Theoretical Computer Science 875 (2021) 28–51

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Exponential slowdown for larger populations: The $(\mu + 1)$-EA on monotone functions [☆]

Johannes Lengler [*,1], Xun Zou [*,1,2]

ETH Zürich, Zürich, Switzerland

ARTICLE INFO

Article history:
Received 30 November 2019
Received in revised form 19 March 2021
Accepted 23 March 2021
Available online 29 March 2021
Communicated by C. Witt

Keywords:
Evolutionary algorithm
Monotone functions
Population size
Mutation rate
Runtime analysis
Hottopic functions

ABSTRACT

Pseudo-Boolean monotone functions are unimodal functions which are trivial to optimize for some hillclimbers, but are challenging for a surprising number of evolutionary algorithms. A general trend is that evolutionary algorithms are efficient if parameters like the mutation rate are set conservatively, but may need exponential time otherwise. In particular, it was known that the $(1 + 1)$-EA and the $(1 + \lambda)$-EA can optimize every monotone function in pseudolinear time if the mutation rate is c/n for some $c < 1$, but that they need exponential time for some monotone functions for $c > 2.2$. The second part of the statement was also known for the $(\mu + 1)$-EA.

In this paper we show that the first statement does *not* apply to the $(\mu + 1)$-EA. More precisely, we prove that for every constant $c > 0$ there is a constant $\mu_0 \in \mathbb{N}$ such that the $(\mu + 1)$-EA with mutation rate c/n and population size $\mu_0 \leq \mu \leq n$ needs superpolynomial time to optimize some monotone functions. Thus, increasing the population size by just a constant has devastating effects on the performance. This is in stark contrast to many other benchmark functions on which increasing the population size either increases the performance significantly, or affects performance only mildly.

The reason why larger populations are harmful lies in the fact that larger populations may temporarily decrease the selective pressure on parts of the population. This allows unfavorable mutations to accumulate in single individuals and their descendants. If the population moves sufficiently fast through the search space, then such unfavorable descendants can become ancestors of future generations, and the bad mutations are preserved. Remarkably, this effect only occurs if the population renews itself sufficiently fast, which can only happen far away from the optimum. This is counter-intuitive since usually optimization becomes harder as we approach the optimum. Previous work missed the effect because it focused on monotone functions that are only deceptive close to the optimum.

[☆] A conference version of this paper was published at FOGA 2019.

[*] Corresponding authors.
E-mail addresses: johannes.lengler@inf.ethz.ch (J. Lengler), xun.zou@inf.ethz.ch (X. Zou).

[1] Both authors contributed equally to this research.

[2] The author was supported by the Swiss National Science Foundation [grant number CR-SII5_173721].

https://doi.org/10.1016/j.tcs.2021.03.025
0304-3975/© 2021 Elsevier B.V. All rights reserved.

1. Introduction

Population-based *evolutionary algorithms* (EAs) are general-purpose heuristics for optimization. Having a population may be helpful, because it allows for diversity in the algorithm's states. Such diversity may be helpful for escaping local minima, and it is a necessary ingredient for crossover operations as they are used in *genetic algorithms* (GAs). Theoretical and practical analyses of population-based algorithms have indeed mostly found positive or neutral effects, and showed a general trend that larger populations are better [1], or at least not worse than a population size of one [2]. The only (mild) observed negative effect is, intuitively speaking, that maintaining a population of size μ may slow down the optimization time by a factor of at most μ. Only few, highly artificial examples are known [3,4] in which a $(\mu+1)$-EA or $(\mu+1)$-GA with time budget μt performs significantly worse than a $(1+1)$-EA with time budget t. In this sense, it is easy to believe that a $(\mu+1)$ algorithm is at least as good as a $(1+1)$ algorithm, except for the runtime increase that comes from each individual only having probability $1/\mu$ per round of creating an offspring.

Our results challenge this belief, and show that it is highly wrong for some monotone functions. Our main results show that increasing μ from 1 to a larger constant can increase the runtime from quasilinear to exponential.

A monotone[3] pseudo-Boolean function is a function $f : \{0,1\}^n \to \mathbb{R}$ such that for every $x, y \in \{0,1\}^n$ with $x \neq y$ and $x_i \geq y_i$ for all $1 \leq i \leq n$ it holds $f(x) > f(y)$. Monotone functions are easy benchmark functions for optimization techniques, since they always have a unique local and global optimum at the all-ones string. Moreover, from every search point there are short, fitness-increasing paths to the optimum, by flipping zero-bits into one-bits. Consequently, there are many algorithms which can easily optimize every monotone function. A particular example is *random local search* (RLS), which is the $(1+1)$ algorithm that flips in each round exactly one bit, uniformly at random. RLS can never increase the distance from the optimum for a monotone function, and it optimizes any such function in time $O(n\log n)$ by a coupon collector argument. Thus monotone functions are regarded as an easy benchmark for evolutionary algorithms. Nevertheless it was shown in [5–8] that a surprising number of evolutionary algorithms need exponential time to optimize some monotone functions, especially if they mutate too aggressively, i.e., the mutation parameter c is too large (see Section 1.2 for a detailed discussion). However, in all considered cases the algorithms were efficient if the mutation parameter satisfied $c < 1$.

1.1. Our results

We show that the $(\mu+1)$-Evolutionary Algorithm, $(\mu+1)$-EA, becomes inefficient even if the mutation strength is smaller than 1. More precisely, we show that for every $c > 0$ there is a $\mu_0 = \mu_0(c) \in \mathbb{N}$ such that for all $\mu_0 \leq \mu \leq n$ there are some monotone functions for which the $(\mu+1)$-EA with mutation rate c/n needs superpolynomial time to find the optimum. If μ is $O(1)$ then this time is even exponential in n. Note that for $0 < c \leq 1$, it is known that the $(1+1)$-EA finds the optimum in quasilinear time for any monotone functions [8–10]. Thus when we increase the population size only slightly (from 1 to μ_0), the optimization time explodes, from quasilinear to exponential.

The monotone functions that are hard to optimize are due to Lengler and Steger [5], and were dubbed HotTopic functions in [6]. These functions look locally like linear functions in which all bits have some positive weights. However, in each region of the search space there is a specific subset of bits (the 'hot topic'), which have very large weights, while all other bits have only small weights. If an algorithm improves in the hot topic, then it will accept the offspring regardless of whether the other bits deteriorate. In [5,6,11] it was shown that an algorithm like the $(1+1)$-EA with mutation rate $c > 2.13..$ will mutate too many of these bits outside of the hot topic, and will thus not make progress towards the global optimum.

The key insight of our paper is that for such weighted linear functions with imbalanced weights, populations may also lead to an accumulation of bad mutations, even if the mutation rate is small. Here is the intuition. For a search point x, we call the number of one-bits in the hot topic in x the *rank* of x. Consider a $(\mu+1)$-EA close to the optimum, and assume for simplicity that all search points in the population S_0 have the same rank i. At some point one of them will improve in the hot topic by flipping a zero-bit there. Let us call the offspring x, and let us assume that its rank is $i+1$. Then x is fitter than all other search points in the population because it has a higher rank. Moreover, every offspring or descendant of x will also be fitter than all the other points in the population, as long as they maintain rank $i+1$. Thus for a while the $(\mu+1)$-EA will accept all (or most) descendants of x, and remove search points of rank i from the population. This goes on until some time t_0 at which search points of rank i are completely eliminated from the population. Note that at time t_0, most descendants x' of x have considerably smaller fitness than x, since the algorithm accepts every type of mutation outside of the hot topic, and most mutations are detrimental. If some descendant x' of x creates an offspring y of even higher rank, then y is accepted and the cycle repeats with y instead of x. The crucial point is that y is an offspring of x', which has accumulated a lot of bad mutations compared to x. So typically, x' is considerably less fit than x, but still it passes on its bad genes.

The above effect needs that the probability of improving in the hot topic has the right order. If the probability is too large (close to one), then x will already spawn an offspring of rank $i+1$ before it has spawned many descendants with the same rank. On the other hand, if the probability is too small then there will be no rank-improving mutations until time t_0, and after time t_0 the algorithm starts to remove the worst individuals of rank $i+1$ from the population. We remark that

[3] Following [5,6], we call them monotone functions, although *strictly monotone functions* would be slightly more accurate.

this latter regime was already studied in [6], for the extreme case in which the improvement probability is so small that typically the population of rank $i+1$ collapses into copies of x before a further improvement is made. (In the terminology of [6], it was the assumption that the parameter ε of the HotTopic function was sufficiently small.) However, there is a rather large range of improvement probabilities that lead to the aforementioned effect, i.e., they typically yield an offspring y from some inferior search point x' of rank $i+1$.

1.2. Related work

The analysis of EAs on monotone functions started in 2010 by the work of Doerr, Jansen, Sudholt, Winzen and Zarges [7,8]. Their contribution was twofold: firstly, they showed that the $(1+1)$-EA, which flips each bit independently with static mutation rate c/n, needs time $O(n \log n)$ on all monotone functions if the mutation parameter c is a constant strictly smaller than one. This result was already implicit in [9].

On the other hand, it was also shown in [7,8] that for large mutation rates, $c > 16$, there are monotone functions for which the $(1+1)$-EA needs exponential time. The construction of hard monotone functions in [7,8] was later simplified by Lengler and Steger [5], who improved the range for c from $c > 16$ to $c > c_0 = 2.13...$ Their construction was later called HotTopic functions in [6], and it will also be the basis for the results in this paper.

For a long time, it was an open question whether $c = 1$ is a threshold at which the runtime switches from polynomial to exponential. On the presumed threshold $c = 1$, a bound of $O(n^{3/2})$ was known due to Jansen [9], but it was unclear whether the runtime is quasilinear. Finally, Lengler, Martinsson and Steger [10] could show that $c = 1$ is not a threshold, showing by an information compression argument an $O(n \log^2 n)$ bound for all $c \in [1, 1+\varepsilon]$ for some $\varepsilon > 0$.

Recently, the limits of our understanding of monotone functions were pushed significantly by Lengler [6,11], who analyzed monotone functions for a manifold of other evolutionary and genetic algorithms. In particular, he analyzed the algorithms on HotTopic functions, and found sharp thresholds in the parameters, such that on one side of the threshold the runtime on HotTopic was $O(n \log n)$, while on the other side of the threshold it was exponential. These algorithms include the $(1+1)$-EA, the $(1+\lambda)$-EA, the $(\mu+1)$-EA, for which the threshold condition was $c < c_0$, where $c_0 = 2.13..$, and it further included the $(1+(\lambda,\lambda))$-GA, and the so-called 'fast $(1+1)$-EA' and 'fast $(1+\lambda)$-EA'.[4] Surprisingly, for the genetic algorithms $(\mu+1)$-GA and the 'fast $(\mu+1)$-GA', any parameter range leads to runtime $O(n \log n)$ on HotTopic if the population size μ is large enough, showing that crossover is strongly beneficial in these cases.

For some of the algorithms, Lengler in [6,11] also complemented the results on HotTopic functions by statements asserting that for less aggressive choices of the parameters the algorithms optimize *every* monotone function efficiently. For example, he proved that for mutation parameter $c < 1$ and for every constant $\lambda \in \mathbb{N}$, with high probability the $(1+\lambda)$-EA optimizes every monotone function in $O(n \log n)$ steps. Analogous statements were proven for the 'fast $(1+1)$-EA' and 'fast $(1+\lambda)$-EA', and for the $(1+(\lambda,\lambda))$-GA, but the condition $c < 1$ needs to be replaced by analogous conditions on the parameters of the respective algorithms. Moreover, in the case of the 'fast $(1+\lambda)$-EA', the result was only proven if the algorithm starts sufficiently close to the optimum. Lengler did not prove any results for general monotone functions for the population-based algorithms $(\mu+1)$-EA and $(\mu+1)$-GA, and for their 'fast' counterparts. Our result shows that at least for the $(\mu+1)$-EA, this gap had a good reason. As mentioned before, we will show that for every (constant) mutation parameter $c > 0$, there are monotone functions on which the $(\mu+1)$-EA needs superpolynomial time if the population size μ is larger than some constant $\mu_0 = \mu_0(c)$. It also shows that the $(\mu+1)$-EA and the $(1+\lambda)$-EA behave completely differently on the class of monotone functions, since the $(1+\lambda)$-EA is efficient for all constant λ whenever $c < 1$.

Surprisingly, our instance of a hard monotone function is again a HotTopic function. This may appear contradictory to the result in [6,11] that the $(\mu+1)$-EA is efficient on HotTopic functions if $c < c_0$. The reason why there is no contradiction is that all the results in [6,11] on HotTopic come with an important catch. The HotTopic functions come with several parameters, and we will give the formal definition and a more detailed discussion in Section 2.3. For now it suffices to know that one of the parameters, ε, essentially determines how close the algorithm needs to come to the optimum before the fitness function starts switching between different hot topics. In [6,11], only small values of ε were considered. More precisely, it was shown that for every $\mu \in \mathbb{N}$ there is an $\varepsilon_0 > 0$ such that the results for the $(\mu+1)$-EA hold for all HotTopic functions with parameter $\varepsilon \le \varepsilon_0$, and there were similar restrictions for other parameters of the HotTopic function. In a nutshell, *the effect of switching hot topics was only studied close to the optimum*. Arguably, this was a natural approach since usually the hardest region for optimization is close to the optimum. In this paper, we consider HotTopic functions in a different parameter regime: we study relatively large values of the parameter ε, which is a regime of the HotTopic functions in which the action happens far away from the optimum. *Consequently, the results from [6,11] on the $(\mu+1)$-EA on* HotTopic *do not carry over to the version of* HotTopic *functions that we consider in this paper.* We stress this point to resolve the apparent contradiction between our results and the results in [6,11].

The above discussion also shows a rather uncommon phenomenon. Consider a small mutation parameter, e.g., $c = 1/2$. Our results show that the $(\mu+1)$-EA fails to make progress if the HotTopic function starts switching hot topics far away from the optimum. On the other hand, by the results in [6], the $(\mu+1)$-EA is not deceived if the HotTopic function starts

[4] The so-called "fast" versions draw the parameter c randomly in each iteration from a heavy-tailed distribution. This avoids that the probability of flipping k bits drops exponentially in k [12].

switching hot topics close to the optimum. Thus, we have found an example where optimization close to the optimum is easier than optimization far away from the optimum, quite the opposite of the usual behavior of algorithms. This strange effect occurs because the problem of the $(\mu+1)$-EA arises from having a non-trivial population. However, close to the optimum, progress is so hard that the population tends to degenerate into multiple copies of a single search point, which effectively decreases the population size to one and thus eliminates the problem (see also the discussion in Section 1.1 above).

Most other work on population-based algorithms has shown benefits of larger population sizes, especially when crossover is used [13–16]. Without crossover, the effect is often rather small [2]. The only exception in which a population has theoretically been proven to be severely disadvantageous is on Ignoble Trails. This rather specific function has been carefully designed to lead into a trap for crossover operators [3], and it is deceptive for $\mu = 2$ if crossover is used, but not for $\mu = 1$. Arguably, the HotTopic functions are also rather artificial, although they were not specifically designed to be deceptive for populations. However, regarding the larger and more natural framework of monotone functions, our results imply that a $(\mu+1)$-EA with mutation parameter $c = 1$ does not optimize all monotone functions efficiently if μ is too large, while the corresponding $(1+1)$-EA is efficient.

Moreover, Lengler and Schaller pointed out an interesting connection between HotTopic functions and a dynamic optimization problem in [17], which is arguably more natural. In that paper, the algorithm should optimize a linear function with positive weights, but the weights of the objective function are re-drawn each round (independently and identically distributed). This setting is similar to monotone functions, since a one-bit is always preferable over a zero-bit, and the all-one string is always the global optimum. However, the weight of each bit changes from round to round, which somewhat resembles that the HotTopic function switches between different hot topics as the algorithm progresses. In [17] the $(1+1)$-EA was studied, and the behavior in the dynamic setting is very similar to the behavior on HotTopic functions. It remains open whether the effects observed in our paper carry over to this dynamic setting.

2. Preliminaries and definitions

2.1. Notation

Throughout the paper we will assume that $f : \{0,1\}^n \to \mathbb{R}$ is a monotone function, i.e., for every $x, y \in \{0,1\}^n$ with $x \neq y$ and such that $x_i \geq y_i$ for all $1 \leq i \leq n$ it holds $f(x) > f(y)$. We will consider algorithms that try to maximize f, and we will mostly focus on the *runtime* of an algorithm, which we define as the number of function evaluations before the first evaluation of the global maximum of f.

For $n \in \mathbb{N}$, we denote $[n] := \{1, \ldots, n\}$. For a search point x, we write $\textsc{Om}(x)$ for the $\textsc{OneMax}$-value of x, i.e., the number of one-bits in x. For $x \in \{0,1\}^n$ and $\emptyset \neq I \subseteq [n]$, we denote by $d(I, x) := |\{i \in I \mid x_i = 0\}|/|I|$ the *density* of zero-bits in I. In particular, $d([n], x) = 1 - \textsc{Om}(x)/n$. Landau notation like $O(n), o(n), \ldots$ is with respect to $n \to \infty$. An event $\mathcal{E} = \mathcal{E}(n)$ holds *with high probability* or *whp* if $\Pr[\mathcal{E}(n)] \to 1$ for $n \to \infty$. A function $f : \mathbb{N} \to \mathbb{R}$ grows *stretched-exponentially* if there is $\delta > 0$ such that $f(x) = \exp\{\Omega(n^\delta)\}$, and it grows *quasilinearly* if there is $C > 0$ such that $f(x) = O(n \log^C n)$.

