(\delta - 1)n)$$

holds for all $\delta \in \mathbb{N}$. This bound can be found in [2]. ■

Lemma 8: Let $n \in \mathbb{N}$ and $\ell \in \omega(\log(n))$. Then the optimization time of the $(1+1)$ -EA on $G_{n,\ell}$ is $\Omega(n^2 \ell)$ with high probability.

Proof: We only consider the number of steps needed until the individual I contains the path $P := (s, v_2, \dots, v_\ell, v_{\ell+1})$. As a vertex is correctly connected after the initialization with probability $\Theta(\frac{1}{n})$, with high probability only a constant number of the vertices of P is correctly connected at the beginning.

Since there is only one good mutation for each vertex, the probability for a good mutation is $\Theta(\frac{1}{n})$. Hence, the $(1+1)$ -EA will with high probability only apply a constant number of mutations in each step, as the number of steps is bounded by $O(n^2 \ell)$ until the optimum is found.

Now set $t := \lambda(n-1)^2 \ell$ for a constant λ . We define random variables X_i , where $X_i = 1$ if the part of P discovered by the $(1+1)$ -EA increases in step i . Here, a vertex $v \in P$ is called discovered, if its shortest path has been found. Let v_j be the vertex of P with smallest j for which the $(1+1)$ -EA has not yet found the shortest path. As there is exactly one mutation increasing the discovered part of P , namely that connecting v_j to v_{j-1} the probability that this happens is $\frac{1}{(n-1)^2}$. As the $(1+1)$ -EA in expectation applies a constant number of mutations per step, we have $p := \Pr[X_i = 1] = \frac{c}{(n-1)^2}$ for some constant $c > 1$. The expected value of $X := \sum_{i=1}^t X_i$ is then

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

Hence, using the Chernoff bound from Theorem 7 we get

$$\begin{aligned} \Pr[X \geq 2\mathbb{E}[X]] &= \Pr[X \geq 2c\lambda \ell] \\ &\leq \exp\left(-\frac{\mathbb{E}[X]}{3}\right) \\ &= \exp\left(-\frac{c\lambda \ell}{3}\right) \\ &\leq \exp\left(-\frac{c\lambda \log(n)}{3}\right) \\ &= n^{-\frac{1}{3}c\lambda}. \end{aligned}$$

But this means, that with high probability the $(1+1)$ -EA did $t' := 2c\lambda\ell$ improvements during the t steps.

Next we must analyze the number of vertices correctly connected during an improvement. As argued at the beginning of the proof, only a constant number c' of vertices that are not correctly connected can become correctly connected during one mutation step. However, there may be quite some vertices $v_k \in P, k > j$ following v_j that are already correctly connected to their predecessor. Thus, v_j may be the first vertex of a sub-path of P that is already correctly connected, and connecting v_j to v_{j-1} will enlarge the part of P that the $(1+1)$ -EA has already found by this sub-path.

To bound the number of vertices added during one improvement, we make the following observation. As long as v_j is not connected, its fitness is $f(v_j) \geq j - 1 + \frac{1}{2j}$ by construction of $G_{n,\ell}$. But then all vertices $v_k \in P, k > j$, would have fitness $f(v_k) \geq k - 1 + \frac{1}{2j}$, if they were connected to v_j using the 1-edges on the shortest path. For each such vertex v_k , there is at least one edge, namely (s, v_k) , that would connect v_k in such a way that its fitness is $f(v_k) = k - 1 + \frac{1}{2k} < k - 1 + \frac{1}{2j}$. Hence, with probability at least $\frac{1}{2}$ they are not connected using 1-edges, as there is at least one mutation that would yield a better fitness of v_k than using the 1-edge connecting it to v_{k-1} .

Let Y_i be the random variable describing the number of additional vertices which are added in the i -th improvement step. By the above argument, we can assume Y_i to be geometrically distributed with parameter $q = \frac{1}{2}$ and $\Pr[Y_i = h] = q^{h-1}q = 2^{-h}$, as for each additional vertex added, its predecessors up to v_j must be correctly connected.

We define $Y := \sum_{i=1}^{t'} Y_i$. Since $\Pr[Y_i > h] < 2^{-h}$ for all $i \in [1..t']$, we can apply the bound from Theorem 7 with $w = 1$ to get

$$\begin{aligned} \Pr[Y \geq 3\delta t'] &\leq \exp(-(\delta-1)t') \\ &= \exp(-(\delta-1)2c\lambda\ell) \\ &\leq \exp(-(\delta-1)2c\lambda \log(n)) \\ &= \frac{1}{n^{2c\lambda(\delta-1)}}. \end{aligned}$$

Since $3\delta t' \in \Theta(\ell)$ for any constant δ , the lemma follows. ■

Combining Lemma 6 and Lemma 8 yields the following theorem.

Theorem 9 (Lower Bound): The optimization time needed by the $(1+1)$ -EA to solve the SSSP problem is $\Omega(n^2 \max\{\log(n), \ell\})$ with high probability.