Throughout the paper we will use n for the dimension of the search space, μ for the population size, and c for the mutation parameter. We will always assume that the mutation parameter c is a constant independent of n, but the population size $\mu = \mu(n)$ may depend on n.

2.2. Algorithm

We will consider the $(\mu+1)$-EA with population size $\mu \in \mathbb{N}$ and mutation parameter $c > 0$ for maximizing a pseudo-boolean fitness function $f : \{0,1\}^n \to \mathbb{R}$. This algorithm maintains a population of μ search points. In each round, it picks one of these search points uniformly at random, the *parent* x^t for this round. From this parent it creates an *offspring* y^t by flipping each bit of x^t independently with probability c/n, and adds it to the population. From the $\mu + 1$ search points, it then discards the one with lowest fitness from the population, breaking ties randomly.[5]

2.3. HotTopic functions

In this section we give the construction of hard monotone functions by Lengler and Steger [5], following closely the exposition in [6]. The functions come with five parameters $n \in \mathbb{N}$, $0 < \beta < \alpha < 1$, $0 < \varepsilon < 1$, and $L \in \mathbb{N}$, and they are given by a randomized construction. Following [6], we call the corresponding function $\textsc{HotTopic}_{n,\alpha,\beta,\varepsilon,L} = \mathrm{HT}_{n,\alpha,\beta,\varepsilon,L} = \mathrm{HT}$.

For $1 \leq i \leq L$ we choose sets $A_i \subseteq [n]$ of size αn independently and uniformly at random, and we choose subsets $B_i \subseteq A_i$ of size βn uniformly at random. We define the *level* $\ell(x)$ of a search point $x \in \{0,1\}^n$ by

$$\ell(x) := \max\left\{\ell' \in [L] : d(B_{\ell'}, x) \leq \varepsilon\right\}, \tag{1}$$

[5] We break ties randomly for simplicity. Other selection schemes may give preference to offspring, or generally to more recent search points in case of ties. However, the tie-breaking scheme does not have an impact on our analysis.

Algorithm 1: The $(\mu+1)$-EA with mutation parameter c for maximizing an unknown fitness function $f : \{0,1\}^n \to \mathbb{R}$. The population S is a multiset, i.e., it may contain some search points several times.

```
Initialization:
    $S_0 \leftarrow \emptyset$;
    for $i = 1, \dots, \mu$ do
        Sample $x^{(0,i)}$ uniformly at random from $\{0,1\}^n$;
        $S_0 \leftarrow S_0 \cup \{x^{(0,i)}\}$;
Optimization:
    for $t = 1, 2, 3, \dots$ do
        Mutation:
        Choose $x^t \in S_{t-1}$ uniformly at random;
        Create $y^t$ by flipping each bit in $x^t$ independently with probability $c/n$;
        Selection:
        Set $S_t \leftarrow S_{t-1} \cup \{y^t\}$;
        Select $x \in \arg\min\{f(x) \mid x \in S_t\}$ (break ties randomly) and update $S_t \leftarrow S_t \setminus \{x\}$;
```

where we set $\ell(x) = 0$, if no such ℓ' exists. Then we define $f : \{0,1\}^n \to \mathbb{R}$ as follows:

$$\mathrm{HT}(x) := \ell(x) \cdot n^2 + \sum_{i \in A_{\ell(x)+1}} x_i \cdot n + \sum_{i \in R_{\ell(x)+1}} x_i, \tag{2}$$

where $R_{\ell(x)+1} := [n] \setminus A_{\ell(x)+1}$, and where we set $A_{L+1} := B_{L+1} := \emptyset$. One easily checks that this function is monotone [6].

So the set $A_{\ell+1}$ defines the hot topic while the algorithm is at level ℓ, where the level is determined by the sets B_i. Following up on the discussion in the introduction, observe that the level ℓ increases if the density of zero-bits in $B_{\ell'}$ drops below ε for some $\ell' > \ell$. From the analysis we will see that with high probability this only happens if the density of zero-bits in $A_{\ell+1}$ and in the whole string is also roughly ε, up to some constant factors. Hence, the parameter ε determines how far away the algorithm is from the optimum when the level changes.

Throughout the paper we will assume that α and β are independent of n, whereas we will choose small constants $\eta, \rho > 0$ and set $\varepsilon = \mu^{-1+\eta}$ and $L = \exp\{\rho\varepsilon n/\log^2\mu\}$, i.e., ε and L may depend of n, since we also allow μ to depend on n.[6]

2.4. Tools

To obtain good tail bounds, we often apply Chernoff's inequality.

Theorem 1 *(Chernoff bound [18]). Let $Y_1, \dots, Y_m$ be independent random variables (not necessarily i.i.d.) that take values in $[0,1]$. Let $S := \sum_{i=1}^m Y_i$, then for all $0 \le \delta \le 1$,*

$$\Pr[S \le (1-\delta)\,\mathbb{E}[S]] \le e^{-\delta^2\,\mathbb{E}[S]/2}$$

and for all $\delta \ge 0$,

$$\Pr[S \ge (1+\delta)\,\mathbb{E}[S]] \le e^{-\min\{\delta^2,\delta\}\,\mathbb{E}[S]/3}.$$

Finally, for all $k \ge 2e\,\mathbb{E}[S]$,

$$\Pr[S \ge k] \le 2^{-k}.$$

In addition, we will need the following theorem to bound the sum of geometrically distributed random variables.

Theorem 2 *(Theorem 1 in [19]). Let Y_j, $1 \le j \le m$, be independent random variables following the geometric distribution with success probability p_j, and let $S := \sum_{j=1}^m Y_j$. If $\sum_{j=1}^m p_j^{-2} \le s < \infty$ then for any $\delta > 0$,*

$$\Pr[S \le \mathbb{E}[S] - \delta] \le \exp\left(-\frac{\delta^2}{2s}\right).$$

For $h := \min\{p_j \mid j \in [m]\}$,

$$\Pr[S \ge \mathbb{E}[S] + \delta] \le \exp\left(-\frac{\delta}{4}\min\left\{\frac{\delta}{s}, h\right\}\right).$$

[6] In the papers [5,6,11] the parameter L was replaced by a constant parameter ρ such that $L = e^{\rho n}$. This had the advantage that their parameters were all independent of n, but since our parameters depend on n anyway, it is more convenient to use the parameter L. However, both versions are equivalent.

The following lemma estimates useful probabilities, e.g. the probability to improve on the current hot topic.

Lemma 3. *Let $\alpha, c > 0$ be constants. Consider a set $A \subseteq [n]$ of size αn where n is large enough, and consider a search point $x \in \{0,1\}^n$.*

1. *The probability that the number of one-bits in A does not decrease after a standard bit mutation with rate c/n on x can be bounded from below by $p_R = e^{-\alpha c}/2$.*
2. *The probability that a standard bit mutation with rate c/n strictly increases the number of one-bits in A has a lower bound $p_L = \varepsilon(x)\alpha c e^{-\alpha c}/2$ and an upper bound $p_U = \varepsilon(x)\alpha c$, where $\varepsilon(x) = d(A,x)$.*
3. *Let $(1-\varepsilon')\alpha n \le i \le \alpha n$ where $0 < \varepsilon' < 1$ and $\varepsilon' n \ge 2ec$. Let $\mathrm{rk}(x) := |\{j \in A \mid x_j = 1\}|$ and let y be an offspring of x. If $\mathrm{rk}(x) < i$, then at least one of the following inequalities holds.*

$$\Pr[\mathrm{rk}(y) \ge i] \le 2^{-\varepsilon'\alpha n} \qquad \textit{or} \qquad \frac{\Pr[\mathrm{rk}(y) \ge i+1]}{\Pr[\mathrm{rk}(y) \ge i]} \le 2\varepsilon'\alpha c.$$

Proof of Lemma 3. We show the statements one by one.

1. One way of creating an offspring with the same number of one-bits in A is to flip no bits at all in A. This probability is $(1-c/n)^{\alpha n} = e^{-\alpha c} - O(1/n) \ge e^{-\alpha c}/2$ when n is large enough.
2. We observe that the probability we consider is at least

$$\begin{aligned}\Pr[\text{flip 1 zero-bit and 0 one-bits in } A] &= \varepsilon(x)\alpha n \cdot \frac{c}{n}\left(1-\frac{c}{n}\right)^{\alpha n-1}\\ &= \varepsilon(x)\alpha c\left(e^{-\alpha c} - O\left(\frac{1}{n}\right)\right)\\ &\ge \frac{1}{2}\varepsilon(x)\alpha c e^{-\alpha c}.\end{aligned}$$

And it is at most

$$\Pr[\text{flip at least 1 zero-bit}] \le \sum_{i=1}^{\varepsilon(x)\alpha n} \Pr[\text{flip the } i\text{-th zero-bit}] = \varepsilon(x)\alpha c,$$

where the second inequality follows from a union bound over all zero-bits in A.
3. Assume first that $\mathrm{rk}(x) < (1-2\varepsilon')\alpha n$. Then for $\mathrm{rk}(y) \ge i$, at least $\varepsilon'\alpha n$ zero-bits must be flipped in one mutation. The expected number of flipped zero-bits is at most $\alpha n \cdot c/n = \alpha c$, so that happens with probability $2^{-\varepsilon'\alpha n}$ by the Chernoff bound. So let us consider the other case, $\mathrm{rk}(x) \ge (1-2\varepsilon')\alpha n$. Let P be a permutation on the αn bits in A such that $P(j) < P(j')$ for all $x_j = 1$ and $x_{j'} = 0$. Consider mutating the bits in x in the permuted order, and we track the number $G := G_0 - G_1$ during that process, where G_0 (G_1) is the number of flipped zero-bits (one-bits). Clearly, G will be decreasing while we are at the one-bits and increasing afterwards. Then $\mathrm{rk}(y) \ge i$ if and only if $G \ge i - \mathrm{rk}(x)$ after flipping some zero-bit j, and $\mathrm{rk}(y) \ge i+1$ if and only if at least one more zero-bit is flipped after bit j. The number of remaining zero-bits is at most $\alpha n - \mathrm{rk}(x) - 1 < 2\varepsilon'\alpha n$, so the probability of flipping at least one remaining zero-bit is at most $2\varepsilon'\alpha c$ by a union bound. Therefore,

$$\begin{aligned}\Pr[\mathrm{rk}(y) \ge i+1] &\le 2\varepsilon'\alpha c \cdot \Pr[G \ge i - \mathrm{rk}(x) \text{ at some zero-bit } j]\\ &= 2\varepsilon'\alpha c \cdot \Pr[\mathrm{rk}(y) \ge i]. \quad \square\end{aligned}$$

We will use the following two theorems to bound the running time of the $(\mu+1)$-EA. The first one states that a sequence of random variables whose differences are small with exponentially decaying tail bound are *sub-Gaussian*.[7]

Theorem 4 *(Timo Kötzing, Theorem 10 in [20]). Let $(Y_i)_{i\ge 0}$ be a supermartingale such that there are $c' > 0$ and δ' with $0 < \delta' < 1$ and, for all $i \ge 0$ and for all $y \ge 0$,*

$$\Pr[|Y_{i+1} - Y_i| \ge y \mid Y_0, \dots, Y_i] \le c'(1+\delta')^{-y}.$$

Then $(Y_i)_{i\ge 0}$ is $(128c'\delta'^{-3}, \delta'/4)$-sub-Gaussian.

[7] The reader can take the concept of being sub-Gaussian as a black box. Theorem 4 asserts that exponential tail bounds guarantee the property, Theorem 5 describes the consequences. For completeness, we also give the definition: a sequence of random variables $(Y_i)_{i\ge 0}$ is (c,δ)-sub-Gaussian if and only if $\mathbb{E}[\exp(z(Y_{i+1} - Y_i)) \mid Y_0, \dots, Y_i] \le \exp(z^2 c/2)$ holds for all $i \ge 0$ and $z \in [0, \delta]$.

The other theorem bounds first hitting times of sub-Gaussian supermartingales.

Theorem 5 *(Timo Kötzing, Theorem 12 in [20]). Let $(Y_i)_{i\geq 0}$ be a sequence of random variables and let $r \in \mathbb{R}$. If, for all $i \geq 0$,*

$$\mathbb{E}[Y_{i+1} - Y_i \mid Y_0, \dots, Y_i] \leq r,$$

then $(Y_i - ri)_{i\geq 0}$ is a supermartingale. If further $(Y_i - ri)_{i\geq 0}$ is (c'', δ'')-sub-Gaussian, then, for all $i \geq 0$ and all $y > 0$,

$$\Pr\Big[\max_{0\leq j\leq i}(Y_j - Y_0) \geq ri + y\Big] \leq \exp\Big(-\frac{y}{2}\min\Big(\delta'', \frac{y}{c''i}\Big)\Big).$$

3. Formal statement of the result

The main result of this paper is the following.

Theorem 6. *For every constant $c > 0$ and $0 < \beta < \alpha < 1$ there exist constants $\mu_0 = \mu_0(c) \in \mathbb{N}$ and $\eta, \rho > 0$ such that the following holds for all $\mu_0 \leq \mu \leq n$ where n is sufficiently large. Consider the $(\mu + 1)$-EA with population size μ and mutation rate c/n on the n-bit* HotTopic *function $\text{HT}_{n,\alpha,\beta,\varepsilon,L}$, where $\varepsilon = \mu^{-1+\eta}$ and $L = \lfloor\exp\{\rho\varepsilon n/\log^2\mu\}\rfloor$. Then with high probability the $(\mu+1)$-EA visits every level of the* HT *function at least once. In particular, it needs at least L steps to find the optimum, with high probability and in expectation.*

That is, if $\mu \geq \mu_0$ is a constant (independent of n) then with high probability the optimization time is exponential.

We remark that the requirement $\mu \leq n$ is not tight, and we conjecture that the runtime is always superpolynomial for $\mu \geq \mu_0$, also for much larger values of μ. However, we did not undertake big efforts to extend the range of μ since we do not feel that it adds much to the statement. For larger values of μ, e.g., $\mu = n^2$, our proof does not go through unmodified. With our definition of $\varepsilon = \mu^{-1+\eta}$, we only get error probabilities of the form $\exp\{-\Omega(\varepsilon n/\log^2\mu)\}$, which are not $o(1)$ if e.g. $\mu = n^2$. Hence we would need to choose larger values of ε, and then we lose a very convenient property, namely that for every fixed i, with high probability no individual of rank at most $i-1$ creates an individual of rank at least $i+1$. To avoid these complications, we only consider $\mu \leq n$.

4. Proof overview

The next three sections are devoted to proving Theorem 6. The key ingredient is to analyze the drift of the density $d([n], x)$ for search points x which have roughly density ε. We start by giving an informal overview, and by discussing similarities and differences to the situation in [5] and [6].

We will analyze the algorithm in the regime where the fittest search point x^* in the population satisfies

$$d(A_{\ell+1}, x^*) \in [\varepsilon/2, 2\varepsilon] \quad \text{and} \quad d(R_{\ell+1}, x^*) \in [\varepsilon/2, 2\varepsilon], \tag{3}$$

where $\ell = \ell(x^*)$ is the current level and $\varepsilon = \mu^{-1+\eta}$ is the parameter of the HotTopic function. It will turn out that for large μ, the algorithm already needs stretched-exponential time to escape this situation.

The main idea is similar to [5,6], in which the $(1+1)$-EA and other algorithms were analyzed. We first sketch the main argument for the $(1+1)$-EA, and explain afterwards which parts must be replaced by new arguments. The crucial ingredient is that while the density $d(A_{\ell+1}, x)$ of zero-bits on the hot topic decreases from 2ε to ε, the total density $d([n], x)$ has a positive drift, i.e., a drift away from the optimum. Moreover, the probability to change k bits in one step has a tail that decays exponentially with k. Therefore, it was shown that with high probability $d([n], x)$ stays above $\varepsilon + \gamma$ for an exponential number of steps, where γ is a small constant. Then it was argued that as long as $d([n], x)$ stays bounded away from ε, it is exponentially unlikely that the level ever increases by more than one. Since there are an exponential number of levels, this implies an exponential runtime.

The analysis of $(\mu+1)$-EA and $(\mu+1)$-GA for constant μ in [6] was obtained by reducing it to the analysis of a related $(1+1)$ algorithm. This was possible since the choice of parameters in [6] (choosing the parameter $\varepsilon = \varepsilon(\mu)$ sufficiently small) made the algorithm operate close to the optimum. In this range, there are only few zero-bits, and thus it is rather unlikely that a mutation improves the fitness. On the other hand, there is always a constant probability (if μ is constant) to create a copy of the fittest individual. In such a situation, the population degenerates frequently into a collection of copies of a single search point. Thus, the population-based algorithms behave similarly to a $(1+1)$ algorithm. This $(1+1)$ algorithm has essentially the same mutation parameter as the $(\mu+1)$-EA, while for the $(\mu+1)$-GA it has a much smaller mutation parameter (less than one), which is the reason why the $(\mu+1)$-GA is efficient on all HotTopic instances with small parameter ε. For us, the situation is more complex since we consider larger values of ε. As a consequence, it is easier to find a search point with better fitness, and the population does not collapse. Hence, it is not possible to represent the population by a single point.

Instead, we proceed as follows. Fix a fitness level ℓ, and consider the auxiliary fitness function

$$f_\ell(x) := n \sum_{j \in A_{\ell+1}} x_j + \sum_{j \in R_{\ell+1}} x_j. \tag{4}$$

We will first study the behavior of the $(\mu+1)$-EA on f_ℓ. Considering this fitness function is essentially the same as assuming that the level remains the same. We will see in the end that this assumption is justified, by the same arguments as in [5,6]. For a search point x, we define the *rank* $\mathrm{rk}(x) := \left|\{j \in A_{\ell+1} \mid x_j = 1\}\right|$ of x as the number of correct bits in the current hot topic. Note that by construction of f_ℓ, a search point with higher rank is always fitter than a search point with smaller rank.

Now we define $\mathcal{X}_i$ to be the set of search points of rank i that are visited by the $(\mu+1)$-EA, and we define Z_i to be the OneMax-value (the number of one-bits) of the last search point in $\mathcal{X}_i$ that the algorithm deletes from its population. Note that due to elitist selection, this search point is also (one of) the fittest search point(s) in $\mathcal{X}_i$ that the algorithm ever visits, and hence it has the largest OneMax-value among all search points in $\mathcal{X}_i$ that the algorithm ever visits. Then our goal is to show that $\mathbb{E}[Z_{i+1} - Z_i] = -\Omega(1)$, under the assumption that the population satisfies (3), i.e., that the density of the fittest search point is close to ε. This assumption can be justified by a coupling argument as in [5,6]. Computing the drift of Z_i is the heart of our proof, and the main technical contribution of this paper. In fact, to simplify the analysis we only prove the slightly weaker statement that $\mathbb{E}[Z_{i+K} - Z_i] = -\Omega(1)$ for a suitable constant K, which is equally suited. Once we have established this negative drift, the remainder of the proof as in [5,6] carries over almost unchanged.

To estimate the drift $\Delta := \mathbb{E}[Z_{i+K} - Z_i]$, we will assume for this exposition that $\mu = \omega(1)$, so that we may use O-notation. (In the formal proof we will use the weaker assumption $\mu \geq \mu_0$ for a sufficiently large constant $\mu_0 = \mu_0(c)$.) We distinguish between *good* and *bad* events. Good events will represent the typical situation; they will occur with high probability, and if they occur K times in a row, then it will deterministically follow that $Z_{i+K} - Z_i \leq -\log\mu$. On the other hand, bad events may lead to a positive difference, but they are unlikely and thus they contribute only a lower order term to the drift. We will discriminate two types of bad events. Firstly, we will show that the probability $\Pr[Z_{i+K} - Z_i > \lambda \log\mu]$ drops exponentially in λ. This implies that the events in which $Z_{i+K} - Z_i > \log^2\mu$ contribute at most a term $o(1)$ to the drift. Hence, we can restrict ourselves to the case that $Z_{i+K} - Z_i \leq \log^2\mu$. Now assume that we have any event of probability $o(\log^{-2}\mu)$. In the case $Z_{i+K} - Z_i \leq \log^2\mu$, this event can contribute at most a $o(1)$ term to the drift. Hence, we may declare any such event as a bad event, and conclude that all bad events together only contribute a $o(1)$ term to the drift.

As we have argued, we may neglect any event with probability $o(\log^{-2}\mu)$. This is a rather large error probability, which allows us to dub many events as 'bad', and to use rather coarse estimates on the error probability. We conclude this overview by describing how a good event, and thus a typical situation, looks like. In what follows, all claims hold with probability at least $1 - o(\log^{-2}\mu)$.

Let us call t_i the first round in which an individual of rank at least i is created, and T_i the round in which the last individual of rank at most i is eliminated. Then typically $T_i - t_i = O(\mu\log\mu) \cap \Omega(\mu)$. Let $|X_i| = |X_i(t)|$ denote the number of search points in the population of rank i at time t. We want to study the *family forest* F_i of $X_{\geq i}$, which is closely related to the family trees and family graphs that have been used in other work on population-based EAs, e.g. [1,2,21,22]. The vertices of this forest are all individuals of rank at least i that are ever included into the population. A vertex is called a *root* if its parent has rank less than i. Otherwise, the forest structure reflects the creation of the search points, i.e., vertex u is a child of vertex v if the individual u was created by a mutation of v.

As X_i grows, eventually the first few search points of rank $i+1$ are created, and form the first roots of the family forest. Then the forest starts growing, both because new roots may appear and because the vertices in the forest may create offspring. At some point we have $|X_{i+1}| = \mu^\delta$ for some (suitably small) $\delta > 0$. At this point, we still have typically $|X_i| = O(\mu^\delta/\varepsilon) = O(\mu^{1+\delta-\eta}) = o(\mu)$, where the latter holds if δ is small enough. Moreover, at this point there are no search points of rank strictly larger than $i+1$. The sets X_i and X_{i+1} both continue to grow with roughly the same speed until the search points of rank at most $i-1$ are eliminated from the population. Afterwards, the search points of rank i are eliminated from the population, until only search points of rank at least $i+1$ remain. Crucially, up to this point every search point of rank at least $i+1$ is accepted into the population. In other words, there is no selective pressure on the search points of rank $i+1$, and every mutation of a search point of rank $i+1$ enters the family tree, as long as the rank $i+1$ is preserved. Therefore, we can contain the family forest F_{i+1} of rank $i+1$ up to this point in a random forests F' which is obtained by certain forest growth processes in which no vertex is ever eliminated and all vertices continue to spawn offspring with a fixed rate.

We want to understand the set of individuals in X_{i+1} that spawn offspring in X_{i+2}, and thus spawn the roots for the family forest F_{i+2}. As before we can argue that no individuals of rank at least $i+2$ are created before the family forest of rank $i+1$ reaches size μ^δ. Moreover, we can show that the time T_{i+1} at which all individuals of rank $i+1$ are eliminated from the population satisfies $T_{i+1} - t_{i+1} \leq C\mu\log\mu$ for a suitable constant $C > 0$. Hence, F_{i+1} is bounded from above by the random forest F' at time $t_{i+1} + C\mu\log\mu$. This forest is only polynomially large in μ.

The recursive trees that we use to bound F_{i+1} are well understood, see also Fig. 1. In particular, it is known that even in F' only a small fraction μ^δ of the vertices are in depth at most $\phi\log\mu$, where $\delta, \phi > 0$ are suitable constants. Since each such vertex creates an offspring of strictly larger rank with probability ε/μ per round, the expected number of offspring of rank $i+2$ of these vertices is at most $O(\mu^\delta\varepsilon/\mu \cdot (T_{i+1} - t_{i+1}))$. With the right choice of parameters, this is $\mu^{-\Omega(1)}$, and we may conclude that no vertices of depth at most $\phi\log\mu$ create roots of rank $i+2$. On the other hand, since we do not

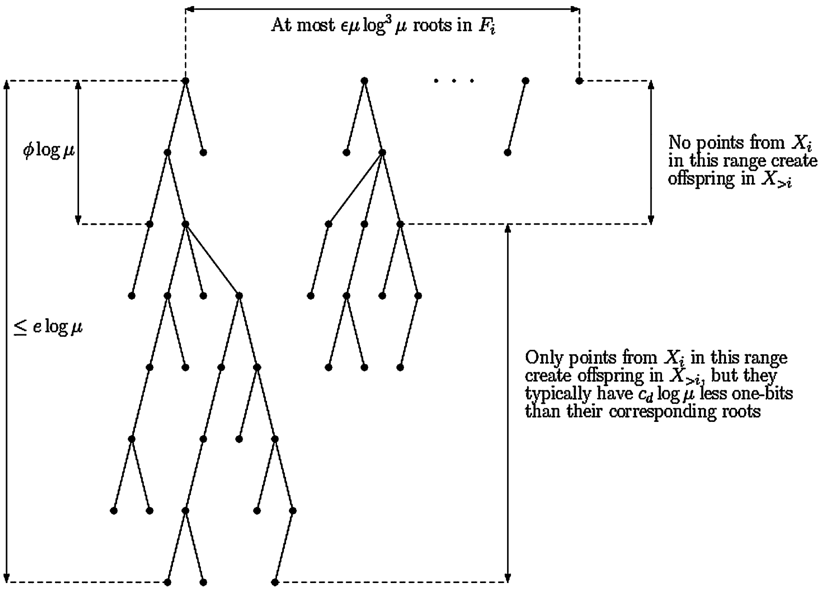
At most $\epsilon\mu\log^3\mu$ roots in F_i
$\phi\log\mu$
$\le e\log\mu$
No points from X_i in this range create offspring in $X_{>i}$
Only points from X_i in this range create offspring in $X_{>i}$, but they typically have $c_d\log\mu$ less one-bits than their corresponding roots

Fig. 1. A depiction of the family forest F_i, where ϕ and c_d are constants to be introduced in Section 5.3. The same picture also applies to its upper bound F'.

truncate any vertices in the creation of F', they are obtained from their parents by unbiased mutations of $[n] \setminus A_\ell$, and we can show that most (all but at most μ^δ) vertices of depth at least $\phi\log\mu$ in F' have accumulated $c'\log\mu$ more bad than good bit-flips when compared to their roots, for a suitable $c' > 0$. For the μ^δ exceptional vertices, none of them will create a root of rank $i+2$ in $T_{i+1} - t_{i+1}$ rounds, even if they are in F_{i+1}.

To summarize, good events consist of the following four main points. Firstly, no vertex of rank at most i creates an offspring of rank at least $i+2$. Secondly, every vertex in X_{i+1} that creates an offspring in X_{i+2} has at least depth $\phi\log\mu$ in the family forest. Thirdly, every vertex in X_{i+1} of depth at least $\phi\log\mu$ that creates an offspring in X_{i+2} has a OneMax value that is at least $c'\log\mu$ smaller than that of its root. Finally, we also require that no vertex in F' exceeds the OneMax value of its root by more than $C\log\mu$, for some $C > 0$. The complete list in the proof contains even more requirements, but these four already imply a decline in Z_i if they hold over K consecutive steps. In this case, inductively the OneMax values of all roots in F_{i+K} are at most $Z_i - Kc'\log\mu$. Moreover, Z_{i+K} exceeds the OneMax value of the corresponding root in X_{i+K} by at most $C\log\mu$, so we have $Z_{i+K} \le Z_i - Kc'\log\mu + C\log\mu$. Choosing K sufficiently large shows that Z_i must decrease in these typical situations.

5. Drift of Z_i

In this main section of the proof, we show that the random variable Z_i has negative drift. We will use the same notation as in the proof outline. In particular, X_i denotes the set of all search points of rank i that the algorithm visits, and Z_i denotes the OneMax-value of the last search point from X_i that the algorithm keeps in its population. If X_i is empty (which, as we will see, is very unlikely), then we set $Z_i := Z_{i-1}$. Moreover, we define $X_{\ge i} := \bigcup_{i' \ge i} X_{i'}$, and the definition of terms like $X_{>i}$ is analogous. For a given parent individual x, we denote by p_I (by p_R) the probability that an offspring of x has rank which is strictly larger than (at least as large as) the rank of x.

Throughout this section, we fix a level ℓ and consider the $(\mu+1)$-EA on the linear function f_ℓ defined in (4). In this section, we will study the case that $i \in [(1-2\varepsilon)\alpha n, (1-\varepsilon/2)\alpha n]$, where $\varepsilon = \mu^{-1+\eta}$. Note that this is a weaker form of Condition (3), i.e., we consider search points for which the density in A is close to ε.

5.1. Preliminaries

In this section we first give bounds on the time that the set $X_{\ge i}$ needs to grow from size 1 to size μ^κ, and we will conclude that $X_{\ge i}$ is large at the latter point in time. We start by bounding the time.

Lemma 7. *For all $0 < \alpha < 1$, $c > 0$, $0 < \eta < \kappa \le 1$, there exists a constant μ_0 such that the following holds for all $\mu_0 \le \mu \le n$. Let $i > (1-2\varepsilon)\alpha n$, where $\varepsilon = \mu^{-1+\eta}$. Consider the $(\mu+1)$-EA with mutation rate c/n on the linear function f_ℓ. Denote by $T_i^\kappa = T^\kappa$ the number of rounds until $|X_{\ge i}|$ reaches μ^κ after the algorithm visits the first point x^i in $X_{\ge i}$. With probability $1 - 2\mu^{-\Omega(1)}$,*

$$\frac{1}{2}(\kappa-\eta)\mu\log\mu \le T^\kappa \le 4\kappa e^{\alpha c}\mu\log\mu.$$

Moreover,

$$\mathbb{E}[T^\kappa] \le 3\kappa e^{\alpha c}\mu \log \mu.$$

Proof. By the definition of f_ℓ, all individuals in $X_{\ge i}$ are fitter than those in $X_{<i}$. So no points in $X_{\ge i}$ will be discarded until $X_{<i}$ becomes empty, and we are interested in the growth of $|X_{\ge i}|$ during this period. Let T_j be the time needed for $|X_{\ge i}|$ to grow from j to $j+1$. By definition we have $T^\kappa = \sum_{j=1}^{\mu^\kappa - 1} T_j$. Denote by x^t the point selected as parent by the algorithm in round t and denote by y^t its offspring. The probability that both x^t and y^t belong to $X_{\ge i}$ is at least $p_j = j/\mu \cdot p_R$, where j is the size of $X_{\ge i}$ at the beginning of round t and $p_R = e^{-\alpha c}/2$ is defined in Lemma 3.1. It is clear that we can dominate T_j by random variable $\bar{T}_j$ that follows a geometric distribution with parameter p_j. By Lemma 1.8.8 in [23], T^κ is dominated by $\bar{T}^\kappa := \sum_{j=1}^{\mu^\kappa - 1} \bar{T}_j$. Next we apply Theorem 2 to bound $\bar{T}^\kappa$ from above.

The expectation of $\bar{T}^\kappa$ is

$$\mathbb{E}[\bar{T}^\kappa] = \sum_{j=1}^{\mu^\kappa - 1} \mathbb{E}[\bar{T}_j] \le 2e^{\alpha c}\mu \sum_{j=1}^{\mu^\kappa} \frac{1}{j}.$$

For the Harmonic series, we have $\log(m+1) < \sum_{j=1}^{m} 1/j \le \log m + 1$, where log denotes the natural logarithm. Therefore, for large enough μ,

$$\mathbb{E}[T^\kappa] \le \mathbb{E}[\bar{T}^\kappa] \le 2e^{\alpha c}\mu(\log(\mu^\kappa) + 1) \le 3\kappa e^{\alpha c}\mu \log \mu. \tag{5}$$

Let $h := \min\{p_j \mid j = 1, \dots, \mu^\kappa - 1\}$, clearly $h = p_1 = e^{-\alpha c}/(2\mu)$. Let $s := \sum_{j=1}^{\mu^\kappa - 1} p_j^{-2}$, we have

$$s \le 4e^{2\alpha c}\mu^2 \sum_{j=1}^{\mu^\kappa} \frac{1}{j^2} \le \frac{2e^{2\alpha c}\pi^2}{3}\mu^2,$$

where the last step follows from $\sum_{j=1}^{\infty} 1/j^2 = \pi^2/6$. Given h and the bound on s, by Theorem 2 it holds for $\delta = \kappa e^{\alpha c}\mu \log \mu$ that

$$\Pr\left[\bar{T}^\kappa \ge \mathbb{E}[\bar{T}^\kappa] + \delta\right] \le e^{-\Omega(\log \mu)} = \mu^{-\Omega(1)}.$$

Since $T^\kappa \preceq \bar{T}^\kappa$, together with equation (5) we conclude that $T^\kappa \le 4\kappa e^{\alpha c}\mu \log \mu$ with probability $1 - \mu^{-\Omega(1)}$.

We still need a lower bound of T^κ. Consider the probability that $X_{\ge i}$ gets a new offspring y^t in a round where $|X_{\ge i}| = j$:

$$\begin{aligned}\Pr\left[y^t \in X_{\ge i}\right] &= \Pr\left[x^t \notin X_{\ge i} \wedge y^t \in X_{\ge i}\right] + \Pr\left[x^t \in X_{\ge i} \wedge y^t \in X_{\ge i}\right] \\ &\le (\mu - j)/\mu \cdot p_U + j/\mu \cdot 1 \le j/\mu + p_U,\end{aligned}$$

where p_U is defined in Lemma 3.2. Let $p'_j = j/\mu + p_U$, similarly as for the upper bound on T^κ, we can subdominate T^κ with a random variable $\hat{T}^\kappa = \sum_{j=1}^{\mu^\kappa - 1} \hat{T}_j$ (Lemma 1.8.8 in [23]), where the $\hat{T}_j$ are independent and geometrically distributed with parameter p'_j, respectively. Then

$$\begin{aligned}\mathbb{E}[\hat{T}^\kappa] &\ge \sum_{j=1}^{\mu^\kappa - 1} \frac{1}{p'_j} = \sum_{j=1}^{\mu^\kappa - 1} \frac{\mu}{j + \mu p_U} \ge \sum_{j=1}^{\mu^\kappa - 1} \frac{\mu}{j + \lceil \mu p_U \rceil} = \sum_{j=1}^{\mu^\kappa - 1 + \lceil \mu p_U \rceil} \frac{\mu}{j} - \sum_{j'=1}^{\lceil \mu p_U \rceil} \frac{\mu}{j'} \\ &> \mu \log\left(\mu^\kappa + \lceil \mu p_U \rceil\right) - \mu\left(\log \lceil \mu p_U \rceil + 1\right).\end{aligned}$$

Since $i \ge (1 - 2\varepsilon)\alpha n$, $P_U = O(\varepsilon) = O(\mu^{-1+\eta})$ for $0 < \eta < \kappa$. So $\lceil \mu p_U \rceil = O(\mu^\eta)$. Hence,

$$\mathbb{E}[T^\kappa] \ge \mathbb{E}[\hat{T}^\kappa] \ge \left(1 - O\left(\log^{-1} \mu\right)\right)(\kappa - \eta)\mu \log \mu.$$

Let $s' := \sum_{j=1}^{\mu^\kappa} p'^{-2}_j$. As $p'_j > p_j$, it holds $s' < s$ that. Applying Theorem 2 with s' and $\delta' = \varepsilon' \mu \log \mu$, we obtain

$$\Pr\left[\hat{T}^\kappa \le \mathbb{E}[\hat{T}^\kappa] - \delta'\right] \le e^{-\Omega(\log^2 \mu)} = \mu^{-\Omega(1)}.$$

Similarly, we have $\hat{T}^\kappa \preceq T^\kappa$, by picking a sufficiently small ε' we conclude that

$$T^\kappa \ge \frac{1}{2}(\kappa - \eta)\mu \log \mu$$

with probability $1 - \mu^{-\Omega(1)}$. □

In the following lemma, we give a lower bound on $|X_{\geq i+1}|$ when $X_{\geq i}$ reaches a certain size.