V. IMPROVEMENTS FOR SPARSE GRAPHS

In applications, most graphs are sparse, i.e., they only have $m = o(n^2)$ edges. We will now give an easy modification of the $(1+1)$ -EA with improved expected optimization time of $O(m \max\{\log(n), \ell\})$ on sparse graphs.

For a sparse graph, the distance matrix D contains many entries with $d_{ij} = \infty$. Denote the set containing all finite entries of D as $D^+ := \{d_{ij} \mid d_{ij} < \infty\}$ with $|D^+| = m$.

To improve the optimization time on sparse graphs, the mutation operator is changed as follows. Instead of picking

one vertex $v_i, i \in [2..n]$ and setting its predecessor $p(v_i)$ to $v_j, j \in [1..n], j \neq i$, the new mutation operator picks one entry $d_{ij} \in D^+$ uniformly at random and sets $p(v_i) = v_j$.

Theorem 10: The optimization time of the modified $(1+1)$ -EA described above is $O(m \max\{\log(n), \ell\})$ with high probability.

Proof: Simply plug $t := \lambda m \ell$ and $p := \Pr[X_i = 1] = \frac{1}{m}$ into the proof of Theorem 5. ■

REFERENCES

- [1] N. Alon and J. H. Spencer, *The Probabilistic Method*, 2nd ed. Wiley, 2000.
- [2] M. Dietzfelbinger and F. Meyer auf der Heide, "A new universal class of hash functions and dynamic hashing in real time," in *Proceedings of the 17th International Colloquium on Automata, Languages and Programming (ICALP)*, ser. Lecture Notes in Computer Science, vol. 443. Springer, 1990, pp. 6–19.
- [3] E. W. Dijkstra, "A note on two problems in connexion with graphs," in *Numerische Mathematik*. Mathematisch Centrum, Amsterdam, The Netherlands, 1959, vol. 1, pp. 269–271.
- [4] B. Doerr, N. Hebbinghaus, and F. Neumann, "Speeding up evolutionary algorithms through restricted mutation operators," in *Proceedings of the 9th International Conference on Parallel Problem Solving From Nature (PPSN)*, ser. Lecture Notes in Computer Science, vol. 4193. Springer, 2006, pp. 978–987.
- [5] B. Doerr and D. Johannsen, "Adjacency list matchings — an ideal genotype for cycle covers," in *Proceedings of the 2007 Conference on Genetic and Evolutionary Computation (GECCO)*. ACM Press, 2007, to appear.
- [6] B. Doerr, C. Klein, and T. Storch, "Faster evolutionary algorithms by superior graph representation," in *Proceedings of the First IEEE Symposium on Foundations of Computational Intelligence (FOCI)*. IEEE Press, 2007, pp. 245–250.
- [7] S. Droste, T. Jansen, and I. Wegener, "On the analysis of the $(1+1)$ evolutionary algorithm," *Theoretical Computer Science*, vol. 276, no. 1-2, pp. 51–81, 2002.
- [8] O. Giel and I. Wegener, "Evolutionary algorithms and the maximum matching problem," in *Proceedings of the 20th Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, ser. Lecture Notes in Computer Science, vol. 2607. Springer, 2003, pp. 415–426.
- [9] J. He and X. Yao, "A study of drift analysis for estimating computation time of evolutionary algorithms," *Natural Computing*, vol. 3, no. 1, pp. 21–35, 2004.
- [10] R. Motwani and P. Raghavan, *Randomized Algorithms*. Cambridge University Press, 1995.
- [11] F. Neumann, "Expected runtimes of evolutionary algorithms for the Eulerian cycle problem," in *Proceedings of the 2004 IEEE Congress on Evolutionary Computation (CEC)*. IEEE Press, 2004, pp. 904–910.
- [12] F. Neumann and I. Wegener, "Randomized local search, evolutionary algorithms, and the minimum spanning tree problem," in *Proceedings of the 2004 conference on Genetic and evolutionary computation (GECCO)*, ser. Lecture Notes in Computer Science, vol. 3102. Springer, 2004, pp. 713–724.
- [13] —, "Minimum spanning trees made easier via multi-objective optimization," in *Proceedings of the 2005 Conference on Genetic and Evolutionary Computation (GECCO)*. ACM Press, 2005, pp. 763–769.
- [14] J. Scharnow, K. Tinnefeld, and I. Wegener, "The analysis of evolutionary algorithms on sorting and shortest paths problems," *Journal of Mathematical Modelling and Algorithms*, vol. 3, no. 4, pp. 349–366, 2004.
- [15] I. Wegener, "Theoretical aspects of evolutionary algorithms," in *Proceedings of the 28th International Colloquium on Automata, Languages and Programming (ICALP)*, ser. Lecture Notes in Computer Science, vol. 2076. Springer, 2001, pp. 64–78, invited paper.
- [16] I. Wegener and C. Witt, "On the optimization of monotone polynomials by simple randomized search heuristics," *Combinatorics, Probability and Computing*, vol. 14, no. 1-2, pp. 225–247, 2005.