Lemma 8. *Let $\alpha, \kappa \in (0,1)$, $c > 0$, $\eta < 1$ be constants such that $\kappa > 1 - \eta/2$. Consider the $(\mu+1)$-EA with $\mu \leq n$ and mutation rate c/n on the linear function f_ℓ. Let $\varepsilon = \mu^{-1+\eta}$ and let $i \leq (1-\varepsilon/2)\alpha n$. Denote by $Y^\kappa_{i+1} = Y^\kappa$ the size of $X_{\geq i+1}$ when $|X_{\geq i}|$ reaches μ^κ. Then with probability $1 - \exp\big(-\Omega\big(\mu^{2(\kappa-1)+\eta}\big)\big)$,*

$$Y^\kappa = \Omega\big(\varepsilon\mu^{2\kappa-1}\big) = \Omega\big(\mu^{2(\kappa-1)+\eta}\big) = \mu^{\Omega(1)}.$$

Proof. Note that we may assume that $\mu \geq \mu_0$ for a constant μ_0 of our choice, since otherwise the probability may be zero and thus the statement is vacuous. In each round $|X_{\geq i}|$ increases by either 0 or 1, so after $X_{\geq i}$ reaches size $R := \lfloor \mu^\kappa/2 \rfloor$ there are at least R more rounds until $|X_{\geq i}| = \mu^\kappa$. In each of the remaining R rounds, the probability of a parent $x \in X_{\geq i}$ being selected and its offspring y belonging to $X_{>i}$ is at least

$$\Pr[y \in X_{>i}] > \Pr[x \in X_{\geq i} \wedge y \in X_{>i}] \geq R/\mu \cdot p_L,$$

where P_L is defined in Lemma 3. Let Y_j be independent Bernoulli variables with parameters $R/\mu \cdot p_L$ for $j \in [R]$. Then Y^κ dominates the sum of Y_j, i.e. $Y^\kappa \succeq \bar{Y}^\kappa := \sum_{j=1}^{R} Y_j$. It holds that

$$\mathbb{E}[\bar{Y}^\kappa] = R \cdot R/\mu \cdot p_L = \Theta\big(\varepsilon\mu^{2\kappa-1}\big) = \Theta\big(\mu^{2(\kappa-1)+\eta}\big).$$

By Chernoff's inequality (Theorem 1), we have for any constant $0 < \delta < 1$,

$$\Pr\big[\bar{Y}^\kappa < (1-\delta)\,\mathbb{E}[\bar{Y}^\kappa]\big] \leq \exp\big(-\Omega\big(\mu^{2(\kappa-1)+\eta}\big)\big).$$

The claim follows from $Y^\kappa \succeq \bar{Y}^\kappa$. □

5.2. Tail bounds

In this section, we will give rather loose tail bounds to show that it is unlikely that Z_i is much larger than Z_{i-1}. All constants in this section are independent of μ. This includes all hidden constants in the O-notation.

5.2.1. Tail bound on the lifetime of X_i

As before, let t_i be the first round in which an individual of rank at least i is created, and let T_i be the round in which the last individual of rank at most i is eliminated.

Lemma 9. *For all $0 < \alpha, \eta < 1$, $c > 0$, there is a constant $\mu_0 \in \mathbb{N}$ such that the following holds for all $\mu_0 \leq \mu \leq n$. Let $i \in [(1-2\varepsilon)\alpha n, (1-\varepsilon/2)\alpha n]$, where $\varepsilon = \mu^{-1+\eta}$. Consider the $(\mu+1)$-EA with mutation rate c/n on the linear function f_ℓ. Then with probability at least $1 - \mu^{-\Omega(1)}$, $T_i - t_i \leq 8e^{\alpha c}\mu\log\mu$. Moreover, for all $\beta \geq 1$ and $C = 16e^{\alpha c}$,*

$$\Pr[T_i - t_i \geq \beta \cdot C\mu\log\mu] \leq 2^{-\beta}.$$

Proof. We first show that $\Pr[T_i - t_i \geq C'\mu\log\mu] \leq 1/2$ for a suitable constant $C' > 0$. Let $x^{\geq i}$ be the first individual of rank at least i and let x^j with rank j be the first individual of rank strictly larger than i. We can divide the process from t_i to T_i into two parts. The first part ends when x^j is created, and we denote by t_j the round when this happens. The second part starts after t_j and ends when $X_{>i}$ reaches size μ. Since we are proving an upper bound of the tail, we can consider the second part ends when $X_{\geq j}$ reaches μ for simplicity.

If $x^{\geq i} = x^j$, then we have $t_j = t_i$, namely the first part does not exist. So for the tail bound of the first part, we may assume that $x^{\geq i} \in X_i$. By Lemma 7, for some $1 - \eta/2 < \kappa < 1$, we have $|X_{\geq i}| \geq \mu^\kappa$ at time $T := t_i + 4\kappa e^{\alpha c}\mu\log\mu$. By Lemma 8 we have $|X_{>i}| > 0$ at this point, so x_j must have been created before time T. For the second part, we apply Lemma 7 again for $X_{\geq j}$. By time $t_j + 4e^{\alpha c}\mu\log\mu$, $X_{\geq j}$ reaches size μ.

To summarize, we have applied Lemma 7 twice and Lemma 8 once. Therefore, with probability at least $1 - 5\mu^{-\Omega(1)}$, $T_i - t_i \leq 8e^{\alpha c}\mu\log\mu$. Since $\mu \geq \mu_0$, for large enough μ_0 we obtain $\Pr[T_i - t_i \geq C'\mu\log\mu] \leq 1/2$ for $C' = 8e^{\alpha c}$.

To conclude the proof, we set $C := 2C'$. Then for all integral $\beta' \in \mathbb{N}$ we consider β' phases and repeat the same argument. This shows $\Pr[T_i - t_i \geq \beta' \cdot C'\mu\log\mu] \leq 2^{-\beta'}$. Hence, for $C = 16e^{\alpha c}$ it holds for all $\beta \geq 1$,

$$\Pr[T_i - t_i \geq \beta \cdot C\mu\log\mu] \leq \Pr[T_i - t_i \geq \lceil\beta\rceil C'\mu\log\mu] \leq 2^{-\lceil\beta\rceil} \leq 2^{-\beta}. \quad \square$$

5.2.2. Family forests

From now on we will be mostly working on family forests, so we introduce the definition and several related lemmas here. The main idea is to couple the algorithm with a process that is not subject to selection. This idea has been used before to analyze population-based algorithms [1,2,21,22].

We denote the *family forest* for search points with rank at least i by F_i. The vertex set of F_i are the vertices in $X_{\geq i}$ that are (once) in the population, while the roots of the trees are vertices whose parents are in $X_{<i}$. Moreover, any path connecting a root and a vertex in F_i corresponds to a series of mutations that create this vertex. Note that the size of F_i increase over time.

As analyzing F_i directly can be complicated, we couple it with a simpler random forest F', which is generated by the following process. In round 0 there is a single root in F'. In each subsequent round, each vertex in F' creates a new child with probability $1/\mu$ and a new root is added. Lemma 10 shows that F_i can be coupled to a subgraph in F'.

Lemma 10. *The family forest F_i can be coupled to F' such that F' contains F_i as a subgraph at any round.*

Proof. Throughout the coupling process we maintain that F_i is a subgraph of F'. The first point $x^{\geq i}$ that the algorithm visits in $X_{\geq i}$ (in round t_i) corresponds to the only root r_0 in round 0 in F'. In every round $t > t_i$, a point x^t in the current population is selected to create an offspring y^t. For each $x \in F_i$, if $x^t = x$ (which happens with probability $1/\mu$ if x is still in the current population, and with probability zero otherwise) then we attach a child to x in F': if $y^t \in X_{\geq i}$ then we attach y^t to x in F', otherwise we attach a dummy child to x in F'. In this case, we still associate the offspring with the dummy child, and in our upcoming considerations we will ignore that this search point does belong to $X_{\geq i}$. If x^t is not in $X_{\geq i}$ while y^t is, we add y^t as a new root r_t to F', otherwise we add a new dummy root to F'. For every node $x \in F'$ that is a dummy node (that has no corresponding node in F) or whose copy in F has been removed from the population, we add another dummy node as its child with probability $1/\mu$. In this way, for each vertex in F' we create a new child with probability $1/\mu$ and a root is added in each round. On the other hand, by construction, F_i is a subgraph of F' at all times. □

Note that the search points associated with the vertices in F' are obtained from the root by mutation only, without any interfering selection step. This makes the process easy to analyze. Such a selection-free mutation process has been analyzed before, e.g. [24]. In Lemma 11 we show several useful properties of F'. Due to the coupling from F_i to F', the properties will also hold for F_i as well.

Lemma 11. *F' satisfies the following properties:*

1. *Let s_t denote the number of vertices in F' in round t, then $\Pr[s_t \geq S] \leq te^{t/\mu}/S$ for all $S > 0$.*
2. *Let x be a search point that corresponds to a vertex in F' of depth at most d with root y. Then for $k \geq 2edc$,*
$$\Pr[x \text{ and } y \text{ differ in more than } k \text{ bits}] \leq 2^{-k}.$$
3. *Let x be a search point that corresponds to a vertex in F' of depth larger than d with root y. If n is sufficiently large and $\textsc{Om}(y) \geq (1-8\varepsilon)n$ then*
$$\Pr[x \text{ has more one-bits than } y] \leq 2e^{-dc/32}$$
and
$$\Pr[y \text{ has less than } dc/16 \text{ more one-bits than } x] \leq 2e^{-dc/128}. \tag{6}$$
If n is sufficiently large and $\textsc{Om}(y) \leq (1-8\varepsilon)n$ then $\Pr[\textsc{Om}(x) \geq (1-4\varepsilon)n] \leq 2\cdot 2^{-\varepsilon n}$.
4. *Let s_t^d denote the number of vertices of depth d in round t for an arbitrary tree from F'. Then*
$$\mathbb{E}[s_t^d] \leq \frac{t^d}{d!\mu^d}.$$
In particular, for $t = O(\mu \log \mu)$ the depth of the tree is at most $e \log \mu$ with probability $1-\mu^{-\Omega(1)}$. Moreover, if $t \geq 2d\mu$, $\sum_{i=0}^{d} \mathbb{E}[s_t^i] \leq 2t^d/(d!\mu^d)$.

Proof. We prove the statements one by one.

1. In t rounds we have added t roots to the forest, and we will give a uniform bound for all of them. So we fix a root and denote by σ_τ the number of vertices in this tree in round τ, where $0 \leq \tau \leq t$. We assume pessimistically that the root is introduced in round 0. Then we have $\sigma_0 = 1$ and $\mathbb{E}[\sigma_{\tau+1} \mid \sigma_\tau] = (1+1/\mu)\sigma_\tau$ for $0 \leq \tau \leq t-1$. By linearity of expectation, we have $\mathbb{E}[\sigma_t] \leq (1+1/\mu)^t$. Since there are t roots, and using that $(1+1/\mu)^\mu \leq e$, we obtain

$$\mathbb{E}[s_t] \le t\,\mathbb{E}[\sigma_t] \le t(1+1/\mu)^t \le te^{t/\mu}.$$

By Markov's inequality, it holds that

$$\Pr[s_t \ge S] \le \frac{\mathbb{E}[s_t]}{S} \le \frac{te^{t/\mu}}{S}.$$

2. Let y_i be the i-th bit in y, the event $y_i \neq x_i$ implies that the i-th bit is flipped at least once. Denote by $d' \le d$ the distance between x and y. By a union bound

$$\begin{aligned}\Pr[y_i \neq x_i] &\le \Pr[\text{bit } i \text{ is flipped at least once}]\\ &\le d' \Pr[\text{bit } i \text{ is flipped in one mutation}] \le dc/n.\end{aligned}$$

Let $D = |\{i \in [n] \mid y_i \neq x_i\}|$ be the number of bits in which y and x differ. Then its expectation is $\mathbb{E}[D] \le dc$. Since the bits are modified independently, we can apply Chernoff's inequality (Theorem 1) for $k \ge 2edc \ge 2e\,\mathbb{E}[D]$, and obtain

$$\Pr[D \ge k] \le 2^{-k}.$$

3. Let the depth of x be $d' \ge d$. First we argue that we may assume $d' \le n/(16ec)$. If $d' \ge n/(16ec)$, then consider just the last $n/(16ec)$ steps. In these, every bit has a constant probability to be touched exactly once, and a constant probability not to be touched at all. If the number of one-bits before the last $n/(16ec)$ steps was at least $n/2$, then with probability $1 - e^{-\Omega(n)}$, x has at least $8\varepsilon n$ zero-bits as each one-bit has a constant probability of being flipped exactly once in those steps, and if the number of one-bits was at most $n/2$, x has also at least $8\varepsilon n$ zero-bits as each zero-bit has a constant probability of being untouched. In either case, x has more zero-bits than y with sufficiently large probability. So we may assume $d' \le n/(16ec)$.

We then consider the case $\textsc{Om}(y) \ge (1-8\varepsilon)n$. Let B_{01} be the number of bits flipped from 0 to 1. Then similarly as for Property 2 we bound $\mathbb{E}[B_{01}]$ by

$$\begin{aligned}&|\{i \mid y_i = 0\}| \cdot \Pr[x_i = 1 \mid y_i = 0]\\ &\le |\{i \mid y_i = 0\}| \cdot \Pr[\text{bit } i \text{ flipped at least once in } d' \text{ mutations}]\\ &\le 8\varepsilon n \cdot (d'c/n) = 8\varepsilon c d',\end{aligned}$$

where the second inequality follows from a union bound. Similarly, let B_{10} be the number of bits flipped from 1 to 0 in d' mutations, its expectation $\mathbb{E}[B_{10}]$ is

$$\begin{aligned}&|\{i \mid y_i = 1\}| \cdot \Pr[x_i = 0 \mid y_i = 1]\\ &\ge |\{i \mid y_i = 1\}| \cdot \Pr[\text{bit } i \text{ flipped exactly once in } d' \text{ mutations}]\\ &\ge \frac{n}{2}\binom{d'}{1}\frac{c}{n}\left(1-\frac{c}{n}\right)^{d'-1} \ge \frac{d'c}{2}\left(1-\frac{c}{n}d'\right) \ge \frac{d'c}{4}.\end{aligned}$$

Since all bits contribute independently, we may apply the Chernoff bound. With probability at least $1 - e^{-d'c/32}$ each, we have $B_{01} \le cd'/8$ and $B_{10} \ge cd'/8$. Both inequalities together imply that $\textsc{Om}(x) \le \textsc{Om}(y)$ as desired, and the probability that at least one of the inequalities is violated is at most $2e^{-d'c/32} \le 2e^{-dc/32}$.

Similarly, the probability that B_{01} (B_{10}) overshoot (undershoot) its expectation by more than $d'c/16$ is at most $e^{-d'c/128}$. Therefore, the probability that $B_{01} \ge B_{10} - d'c/16$ is at most $2e^{-d'c/128} \le 2e^{-dc/128}$.

For the second statement, assume $\textsc{Om}(y) \le (1-8\varepsilon)n$, and consider the first vertex x' on the path from y to x such that $\textsc{Om}(x') \ge (1-6\varepsilon)n$. The probability that more than εn bits were flipped in the creation of x' is at most $2^{-\varepsilon n}$ by the Chernoff bound, since by definition of x' the parent of x' has an $\textsc{Om}$-value smaller than $(1-6\varepsilon)n$, we may assume that $\textsc{Om}(x') \le (1-5\varepsilon)n$. Then, starting from x' we may use the same calculation as above, only that we need to bound the probability that εn more zero-bits than one-bits are flipped. This is bounded by the probability that $B_{01} \ge \varepsilon n$. Since $d' \le n/(16ec)$ we have $\varepsilon n \ge 16\varepsilon c d' \ge 2e\,\mathbb{E}[B_{01}]$, by the Chernoff bound, this probability is at most $2^{-\varepsilon n}$.

4. There can only be one root in a tree, so $s_t^0 = 1$ for all $t \ge 0$. For $d \ge 1$ and $t \ge 1$, it holds that

$$s_t^d = s_{t-1}^d + \sum_{i=1}^{s_{t-1}^{d-1}} Y_i,$$

where Y_i is an indicator variable that takes value 1 if the i-th vertex of depth $d-1$ creates a offspring in round t. By Wald's equation, we obtain

$$\mathbb{E}[s_t^d] = \mathbb{E}[s_{t-1}^d] + \frac{\mathbb{E}[s_{t-1}^{d-1}]}{\mu}.$$

Plugging in $\mathbb{E}[s_t^d] = 0$ for all $t < d$, we can derive that

$$\mathbb{E}[s_t^d] = \sum_{i=d-1}^{t-1} \frac{\mathbb{E}[s_i^{d-1}]}{\mu} \tag{7}$$

for all $t \geq d \geq 1$.

We show the result by induction. For $d = 1$, by equation (7) we have $\mathbb{E}[s_t^1] = t/\mu$ for all $t \geq 1$. Now assume that $\mathbb{E}[s_t^d] \leq t^d/(d!\mu^d)$ for all $t \geq d$ where $d \geq 1$, again by equation (7) it holds that

$$\mathbb{E}[s_t^{d+1}] \leq \sum_{i=d}^{t-1} \frac{i^d}{d!\mu^d}\frac{1}{\mu} = \frac{1}{d!\mu^{d+1}}\sum_{i=0}^{t-1} i^d \leq \frac{1}{d!\mu^{d+1}}\sum_{i=0}^{t-1} t^d = \frac{t^{d+1}}{(d+1)!\mu^{d+1}} \tag{8}$$

for all $t \geq d+1$.

Now consider $t = O(\mu\log\mu)$ and $d = k\log\mu$ for some constant $k > e$. With Stirling's approximation $d! = (1 + O(1/d))\sqrt{2\pi d}(d/e)^d$ and equation (8), we have

$$\mathbb{E}[s_t^d] = O\left(\frac{\mu^d(\log\mu)^d}{(d/e)^d\mu^d}\right) = O\left(\frac{(e\log\mu)^{k\log\mu}}{(k\log\mu)^{k\log\mu}}\right) = O\left(\frac{\mu^k}{\mu^{k\log k}}\right) = O\left(\mu^{k(1-\log k)}\right).$$

By Markov's inequality, $\Pr[s_t^d \geq 1] = O(\mu^{k(1-\log k)}\sqrt{\log\mu}) = \mu^{-\Omega(1)}$ as $k(1-\log k) < 0$. Therefore, with probability $1 - \mu^{-\Omega(1)}$, $s_t^d = 0$ for any $d > e\log\mu$, which implies that the depth of the tree is at most $e\log\mu$.

For the last statement, let $a_t^d := t^d/(d!\mu^d)$. If $t \geq 2d\mu$, $a_t^{d-1}/a_t^d = d\mu/t \leq 1/2$ for $d \geq 1$. Therefore,

$$\sum_{i=0}^{d}\mathbb{E}[s_t^i] \leq \sum_{i=0}^{d} a_t^i \leq \sum_{i=0}^{d} 2^{-(d-i)}a_t^d < 2a_t^d = \frac{2t^d}{d!\mu^d}. \quad \square$$

5.2.3. *Tail bound on steps of Z_i*

The first consequence of the coupling is an exponential tail bound on the difference $Z_i - Z_{i-1}$. Note that the tail bound only holds in one direction. There is no comparable tail bound for $Z_{i-1} - Z_i$, at least not without further knowledge on X_{i-1}: if there is a single search point $x \in X_{i-1}$ that has k more one-bits than all other search points in X_{i-1}, then x might not spawn an offspring and Z_i could drop by k or more, and k could be as large as $\Omega(n)$ without assumptions on X_{i-1}.

Lemma 12. *For all $0 < \alpha, \eta < 1$, $c > 0$ there is a constant $\mu_0 \in \mathbb{N}$ such that the following holds for all $\mu_0 \leq \mu \leq n$, where n is sufficiently large. Let $i \in [(1-2\varepsilon)\alpha n, (1-\varepsilon/2)\alpha n]$, where $\varepsilon = \mu^{-1+\eta}$. Assume that the $(\mu+1)$-EA with mutation rate c/n on the linear function f_ℓ satisfies $Z_{i-1} \geq (1-4\varepsilon)n$. Then for all $1 \leq \beta \leq \varepsilon n/\log^2\mu$ and $C_2 = 6400e^{\alpha c+1}$,*

$$\Pr[Z_i - Z_{i-1} \geq \beta \cdot C_2\log\mu] \leq 2^{-\beta}.$$

If on the other hand $Z_{i-1} < (1-4\varepsilon)n$, then $Z_i < (1-2\varepsilon)n$ with probability $1 - e^{-\Omega(\varepsilon n/\log^2\mu)}$.

Proof. By Lemma 9, there is $C = 16e^{\alpha c}$ such that for all $\beta \geq 1$,

$$\Pr[T_i - t_i \geq (\beta+2)\cdot C\mu\log\mu] \leq \tfrac{1}{4}2^{-\beta}.$$

By Lemma 11.1, at round $t = (\beta+2)\cdot C\mu\log\mu$ we have

$$\Pr[s_t \geq \mu^{2(\beta+2)C}] \leq \frac{t}{\mu^{2(\beta+2)C}} \leq (\beta+2)C\mu^{1-2(\beta+2)C}\log\mu < \tfrac{1}{4}2^{-\beta},$$

where the last step holds for all $\mu \geq \mu_0$ if μ_0 is sufficiently large. That is, the probability that the algorithm visits at least $\mu^{2(\beta+2)C}$ vertices in $X_{\geq i}$ is at most $\frac{1}{4}2^{-\beta}$.

From now on, we consider F' at a time when it has at most $\mu^{2(\beta+2)C}$ vertices. Let x be a search point that corresponds to a vertex in F' of depth at most $d = \beta C_2'\log\mu$ with root r, where $C_2' = 200C/c$. By Lemma 11.2, for $C_2 = 400eC$ it holds for large enough μ_0 that

$$\Pr[x \text{ and } r \text{ differ in more than } \beta C_2\log\mu \text{ bits}] \leq 2^{-\beta C_2\log\mu} \leq \tfrac{1}{4}2^{-\beta}\cdot\mu^{-2(\beta+2)C}.$$

By a union bound over all vertices in F', the probability that there exists such a vertex x among them is at most $\frac{1}{4}2^{-\beta}$.

Now let x be a search point that corresponds to a vertex in F' of depth larger than d with root r. For large enough μ_0 by Lemma 11.3, if $\mathrm{OM}(r) \geq (1-8\varepsilon)n$ then

$$\Pr[x \text{ has more one-bits than } r] \le 2e^{-dc/32} \le \tfrac{1}{8}2^{-\beta} \cdot \mu^{-2(\beta+2)C}.$$

The probability that there exists such a vertex x in F' is at most $\frac{1}{8}2^{-\beta}$ by a union bound. On the other hand, if n is sufficiently large and $\textsc{Om}(r) \le (1-8\varepsilon)n$ then for $\beta \le \varepsilon n/\log^2\mu$,

$$\Pr[\textsc{Om}(x) \ge (1-4\varepsilon)n] \le 2 \cdot 2^{-\varepsilon n} \le \tfrac{1}{8}2^{-\beta} \cdot \mu^{-2(\beta+2)C}.$$

Similarly, the probability that such a vertex x exists in F' is at most $\frac{1}{8}2^{-\beta}$.

To summarize, we have shown that each of the following four events happens with probability at least $1 - 1/4 \cdot 2^{-\beta}$.

- $\mathcal{E}_1$: $T_i - t_i < (\beta+2)C\mu\log\mu$.
- $\mathcal{E}_2$: $s_t < \mu^{2(\beta+2)C}$ at time $t = (\beta+2)C\mu\log\mu$.
- $\mathcal{E}_3$: Among the first $\mu^{2(\beta+2)C}$ vertices in F', there is no search point x with a distance at most $\beta C_2'\log\mu$ to its root r such that $|\{i \in [n] \mid r_i \neq x_i\}| > \beta C_2\log\mu$.
- $\mathcal{E}_4$: Among the first $\mu^{2(\beta+2)C}$ vertices in F', there is no search point x with a distance larger than $\beta C_2'\log\mu$ to its root r such that either $\textsc{Om}(r) \ge (1-8\varepsilon)n$ and $\textsc{Om}(x) > \textsc{Om}(r)$ or $\textsc{Om}(r) \le (1-8\varepsilon)n$ and $\textsc{Om}(x) \ge (1-4\varepsilon)n$.

Now we argue how the bounds for these events imply the lemma. By $\mathcal{E}_1$ and $\mathcal{E}_2$, we may restrict ourselves to the first $\mu^{2(\beta+2)C}$ vertices in F'. We claim that there are no offspring x in distance at most $\beta C_2'\log\mu - 1$ from their root r that have $\textsc{Om}$-value larger than $Z_{i-1} + \beta C_2\log\mu$. To see this, we add the parent of r, $r' \in X_{<i}$, and the edge between r' and r to F'. Now r' is the root of x and it can act as a reference point: by the definition of Z_{i-1} we have $Z_{i-1} \ge \textsc{Om}(r')$. If the distance from r' to x is at most $\beta C_2'\log\mu$, by $\mathcal{E}_3$ we have $\textsc{Om}(x) \le \textsc{Om}(r') + \beta C_2\log\mu$. If x is of larger distance from the added root r', we need to discriminate two cases. Either r' has $\textsc{Om}$-value at least $(1-8\varepsilon)n$ in which case $\textsc{Om}(x)$ do not exceed $\textsc{Om}(r')$ by the first part of $\mathcal{E}_4$. Or r' has $\textsc{Om}$-value at most $(1-8\varepsilon)n$, in which case x do not exceed a $\textsc{Om}$-value of $(1-4\varepsilon)n$ by the second part of $\mathcal{E}_4$. Therefore, if $Z_{i-1} \ge (1-4\varepsilon)n$, we can conclude that $\textsc{Om}$-value of x do not exceed Z_{i-1} in both cases. Hence, we have shown that $\textsc{Om}(x) - Z_{i-1} > \beta \cdot C_2\log\mu$ is only possible if at least one of the events $\mathcal{E}_1$ - $\mathcal{E}_4$ does not occur, and thus

$$\Pr[Z_i - Z_{i-1} > \beta \cdot C_2\log\mu] \le \sum_{j=1}^{4}(1 - \Pr[\mathcal{E}_j]) \le 2^{-\beta}.$$

If $Z_{i-1} < (1-4\varepsilon)n$, with the same arguments and letting $\beta \ge \varepsilon n/(\log^2\mu)$ we have $Z_i < (1-2\varepsilon)n$ with probability $1 - 2^{-\Omega(\varepsilon n/\log^2\mu)}$. □

5.3. Typical situations

As outlined in the overview, our analysis of the drift will be based on studying what happens in 'typical' situations. To characterize these, we use the following definition of 'good' events. Again we consider the $(\mu+1)$-EA on the linear function f_ℓ. For parameters $\phi, c_d, c_e > 0$ we define the event $\mathcal{E}_{\text{good}}(i) := \mathcal{E}_a \cap \mathcal{E}_b \cap \ldots \cap \mathcal{E}_e$, where $\mathcal{E}_a$ etc. are the following events about the family forest F_i of rank i. Recall the family forest consists of all $x \in X_{\ge i}$, and a vertex u is a child of v if u was created as an offspring of v. We will be concerned about those vertices in the family forest in X_i, i.e., vertices of rank exactly i.

- $\mathcal{E}_a$: No vertex in $X_{\le i-1}$ creates offspring in $X_{\ge i+1}$.
- $\mathcal{E}_b$: There are at most $\varepsilon\mu\log^3\mu$ roots in F_i.
- $\mathcal{E}_c$: No vertex in X_i of depth at most $\phi\log\mu$ in F_i creates offspring in $X_{>i}$.
- $\mathcal{E}_d$: For every vertex $x \in X_i$ that creates an offspring in $X_{\ge i+1}$, if the root r of x has $\textsc{Om}(r) \ge (1-8\varepsilon)n$ then $\textsc{Om}(x) \le \textsc{Om}(r) - c_d\log\mu$, and if $\textsc{Om}(r) \le (1-8\varepsilon)n$ then $\textsc{Om}(x) \le (1-4\varepsilon)n$. Moreover, the mutation changes at most $c_d/2 \cdot \log\mu$ bits.
- $\mathcal{E}_e$: No vertex in X_i has an $\textsc{Om}$-value which exceeds the $\textsc{Om}$-value of its root in F_i by more than $c_e\log\mu$.

Lemma 13. *For every $0 < \alpha < 1$, $c > 0$ there are $c_d, c_e > 0$ such that the following holds. For any constant parameters $0 < \phi < 1$ and $\eta > 0$ that satisfy the following conditions, where $g(\phi) = \phi(\log(8e^{\alpha c+1}) - \log\phi)$,*

$$\eta < \min\left\{g(\phi),\ \frac{1}{2} - g(\phi),\ \frac{c\phi}{128},\ \frac{c_d}{6}\right\}, \tag{9}$$

there exists μ_0 such that for all $\mu_0 \le \mu \le n$ and all $i \ge (1-8\varepsilon)\alpha n$, the $(\mu+1)$-EA on f_ℓ satisfies

$$\Pr\left[\mathcal{E}_{\text{good}}(i)\right] \ge 1 - O\left(\log^{-2}\mu\right).$$

We remark that $g(\phi) > 0$ for $0 < \phi < 1$ and $g(\phi) < 1/2$ for small enough ϕ, so there exists $\eta > 0$ that satisfies (9).

Proof. We need to show that $\Pr[\mathcal{E}] = 1 - O(\log^{-2}\mu)$ holds for $\mathcal{E} = \mathcal{E}_a, \dots, \mathcal{E}_e$. Thus we split the proof into five parts. Note that we actually show the stronger statement $\Pr[\mathcal{E}] = 1 - \mu^{-\Omega(1)}$ for $\mathcal{E} = \mathcal{E}_a, \mathcal{E}_c, \mathcal{E}_d, \mathcal{E}_e$.

$\mathcal{E}_a$: No vertex in $X_{\le i-1}$ creates offspring in $X_{\ge i+1}$ for $\eta < 1/2$.

We consider the number of offspring that are created from points in $X_{\le i-1}$ and are members of $X_{\ge i+1}$ after the first point x in $X_{\ge i}$ is created.

We first argue that the probability that $x \in X_i$ is $1 - O(\varepsilon)$. Since we assume $i \ge (1 - 8\varepsilon)\alpha n$ and the rank of x is at least i, the density of zero-bits is $d(A_{\ell+1}, x) \le 8\varepsilon$. By Lemma 3,

$$\frac{\Pr[x \in X_{\ge i+1}]}{\Pr[x \in X_{\ge i}]} \le 16\varepsilon\alpha c,$$

which implies $\Pr[x \in X_i \mid x \in X_{\ge i}] = 1 - O(\varepsilon) = 1 - O(\mu^{\eta-1})$.

By Lemma 9, after the first search point in X_i is created, with probability $1 - O(\mu^{-\Omega(1)})$ it takes at most $T = 8e^{\alpha c}\mu\log\mu$ rounds until the set $X_{\le i}$ is completely deleted. If a search point in $X_{\le i-1}$ creates an offspring in $X_{\ge i+1}$, at least 2 zero-bits need to be flipped. This probability is $O(\varepsilon^2)$ by a union bound, and hence the expected number of offspring in $X_{\ge i+1}$ created from $X_{\le i-1}$ is at most $O(\varepsilon^2 T) = O(\mu^{-1+2\eta}\log\mu)$. Since $\eta < 1/2$, by Markov's inequality, the probability that the number of such offspring is at least 1 can be bounded by $O(\mu^{-1+2\eta}\log\mu) = O(\log^{-2}\mu)$, as required.

$\mathcal{E}_b$: There are at most $\varepsilon\mu\log^3\mu$ roots in F_i.

We know from $\mathcal{E}_a(i-1)$ that we may assume that no points in X_i are created from $X_{\le i-2}$. Hence, it suffices to count the number of roots in X_i that are created from X_{i-1}. As in the proof for $\mathcal{E}_a$, by Lemma 9, after the first search point in X_{i-1} is created, with probability $1 - \mu^{-\Omega(1)}$ it takes at most $T = 8e^{\alpha c}\mu\log\mu$ rounds until the set $X_{\le i-1}$ is completely deleted. In each round we have a probability of at most $p_U = O(\varepsilon)$ to create a new root in X_i (p_U defined in Lemma 3), so the expected number of roots in X_i is $O(\varepsilon T)$. By Markov's inequality, the number of roots is at most $\varepsilon\mu\log^3\mu$ with probability $O(\varepsilon T/(\varepsilon\mu\log^3\mu)) = O(\log^{-2}\mu)$.

$\mathcal{E}_c$: No vertex in X_i of depth at most $\phi\log\mu$ in F_i creates offspring in $X_{>i}$.

As a sketch for the proof, we first show that the number of vertices of depth at most $\phi\log\mu$ in F_i is at most $\mu^{2g(\phi)}$ with high probability. Then by a simple estimation, the expected number of offspring in $X_{>i}$ created by those vertices is $O(\mu^{-1+2\eta+2g(\phi)}\log^4\mu)$. Since $g(\phi) < 1/2$ for small enough ϕ, for $\eta < 1/2 - g(\phi)$ with probability $1 - O(\mu^{-\Omega(1)})$ no such offspring is created.

By Lemma 10, we couple F_i with F'. Since by $\mathcal{E}_b$ there are at most $\varepsilon\mu\log^3\mu$ roots in F_i, we only need to consider $\varepsilon\mu\log^3\mu$ trees in F'.

Recall that by Lemma 9, the lifetime of X_i is at most $T := 8e^{\alpha c}\mu\log\mu$ with probability at least $1 - \mu^{-\Omega(1)}$, if $\mu \ge \mu_0$ for a sufficiently large μ_0. Hence, it suffices to study F' after T rounds. We want to bound the number of vertices with depth at most $\phi\log\mu$. We fix a root, and consider the tree attached to this root. By Property Lemma 11.4 and by the Stirling formula $k! = \Theta(\sqrt{k}(k/e)^k)$ in the second step, the expected number of vertices with depth at most $\phi\log\mu$ at round T is

$$\sum_{d=0}^{\phi\log\mu} \mathbb{E}[s_T^d] < 2\,\mathbb{E}[s_T^{\phi\log\mu}] = \frac{2T^{\phi\log\mu}}{(\phi\log\mu)!\mu^{\phi\log\mu}} = \Theta\Big(\frac{(8e^{\alpha c}\mu\log\mu)^{\phi\log\mu}}{\sqrt{\log\mu}(\phi\log\mu/e)^{\phi\log\mu}\mu^{\phi\log\mu}}\Big)$$
$$= o\big(\mu^{\phi(\log(8e^{\alpha c+1})-\log\phi)}\big) = o\big(\mu^{g(\phi)}\big).$$

Note that for $0 < \phi < 1$ we have $g(\phi) > 0$. By Markov's inequality,

$$\Pr\Bigg[\sum_{d=0}^{\phi\log\mu} s_T^d \ge \mu^{2g(\phi)}\Bigg] = o\big(\mu^{-g(\phi)}\big).$$

Since we consider $\varepsilon\mu\log^3\mu = \mu^\eta\log^3\mu$ trees in F', by a union bound over all trees, with probability at least $1 - o(\mu^{\eta-g(\phi)}\log^3\mu)$ the number of vertices with depth at most $\phi\log\mu$ is at most $\mu^{\eta+2g(\phi)}\log^3\mu$. Note that the error probability is $o(1)$ since we assumed that $\eta < g(\phi)$.

In each round, every such vertex has a probability of at most $O(\varepsilon/\mu)$ to create an offspring of strictly larger rank: it must be selected as parent and its offspring must have strictly larger rank. Since the vertices in X_i are present for at most $T = 8e^{\alpha c}\mu\log\mu$ rounds, the expected number of offspring in $X_{\ge i+1}$ created by vertices in X_i of depth at most $\phi\log\mu$ is $O(T \cdot \varepsilon/\mu \cdot \mu^{\eta+2g(\phi)}\log^3\mu) = O(\mu^{-1+2\eta+2g(\phi)}\log^4\mu)$. By Markov's inequality, the probability that the number of such offspring is at least 1 is $O(\mu^{-1+2\eta+2g(\phi)}\log^4\mu)$. Since $g(x)$ is monotonically increasing in $(0, 1)$ and $g(0) = 0$, $\eta < 1/2 - g(\phi)$ holds for small enough constant ϕ, making the error probability $\mu^{-\Omega(1)}$. Hence, we have shown that with sufficiently small probability the vertices in depth at most $\phi\log\mu$ do not create offspring in $X_{>i}$.

$\mathcal{E}_d$: For every vertex $x \in X_i$ that creates an offspring in $X_{\geq i+1}$, if the root r of x has $\textsc{Om}(r) \geq (1-8\varepsilon)n$ then $\textsc{Om}(x) \leq \textsc{Om}(r) - c_d \log\mu$, and if $\textsc{Om}(r) \leq (1-8\varepsilon)n$ then $\textsc{Om}(x) \leq (1-4\varepsilon)n$. Moreover, the mutation changes at most $c_d/2 \cdot \log\mu$ bits.

If $\mathcal{E}_c$ holds, the vertices in X_i that create offspring in $X_{\geq i+1}$ must be of distance at least $d = \phi' \log\mu$ where $\phi' > \phi$ from their roots. Consider a root r with $\textsc{Om}(r) \geq (1-8\varepsilon)n$. By equation (6) in Lemma 11, for $c_d = c\phi/16$, $\Pr[\textsc{Om}(x) - \textsc{Om}(r) \geq -c_d \log\mu] \leq 2e^{-c_d \log\mu/8} = 2\mu^{-M}$, with $M := c\phi/128$. If $\mathcal{E}_b(i+1)$ holds, the number of offspring in $X_{\geq i+1}$ created by points in X_i is at most $\varepsilon\mu\log^3\mu$, which means the number of points in X_i that create offspring in $X_{\geq i+1}$ is at most $\varepsilon\mu\log^3\mu$. By a union bound, with probability at least $1 - O(\varepsilon\mu\log^3\mu \cdot 2\mu^{-M}) = 1 - O(\mu^{-M+\eta}\log^3\mu)$, a vertex in X_i that creates an offspring in $X_{\geq i+1}$ has a $\textsc{Om}$-value which is at least $c_d \log\mu$ smaller than that of its root. Since we assumed $\eta < M$, this probability is $1 - \mu^{-\Omega(1)}$, and thus sufficiently large. This concludes the case that the root has $\textsc{Om}$-value at least $(1-8\varepsilon)n$.

If a vertex x has a root which has at most $\textsc{Om}$-value $(1-8\varepsilon)n$, we consider the first vertex x' of $\textsc{Om}$-value at least $(1-6\varepsilon)n$ on the path from the root to x. Then we know that x' has $\textsc{Om}$-value at most $(1-5\varepsilon)n$, since its direct parent has $\textsc{Om}$-value less than $(1-6\varepsilon)n$ and the probability to flip at least εn bits in one mutation is $2^{-\varepsilon n}$. Then by similar arguments as above, the probability that a descendant of x' has $\textsc{Om}$-value which is εn larger than x' is also $2^{-\varepsilon n} = \mu^{-\omega(1)}$, and thus we can easily apply a union bound over $\varepsilon\mu\log^3\mu$ vertices in X_i that create offspring in $X_{\geq i+1}$.

Finally, we come to the number of bit flips in the improving mutation. In one mutation the expected number of changed bits is c. Let $c_d/2 \cdot \log\mu = (1+\delta')c$ for some $\delta' > 1$, by Chernoff bound, the probability that the number of changed bits is larger than $c_d/2 \cdot \log\mu$ can be bounded by $e^{-\delta' c/3} = \Theta(\mu^{-c_d/6})$. Similarly, by a union bound, the error probability is at most $O(\varepsilon\mu\log^3\mu \cdot \mu^{-c_d/6}) = O(\mu^{\eta - c_d/6}\log^3\mu)$, which is $\mu^{-\Omega(1)}$ since $\eta < c_d/6$.

$\mathcal{E}_e$: No vertex in X_i has an $\textsc{Om}$-value which exceeds the $\textsc{Om}$-value of its root in F_i by more than $c_e \log\mu$.

We set $c_e := 2e^2ck$ where $k > 1$ is a positive constant to be chosen later and assume the distance between a vertex x and its root r is d. By Lemma 11.4, with probability $1 - \mu^{-\Omega(1)}$, $d \leq e\log\mu$, and thus $c_e \log\mu \geq 2edc$. By Lemma 11.2, the probability that x and r differ in more than $c_e \log\mu$ is at most $2^{-c_e \log\mu} = \mu^{-2e^2ck\log 2}$. Therefore, the probability that $\textsc{Om}(x)$ exceeds $\textsc{Om}(y)$ by more than $c_e \log\mu$ is at most $\mu^{-2e^2ck\log 2}$. Moreover, the lifetime of X_i is $8e^{\alpha c}\mu\log\mu$ with probability $1 - \mu^{-\Omega(1)}$ by Lemma 9. By Lemma 11.1, with probability $1 - O(\mu^{1-e^{\alpha c}}\log\mu)$ there are at most $\mu^{9e^{\alpha c}}$ vertices in F_i. By a union bound over all these vertices, the error probability is at most $O(\mu^{9e^{\alpha c} - 2e^2ck\log 2})$. By choosing $k > 9e^{\alpha c}/(2e^2 c\log 2)$, this probability is $\mu^{-\Omega(1)}$. □

5.4. Estimating the drift

We are now ready to collect the information to prove negative drift of the Z_i. We first give a lemma that shows that $Z_{i+K} - Z_i$ is negative in case of good events. As outlined in the introduction, good events don't imply that $Z_{i+1} - Z_i$ is negative, we need to make K steps for some constant $K \in \mathbb{N}$.

Lemma 14. *Let $\ell \in [L]$ and $i \in \mathbb{N}$. Consider the $(\mu+1)$-EA on the linear auxiliary function $f_\ell(x) := n\sum_{j\in A_\ell} x_j + \sum_{j\in R_\ell} x_j$. Assume that in some step $t \geq 0$ the highest rank in the population is i, that $\mathcal{E}_{\text{good}}(i), \ldots, \mathcal{E}_{\text{good}}(i+K)$ hold, where $K := \lceil 2(c_e+1)/c_d \rceil$, and that $\textsc{Om}(r) \geq (1-8\varepsilon)n$ holds for all roots r in $F_i, \ldots, F_{i+K-1}$. Then $Z_{i+K} \leq Z_i - \log\mu$.*

Proof. Let $j \in \{i+1, \ldots, i+K\}$, and let $r \in X_j$ be any root in F_j. By $\mathcal{E}_a(j-1)$, the parent individual x of r is in X_{j-1}. By $\mathcal{E}_d(j-1)$, the root r' of x in F_{j-1} satisfies $\textsc{Om}(r') \geq \textsc{Om}(x) + c_d \log\mu \geq \textsc{Om}(r) + c_d/2 \cdot \log\mu$. By induction, we obtain that for every root $r \in X_j$ there exists a root $\tilde{r} \in X_i$ such that $\textsc{Om}(r) \leq \textsc{Om}(\tilde{r}) - (j-i)c_d/2 \cdot \log\mu \leq Z_i - (j-i)c_d/2 \cdot \log\mu$, where the second step holds since $\textsc{Om}(\tilde{r}) \leq Z_i$ by definition of Z_i. Now consider any individual $\tilde{x} \in X_{i+K}$, and let $r \in X_j$ be its root. By $\mathcal{E}_e(i+K)$, we have

$$\textsc{Om}(\tilde{x}) \leq \textsc{Om}(r) + c_e \log\mu \leq Z_i - K \cdot c_d/2 \cdot \log\mu + c_e \log\mu \leq Z_i - \log\mu, \tag{10}$$

where the last inequality follows from the definition of K. Since (10) holds for all $\tilde{x} \in X_{i+K}$, we obtain $Z_{i+K} \leq Z_i - \log\mu$, as required. □

We are now ready to prove the main theorem on the drift of Z_i. Recall that we have upper, but no lower tail bounds on $Z_i - Z_{i-1}$, cf. the comment before Lemma 12. In order to still be able to apply the negative drift theorem later, we show that the drift is even negative if we truncate the difference $Z_{i+K} - Z_i$ at $-\log\mu$.

Theorem 15. *For every $c > 0$ there is a $\mu_0 \in \mathbb{N}$ and a $K \in \mathbb{N}$ such that for all $\mu_0 \leq \mu \leq n$ where n is sufficiently large the following holds for the $(\mu+1)$-EA with mutation parameter c on the auxiliary function f_ℓ. Assume that in some generation the fittest search point satisfies (3). Then*

$$\mathbb{E}[\max\{Z_{i+K} - Z_i, -\log\mu\}] \leq -1.$$

Proof. Let K be the constant from Lemma 14. Recall from Lemma 13 that the event $\mathcal{E}_{\text{good}}$ has probability $1 - O(\log^{-2}\mu)$, which is at least $1/2$ if μ is sufficiently large. By Lemma 14, the event $\mathcal{E}_{\text{good}}$ implies $Z_{i+K} - Z_i \le -\log\mu$, so in this case the term $\max\{Z_{i+K} - Z_i, -\log\mu\}$ evaluates to $-\log\mu$. Hence, let $E_{\text{good}} := \mathbb{E}[\max\{Z_{i+K} - Z_i, -\log\mu\} \mid \mathcal{E}_{\text{good}}] \cdot \Pr[\mathcal{E}_{\text{good}}]$, it holds that

$$\begin{aligned} E_{\text{good}} &= \sum_{j=-\infty}^{\infty} \max\{j, -\log\mu\} \cdot \Pr[Z_{i+K} - Z_i = j \wedge \mathcal{E}_{\text{good}}] \\ &= (-\log\mu) \cdot \sum_{j=-\infty}^{-\lceil \log\mu \rceil} \Pr[Z_{i+K} - Z_i = j \wedge \mathcal{E}_{\text{good}}] \\ &= -\log\mu \cdot \Pr[\mathcal{E}_{\text{good}}] \le -2, \end{aligned}$$

where the second equality holds because $\Pr[Z_{i+K} - Z_i = j \wedge \mathcal{E}_{\text{good}}] = 0$ for $j \ge -\lfloor \log\mu \rfloor$ and the last step follows from $\Pr[\mathcal{E}_{\text{good}}] \ge 1/2$ if μ is sufficiently large.

In the remainder, we will show that the term E_{good} is very close to $\mathbb{E}[\max\{Z_{i+1} - Z_i, -\log\mu\}]$. In fact, the difference is

$$\begin{aligned} \mathbb{E}[\max\{Z_{i+K} - Z_i, -\log\mu\}] - E_{\text{good}} &= \sum_{j=-\infty}^{\infty} \max\{j, -\log\mu\} \cdot \Pr[Z_{i+K} - Z_i = j \wedge \neg\mathcal{E}_{\text{good}}] \\ &\le \sum_{j=1}^{\infty} j \cdot \Pr[Z_{i+K} - Z_i = j \wedge \neg\mathcal{E}_{\text{good}}]. \end{aligned} \tag{11}$$

For an arbitrary constant $C > 0$ we may define $j_0 := \lceil C \log\mu \log\log\mu \rceil$. Then we bound j by j_0 in the range $j \le j_0$, and we bound $\Pr[Z_{i+K} - Z_i = j \wedge \neg\mathcal{E}_{\text{good}}]$ by $\Pr[Z_{i+K} - Z_i = j]$ for $j > j_0$. Since for $j > j_0$,

$$\begin{aligned} \Pr[Z_{i+K} - Z_i = j] \le \Pr[Z_{i+K} - Z_i \ge j] &= \Pr\left[\sum_{k=1}^{K} (Z_{i+k} - Z_{i+k-1}) \ge j\right] \\ &\le \sum_{k=1}^{K} \underbrace{\Pr[Z_{i+k} - Z_{i+k-1} \ge j/K]}_{\le 2^{-j/(KC_2 \log\mu)} \text{by Lemma 12}} \le K 2^{-j/(KC_2 \log\mu)}. \end{aligned} \tag{12}$$

We obtain

$$\begin{aligned} (11) &\le \sum_{j=1}^{j_0} j_0 \cdot \Pr[Z_{i+K} - Z_i = j \wedge \neg\mathcal{E}_{\text{good}}] + \sum_{j=j_0+1}^{\infty} j \cdot \Pr[Z_{i+K} - Z_i = j] \\ &\le j_0 \cdot \underbrace{\Pr[\neg\mathcal{E}_{\text{good}}]}_{=O(\log^{-2}\mu)} + \sum_{j=j_0+1}^{\infty} j \cdot \underbrace{\Pr[Z_{i+1} - Z_i = j]}_{\le K 2^{-j/(KC_2 \log\mu)} \text{ by } (12)} \\ &= O\left(j_0 \log^{-2}\mu\right) + O\left(\log\mu \cdot j_0 2^{-j_0/(KC_2 \log\mu)}\right), \end{aligned}$$

where the factor $\log\mu$ in the second term appears because $\sum_{s=s_0}^{\infty} s 2^{-s/x} = O(x s_0 2^{-s_0/x})$ for $x \ge 1$, which can be seen by grouping the sum into batches of x summands. The second term is $O(\log^{-1}\mu)$ if we choose the constant $C > 0$ in the definition of $j_0 = \lceil C \log\mu \log\log\mu \rceil$ appropriately. The first term is $O(\log\log\mu \cdot \log^{-1}\mu)$. Hence, by choosing μ sufficiently large, we can make both terms smaller than $1/2$, and obtain that $\mathbb{E}[\max\{Z_{i+K} - Z_i, -\log\mu\}] \le E_{\text{good}} + 1 \le -1$, as desired. □

6. Proof of Theorem 6

In the previous section we have analyzed the random variable Z_i, and in particular we have shown that it has negative drift. In this section we will show how our main result, the lower bound on the runtime for the $(\mu + 1)$-EA, follows from the negative drift of Z_i. The proof follows from similar ideas as in [5] and [6]. We start with a lemma that describes the behavior of the $(\mu + 1)$-EA on f_ℓ.

Lemma 16. *For every constant $0 < \delta < 2/7$ the following holds. Let $\ell \in [L]$ and consider the $(\mu + 1)$-EA on f_ℓ under the assumption that $d([n], x) \ge \varepsilon(1 + 2\delta)$ and $d(A_{\ell+1}, x) \ge \varepsilon(1 + \delta)$ hold for all x in the initial population. For $t \ge 0$, let x^t be the offspring in round t. Then with probability $1 - \exp\{-\Omega(\varepsilon n / \log^2 \mu)\}$, the following holds for all $t \le L$.*

- *$d([n], x^t) \geq \varepsilon(1+\delta)$.*
- *$d([n], x^t) \geq \varepsilon(1+2\delta)$ or $d(A_{\ell+1}, x^t) \geq \varepsilon(1+\delta/4)$.*

Before we come to the proof, let us briefly explain why the lemma is useful. It is tailored to support an inductive proof for Theorem 6 for HotTopic. In this induction, we will show that $d([n], x^t) \geq \varepsilon(1+\delta)$ for exponential time. In fact, when the algorithm enters a new level then the density is at least $\varepsilon(1+2\delta)$. Moreover, one can show that with high probability the new hot topic did not influence the algorithm up to this point, so it behaves just as a random subset of positions of size αn. In particular, with high probability its density is at least $\varepsilon(1+\delta)$, so the assumptions of the lemma are satisfied. As long as the level does not change, the HotTopic function is identical to f_ℓ, so we may apply Lemma 16. The first item implies what we actually want to prove, at least as long as we stay on the same level. For the second item, by the construction of the HotTopic function the level increases when $d(A_{\ell+1}, x^t) \approx \varepsilon < \varepsilon(1+\delta/4)$. So the second item implies that at this point in time we have $d([n], x^t) \geq \varepsilon(1+2\delta)$, which is the requirement for the next step in the induction. Note that we can't just merge the items into one. For example, if we would weaken the second item to assert $d([n], x^t) \geq \varepsilon(1+\delta)$ at the beginning of a level, then we could not conclude that the next offspring satisfies the same bound with exponentially small error probability.

Proof of Lemma 16. Let i_0 be the largest rank in the initial population, i.e., the largest number of one-bits in $A_{\ell+1}$ in the initial population. We fix an offset $a \in \{0, \dots, K-1\}$ and consider the sequence of random variables $Y_{i,a} := Z_{i_0+a+iK}/\log\mu$, where i is a non-negative integer. In the initial population, each individual has at most $n(1-\varepsilon(1+2\delta))$ one-bits by assumption. Hence, we also have $Z_{i_0+a} \leq n(1-\varepsilon(1+3\delta/2))$ with probability $1-\exp\{-\Omega(\varepsilon n)\}$ for all offsets $a \in \{0, \dots, K-1\}$, since otherwise at least one of the K mutations would need to flip $\Omega(\varepsilon n)$ bits, which happens only with probability $\exp\{-\Omega(\varepsilon n)\}$ by the Chernoff bound. Thus for the first statement it suffices to show that $Y_{i,a} \leq Y_{0,a} + \varepsilon\delta n/(2\log\mu)$ for all $i \geq 0$. Since that is equivalent to $Z_{i_0+a+iK} \leq Z_{i_0+a} + \varepsilon\delta n/2$ for all $i \geq 0$, and we already have $Z_{i_0+a} \leq n(1-\varepsilon(1+3\delta/2))$ for all a with high probability, altogether it implies $Z_{i'} \leq n(1-\varepsilon(1+\delta))$ for all $i' \geq i_0$. As $Z_{i'}$ denotes the maximum number of one-bits in rank i', we conclude that $d([n], x^t) \geq \varepsilon(1+\delta)$ holds for any individual x^t of rank $i' \geq i_0$. For the second statement, we distinguish between two cases. Note that the index i counts, up to the factor K, the increase in one-bits in $A_{\ell+1}$. If $i \leq \alpha n\varepsilon\delta/(4K) - 1$, then for any x^t of rank $i_0 + a + iK$, $d(A_{\ell+1}, x^t) \geq (\alpha n\varepsilon(1+\delta/2) - a - iK)/(\alpha n) > \varepsilon(1+\delta/2) - (i+1)K/(\alpha n) \geq \varepsilon(1+\delta/4)$. For $i > \alpha n\varepsilon\delta/(4K) - 1$, we aim to show that $Y_{i,a} \leq Y_{0,a}$.

We would like to apply the negative drift theorem to $Y_{i,a}$ for the range $[(1-\varepsilon(1+3\delta/2))n/\log\mu, (1-\varepsilon(1+\delta))n/\log\mu]$. First note that we study a linear function, and that the bits in A_ℓ have larger weights than the remaining bits. Thus, it can be shown by a coupling argument (Lemma 4.2 in [5]) that if $d(A_{\ell+1}, x) \leq d([n], x) + \delta\varepsilon$ holds initially, then the slightly weaker condition $d(A_{\ell+1}, x) \leq d([n], x) + 2\delta\varepsilon$ remains true for all individuals in the population for the next L rounds, with probability at least $1 - Le^{-\Omega(\varepsilon n)}$. By choosing the constant parameter ρ in the definition of $L = \exp\{\rho\varepsilon n/\log^2\mu\}$ small enough, the factor L can be swallowed by the term $e^{-\Omega(\varepsilon n)}$. Thus we may assume that whenever $Y_{i,a}$ is in the range $[(1-\varepsilon(1+3\delta/2))n/\log\mu, (1-\varepsilon(1+\delta))n/\log\mu]$ then $d(A_{\ell+1}, x) \leq d([n], x) + 2\delta\varepsilon \leq \varepsilon(1+3\delta/2+2\delta) \leq 2\varepsilon$ as $\delta < 2/7$. In addition, we have $d(A_{\ell+1}, x) \geq \varepsilon/2$ before the level changes, since otherwise with probability $1 - e^{-\Omega(\varepsilon n)}$ it holds that $d(B_{\ell+1}, x) < \varepsilon$, which implies an increase of level. Thus the conditions in (3) are satisfied, and thus Lemma 6 is applicable.

So let us study the drift of $Y_{i,a}$ in the range $[(1-\varepsilon(1+3\delta/2))n/\log\mu, (1-\varepsilon(1+\delta))n/\log\mu]$. First note that the probability to jump over more than half of this interval is $\exp\{-\Omega(\varepsilon n/\log^2\mu)\}$: for $Y_{i,a} \geq (1-4\varepsilon)n/\log\mu$ this follows from the first statement in Lemma 12, for $Y_{i,a} < (1-4\varepsilon)n/\log\mu$ it follows from the second statement in Lemma 12. So we may assume that $Y_{i,a}$ is contained in the first half of the interval for some $i = i^*$. To ease notation, we will assume $i^* = 0$. Inside of the interval, by Theorem 15, it holds that

$$\mathbb{E}[Y_{i+1,a} - Y_{i,a}] = \mathbb{E}[Z_{i_0+a+(i+1)k} - Z_{i_0+a+ik}]/\log\mu \leq -1/\log\mu.$$

Moreover, by Lemma 12 the sequence of random variables $(Y_{i,a})_{i\geq 0}$ has an upper exponential tail bound, i.e., $\Pr[Y_{i+1,a} - Y_{i,a} \geq K \cdot \beta C_2] \leq K \cdot 2^{-\beta}$ for all $1 \leq \beta \leq \varepsilon n/\log^2\mu$. (In particular, the probability that there is ever a jump larger than $KC_2\varepsilon n/\log^2\mu$ within L steps is at most $O(L \cdot 2^{-\varepsilon n/\log^2\mu}) = o(1)$, so we may assume that such jumps never occur.) To show sub-Gaussianity, we should extend the inequality also for $\beta < 1$. Since any probability is bounded by 1, the bound $\Pr[Y_{i+1,a} - Y_{i,a} + 1/\log\mu \geq K\beta C_2 + 1/\log\mu] \leq 2K \cdot 2^{-\beta}$ is not just true for $\beta \geq 1$, but also trivially satisfied for any $\beta \in [0,1]$. Therefore, for any $y \geq 0$, it holds that

$$\begin{aligned}\Pr[Y_{i+1,a} - Y_{i,a} + 1/\log\mu \geq y] &\leq 2^{1+1/(KC_2\log\mu)}K(2^{1/(KC_2)})^{-y}\\ &\leq 2^{1+1/(KC_2\log 2)}K(2^{1/(KC_2)})^{-y}.\end{aligned}$$

However, we need exponential tail bounds in both directions, so we need to truncate the downwards steps of $Y_{i,a}$ as follows. We set $\tilde{Y}_{0,a} := Y_{0,a}$, and we define $\tilde{Y}_{i,a}$ recursively by $\tilde{Y}_{i,a} - \tilde{Y}_{i+1,a} := \min\{Y_{i,a} - Y_{i+1,a}, 1/\log\mu\}$. Then clearly we have $\tilde{Y}_{i,a} \geq Y_{i,a}$ for all $i \geq 0$, and $\tilde{Y}_{i,a}$ satisfies the tail bound condition that

$$\Pr[\tilde{Y}_{i,a} - \tilde{Y}_{i+1,a} - 1/\log\mu > 0] = 0.$$

Therefore, by Theorem 4, $(\tilde{Y}_{i,a}+i/\log\mu)_{i\geq 0}$ is $(128c'\delta'^{-3}, \delta'/4)$-sub-Gaussian, where $c' = 2^{1+1/(KC_2\log 2)}K$ and $\delta' = 2^{1/(KC_2)} - 1$. And by Theorem 5,

$$\Pr\Big[\max_{0\leq j\leq i}(Y_{j,a} - Y_{0,a}) \geq -i/\log\mu + y\Big] \leq \exp\Big(-\frac{\delta' y}{8}\min\Big(1, \frac{\delta'^2}{32c'}\cdot\frac{y}{i}\Big)\Big).$$

Now for any $i \geq 0$ and $y = i/\log\mu + \varepsilon\delta n/(4\log\mu)$, with probability $1-\exp\{-\Omega(\varepsilon n/\log^2\mu)\}$ we have $Y_{i,a} \leq Y_{0,a} + \varepsilon\delta n/(4\log\mu)$. Note that $\varepsilon\delta n/(4\log\mu)$ is half of the length of the interval of interest, which implies that $Y_{i,a}$ does not go beyond the interval with high probability. Similarly, for every fixed $i \geq \alpha n\varepsilon\delta/(4K)$ and $y = i/\log\mu$ we have $Y_{i,a} \leq Y_{0,a}$ with probability $1-\exp\{-\Omega(\varepsilon n/\log^2\mu)\}$. The proof is concluded by a union bound over all possible i. Since there are at most n possible values, this increases the error probability by a factor of n, which we can swallow in the expression $\exp\{-\Omega(\varepsilon n/\log^2\mu)\}$. □

Finally, we have collected all ingredients to prove our main result.

Proof of Theorem 6. Let $L := \exp\{\rho\varepsilon n/\log^2\mu\}$ be the number of levels. For the proof, we will consider an auxiliary run of the $(\mu+1)$-EA with a dynamic fitness function $\tilde{f}$ in which we only allow the levels to increase by one. In particular, the function $\tilde{f}$ does not only depend on the current state of the algorithm, but also on the algorithm's history. More precisely, we define an auxiliary level $\tilde{\ell}(x,t)$ of a search point x, which we only allow to increase by at most one per round. Recall that $\ell(x)$ was defined in (1) as $\ell(x) = \max\{\ell' \in [L] : d(B_{\ell'}, x) \leq \varepsilon\}$. For $\tilde{\ell}(t)$, we use the same definition except that we let the maximum go over only $\ell' \leq \min\{\tilde{\ell}(t-1)+1, L\}$. I.e., we set $\tilde{\ell}(0) := 0$, and if an offspring y^t of x^t enters the population in round t, then we set $\tilde{\ell}(y^t,t) := \max\{\ell' \in [\min\{\tilde{\ell}(x^t,t-1)+1, L\}] : d(B_{\ell'}, y^t) \leq \varepsilon\}$. (If the population stays the same in round t, then we leave $\tilde{\ell}$ unchanged.) Then we define the auxiliary fitness of y^t as

$$\tilde{f}(y^t) := \tilde{\ell}(y^t,t)\cdot n^2 + \sum_{i\in A_{\tilde{\ell}(y^t,t)+1}} y_i^t \cdot n + \sum_{i\in R_{\tilde{\ell}(y^t,t)+1}} y_i^t,$$

i.e., we use the same definition as for the HotTopic function except that we replace $\ell(y^t)$ by $\tilde{\ell}(t)$. Then we proceed as the $(\mu+1)$-EA, i.e., in each round we compute and store the auxiliary fitness of the new offspring (which may depend on the whole history of the algorithm), and we remove the search point for which we have stored the lowest auxiliary fitness. This definition does not make much sense from an algorithmic perspective, but we will see in hindsight that the auxiliary process behaves identical to the actual $(\mu+1)$-EA. We will next argue why this is the case.

For the auxiliary process, it is obvious that we only need to uncover the set A_{i+1} and B_{i+1} when we reach level $\tilde{\ell}(t) = i$. As we will show later for the auxiliary process, with high probability the density $d([n], x^t)$ stays strictly above $\varepsilon\cdot(1+\delta)$ for a suitable constant $\delta > 0$. Now fix any round t with auxiliary level $\tilde{\ell}(t)$. Since we do need to uncover $B_{\tilde{\ell}(t)+2}$ at some point after time t, its choice does not influence the behavior of the auxiliary process until time t. Hence, we can first let the auxiliary process run until time t, and afterwards uncover the set $B_{\tilde{\ell}(t)+2}$. Since $B_{\tilde{\ell}(t)+2} \subset [n]$ is a uniformly random subset of size βn, it contains at least $\beta\varepsilon(1+\delta)n$ zero-bits in expectation, and the probability that $B_{\tilde{\ell}(t)+2}$ contains at most $\beta\varepsilon n$ zero-bits is $\exp\{-\Omega(\beta\varepsilon n)\}$. The same argument also holds for $B_{\tilde{\ell}(t)+3}, \dots, B_L$. Since $L = \exp\{\rho\varepsilon n/\log^2\mu\}$ with desirably small $\rho > 0$, we can afford a union bound over all such sets and all times $t \leq L$, which is a union bound over less than $L^2 = \exp\{2\rho\varepsilon n/\log^2\mu\}$ terms. Hence, with high probability we have $d(B_i, x^t) > \varepsilon$ for all $1 \leq t \leq L$ and all $\tilde{\ell}(t)+2 \leq i \leq L$. A straightforward induction shows that this implies $\ell(t) = \tilde{\ell}(t)$ for all $t \leq L$, and thus the $(\mu+1)$-EA behaves identical to the auxiliary process. Note that this already implies that the $(\mu+1)$-EA visits each of the L levels, which implies the desired runtime bound. It only remains to show that there is a constant $\delta > 0$ such that the auxiliary process satisfies $d([n], x^t) > \varepsilon\cdot(1+\delta)$ for all $t \leq L$.

The advantage of the auxiliary process is that we may postpone drawing $A_{\ell+1}$ until we reach level $\tilde{\ell} = \ell$. In particular, since $A_{\ell+1} \subseteq [n]$ is a uniformly random subset, we may use the same argument as before and conclude that $|d(A_{\ell+1}, x) - d([n], x)| < \delta\varepsilon$ holds with probability $1-\exp\{-\Omega(\varepsilon n)\}$ for any constant $\delta > 0$ that we desire, and for all members x of the population when we reach level ℓ. In fact, we have exponentially small error probability, so we may afford a union bound and conclude that with high probability the same holds for all ℓ. We want to show that the auxiliary process, if running on level ℓ and starting with a population that initially satisfies $|d(A_{\ell+1}, x) - d([n], x)| < \delta\varepsilon$ for $\delta < 2/7$, maintains $d([n], x^t) \geq \varepsilon(1+\delta)$ for all new search points x^t until $t > L$.

By the first conclusion from Lemma 16, $d([n], x^t) \geq \varepsilon(1+\delta)$ holds as long as the level remains to be ℓ and $t \leq L$. When a point x reaches level $\ell+1$, by definition we have $d(B_{\ell+1}, x) < \varepsilon$. Since $B_{\ell+1}$ is a uniformly random subset of $A_{\ell+1}$, by the Chernoff bound $d(A_{\ell+1}, x) < \varepsilon(1+\delta/4)$ holds with probability $1-\exp\{-\Omega(\varepsilon n)\}$. So we apply the second conclusion of Lemma 16 to x and conclude that $d([n], x) \geq \varepsilon(1+2\delta)$. With high probability, it holds that $d(A_{\ell+2}, x) \geq \varepsilon(1+2\delta) - \varepsilon\delta$ and the conditions in Lemma 16 are satisfied again for level $\ell+1$. By induction we obtain $d([n], x^t) \geq \varepsilon(1+\delta)$ for all $t \leq L$. As the choice of ℓ is arbitrary, we start with $\ell = 0$ and $d([n], x^t) \geq \varepsilon(1+\delta)$ holds for all $t \leq L$. This concludes the proof. □

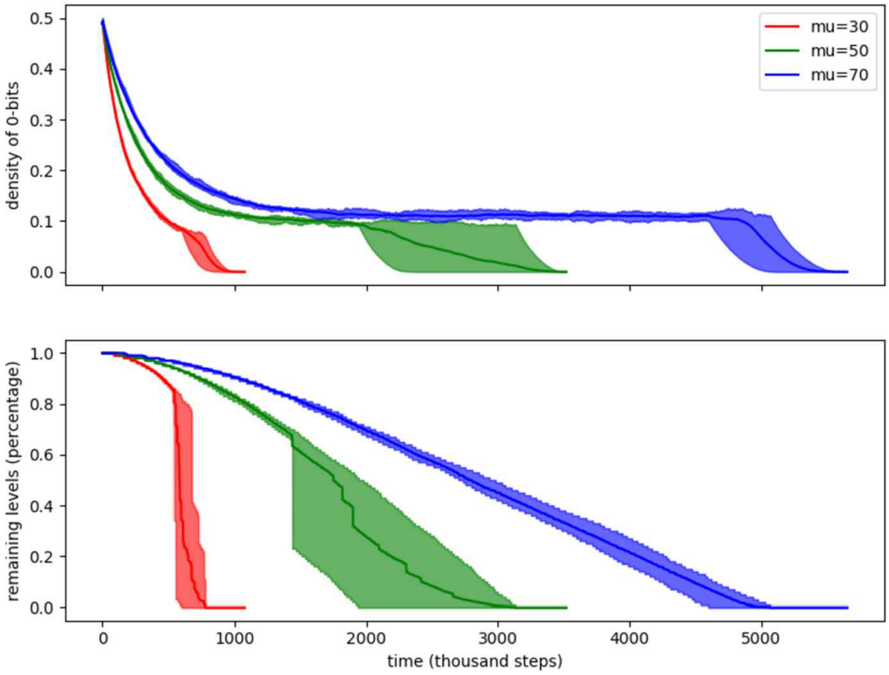

Fig. 2. Distance to the optimum as $(\mu + 1)$-EAs with $c = 1.0$ proceed. The solid lines are the mean of 10 simulations, while the shaded areas are bounded by corresponding minimum and maximum values at each time step. (For interpretation of the colors in the figure, the reader is referred to the web version of this article.)

7. Simulation

In this section we will illustrate the detrimental effect of large populations on $(\mu + 1)$-EAs by numerical simulations. Unless otherwise stated, the parameters that we used to generate the HotTopic functions are $n = 10000$, $L = 100$, $\alpha = 0.25$, $\beta = 0.05$ and $\varepsilon = 0.05$. Each data point is obtained by 10 independent runs on the same HotTopic function with different random seeds. Our implementation is available at https://github.com/zuxu/MuOneEA-HotTopic.

7.1. Population size

First of all, we plot typical behaviors of evolutionary algorithms with small, medium and large population sizes. Fig. 2 shows the distance between the optimum and the fittest point p^* in the population with respect to time. We have two metrics for the distance: the density of 0-bits in p^* and the remaining levels of p^* divided by L. As indicated by the sudden drops in level, for a small population size ($\mu = 30$), the algorithm skips many levels and reaches the optimum quickly. In contrast, an algorithm with large $\mu = 70$ visits the levels one by one, without improvement on the fitness. This happens where the density of 1-bits is relatively high, such that even though it gradually improves on the current hot topic, it often accepts offspring that flip 1-bits to 0-bits outside of the hot topic. With such offspring accumulating in a large population, the average density of 0-bits remains significantly above ε before reaching the last level. Therefore, with high probability the algorithm does not skip any level. Once the highest level is reached, the remaining bits can be optimized easily as in the coupon collector. For $\mu = 50$, the density gets close, but slightly above ε, so that it depends on chance whether levels are skipped or not. This leads to a high variance in the running time.

In Fig. 3, we show the running time and the number of visited levels for a wide range of μ. The running time is highly concentrated when μ is very small or very large. The reason is that the algorithm keeps skipping levels with small μ and visits all levels with large μ. For a medium sized μ like 50, level skipping only happens a few times. Since each time when the algorithm skips a level, it lands at some higher level uniformly at random due to the definition of the HotTopic function, which results in a larger variance in the running time.

7.2. Mutation rate

The mutation rate c is the other factor that affects the magnitude of the negative drift, so we also plot the running time for various values of c, see Fig. 4. For a small c, detrimental mutations do not occur frequently and thus the average density of 1-bits in the population keeps increasing. Conversely, with a large c, the algorithm tends to visit all the levels. To demonstrate the resulting effect on the running time, we compare the cases where $L = 100$ and $L = 200$. If c is small ($c \leq 0.9$), the algorithm skips levels quickly, and the running time is almost independent of the number of levels. On

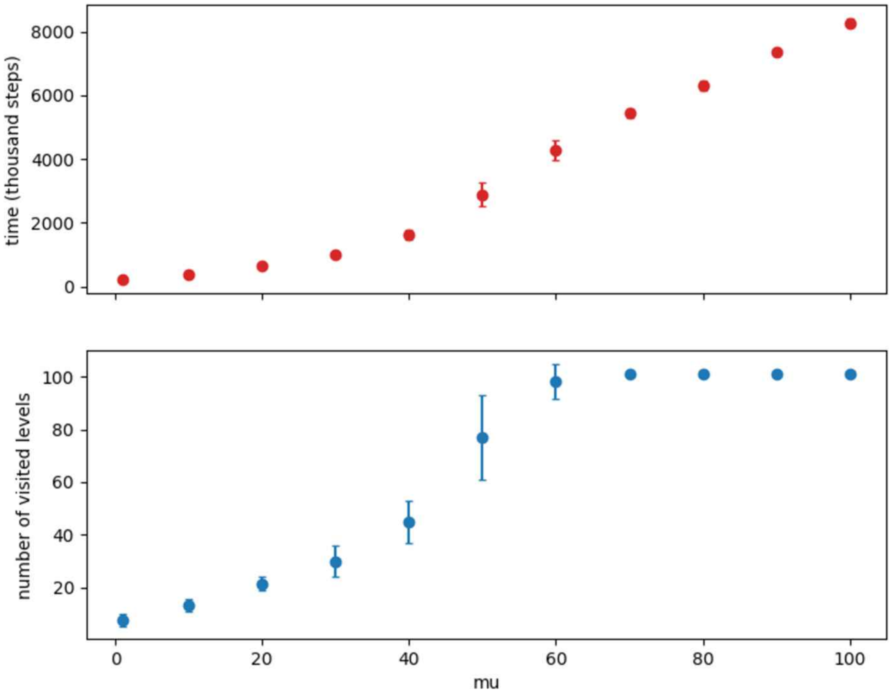

Fig. 3. Running time and number of visited levels for $(\mu + 1)$-EAs with different values of μ and $c = 1.0$. Solid dots indicate the means and error bars show the standard deviations. (For interpretation of the colors in the figure, the reader is referred to the web version of this article.)

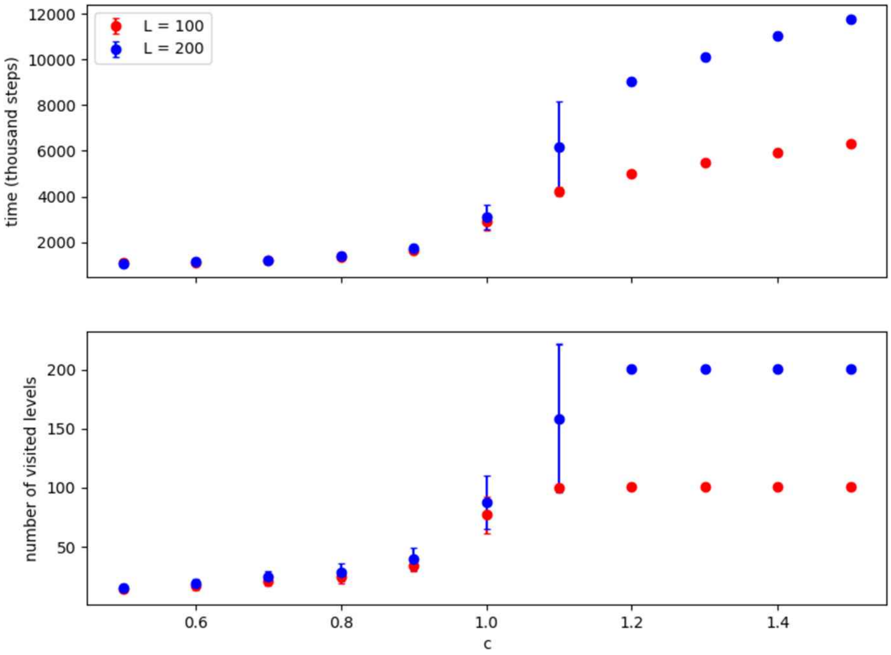

Fig. 4. Running time and number of visited levels for $(\mu + 1)$-EAs with different values of c and $\mu = 50$. Solid dots indicate the means and error bars show the standard deviations. (For interpretation of the colors in the figure, the reader is referred to the web version of this article.)

the other hand, if c is large ($c \geq 1.2$) then the algorithm visits every level. In this case, the running time is essentially proportional to the number of levels, plus some initial phase. Note that in this range the running time can get almost arbitrarily bad, since doubling the number of levels L will essentially lead to a doubling of the running time. As our theoretical analysis shows, this holds even when L becomes exponential in n, but for so many levels the running time becomes too large to run experiments. Finally, for a medium sized c like 1.1, level skipping only happens a few times. Each

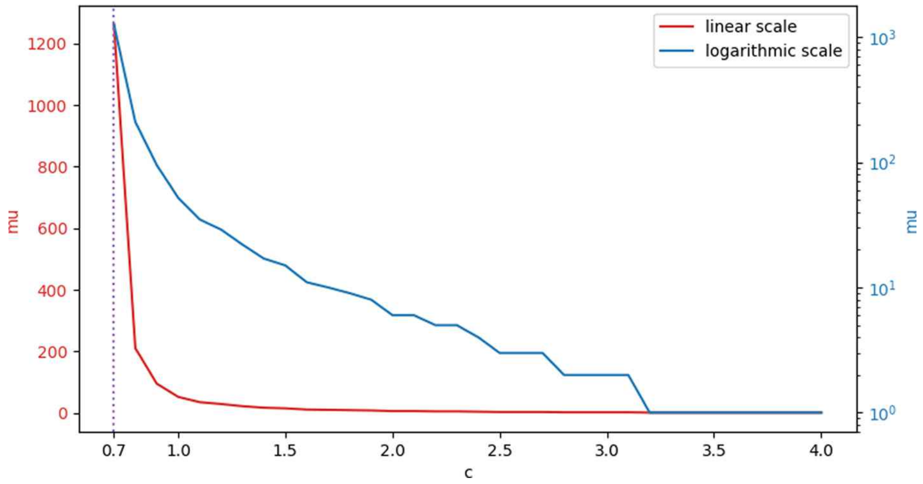

Fig. 5. Minimum values of μ with respect to c such that the $(\mu+1)$-EA visits all levels in at least 5 out of 10 simulations. The choices of c ranges from 0.7 to 4.0 with step size 0.1. (For interpretation of the colors in the figure, the reader is referred to the web version of this article.)

time when the algorithm skips a level, it lands at some higher level uniformly at random, which results in a larger variance in the running time, similar to Fig. 3.

Finally, we investigate how values of μ and c jointly influence the behavior of a $(\mu+1)$-EA. For a fixed c, we search for the minimum value of μ such that the algorithm visits all levels in at least half of the 10 simulations. That is, we seek for a minimum μ that induces long running times with constant probability. As we can see from Fig. 5, large values of c (≥ 3.2) are extremely harmful even when there is only one individual in the population. An algorithm with a large population can benefit greatly from having a small mutation rate. We did not observe a stable slowdown for $c = 0.7$ until we raise the value of μ to more than 1000.

8. Conclusion

We have shown that the $(\mu+1)$-EA with arbitrary mutation parameter $c > 0$ needs exponential time on some monotone functions if μ is too large. This is one of the very few known situations in which even a slightly larger population size μ can lead to a drastic decrease in performance. The main reason is that, if progress is steady enough that the population does not degenerate, the search points that produce offspring are typically not the fittest ones. We believe that this is an interesting phenomenon which deserves further investigations, also in less artificial contexts.

For example, consider the $(\mu+1)$-EA on weighted linear functions with a skewed distribution (e.g., on BinVal), and with a fixed time budget (so that the action happens away from the optimum). It is quite conceivable that the same effect hurts performance, i.e., if the algorithm flips a high-weight bit, it will allow (almost) any offspring of this individual into the population, even though this offspring has probably fewer correct bits than other search points in the population. Does that mean that the fixed-budget performance of the $(\mu+1)$-EA on BinVal deteriorates with increasing μ? Are the resulting individuals further away from the optimum?

An even more pressing question is about crossover. We have studied the $(\mu+1)$-EA, but do the same results also apply for the $(\mu+1)$-GA? In [6] it was shown that close to the optimum (for small values of the HotTopic parameter ε) crossover helps dramatically, and that a large population size can even counterbalance large mutation parameters c. So, close to the optimum, for the $(\mu+1)$-GA the effect of large population size was beneficial, while for the $(\mu+1)$-EA it was neutral and did not affect the threshold c_0. Thus if we study the $(\mu+1)$-GA on HotTopic functions with large ε, then a beneficial effect of large populations is competing with a detrimental effect. Understanding this interplay would be a major step towards a better understanding of crossover in general.

Similarly, since the problems originate in non-trivial populations, what happens if we equip the $(\mu+1)$-EA with a diversity mechanism (duplication avoidance, genotypical or phenotypical niching), and study it close to the optimum? Does it fall for the same traps? This question was already asked in [6], but our results shed additional light on the question.

Finally, it is open whether the $(\mu+1)$-EA is fast on any monotone function if it starts close enough to the optimum. i.e., for every $\mu \in \mathbb{N}$, does there exist an $\varepsilon = \varepsilon(\mu)$ such that the $(\mu+1)$-EA, initialized with a random search point with εn zero-bits, has runtime $O(n \log n)$ for every monotone function? Of course, the same question also applies to other algorithms like the $(\mu+1)$-GA and the 'fast' counterparts of the $(\mu+1)$-EA and the $(\mu+1)$-GA. Interestingly, the result in [6] that the 'fast $(1+\lambda)$-EA' with good parameters is efficient for every monotone function was only proven under this assumption, that the algorithm starts close to the optimum. So this also raises the question whether there are traps for the 'fast $(1+\lambda)$-EA' that only take effect far away from the optimum.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] C. Witt, Runtime analysis of the $(\mu + 1)$ EA on simple pseudo-boolean functions, Evol. Comput. 14 (1) (2006) 65–86.
[2] D. Antipov, B. Doerr, A tight runtime analysis for the $(\mu + \lambda)$ EA, Algorithmica (2020) 1–42.
[3] J.N. Richter, A. Wright, J. Paxton, Ignoble trails-where crossover is provably harmful, in: International Conference on Parallel Problem Solving from Nature, in: LNCS, vol. 5199, Springer, 2008, pp. 92–101.
[4] C. Witt, Population size versus runtime of a simple evolutionary algorithm, Theor. Comput. Sci. 403 (1) (2008) 104–120.
[5] J. Lengler, A. Steger, Drift analysis and evolutionary algorithms revisited, Comb. Probab. Comput. 27 (4) (2018) 643–666.
[6] J. Lengler, A general dichotomy of evolutionary algorithms on monotone functions, in: International Conference on Parallel Problem Solving from Nature, in: LNCS, vol. 11102, Springer, 2018, pp. 3–15.
[7] B. Doerr, T. Jansen, D. Sudholt, C. Winzen, C. Zarges, Optimizing monotone functions can be difficult, in: International Conference on Parallel Problem Solving from Nature, in: LNCS, vol. 6238, Springer, 2010, pp. 42–51.
[8] B. Doerr, T. Jansen, D. Sudholt, C. Winzen, C. Zarges, Mutation rate matters even when optimizing monotonic functions, Evol. Comput. 21 (1) (2013) 1–27.
[9] T. Jansen, On the brittleness of evolutionary algorithms, in: International Workshop on Foundations of Genetic Algorithms, Springer, 2007, pp. 54–69.
[10] J. Lengler, A. Martinsson, A. Steger, When does hillclimbing fail on monotone functions: an entropy compression argument, in: Workshop on Analytic Algorithmics and Combinatorics, SIAM, 2019, pp. 94–102.
[11] J. Lengler, A general dichotomy of evolutionary algorithms on monotone functions, IEEE Trans. Evol. Comput. (2019) 1–15.
[12] B. Doerr, H.P. Le, R. Makhmara, T.D. Nguyen, Fast genetic algorithms, in: Proceedings of the Genetic and Evolutionary Computation Conference, 2017, pp. 777–784.
[13] T. Kötzing, J.G. Lagodzinski, J. Lengler, A. Melnichenko, Destructiveness of lexicographic parsimony pressure and alleviation by a concatenation crossover in genetic programming, in: International Conference on Parallel Problem Solving from Nature, in: LNCS, vol. 11102, Springer, 2018, pp. 42–54.
[14] T. Friedrich, T. Kötzing, M.S. Krejca, A.M. Sutton, The benefit of recombination in noisy evolutionary search, in: International Symposium on Algorithms and Computation, Springer, 2015, pp. 140–150.
[15] C. Qian, Y. Yu, Z.-H. Zhou, An analysis on recombination in multi-objective evolutionary optimization, Artif. Intell. 204 (2013) 99–119.
[16] T. Kötzing, D. Sudholt, M. Theile, How crossover helps in pseudo-boolean optimization, in: Conference on Genetic and Evolutionary Computation, ACM, 2011, pp. 989–996.
[17] J. Lengler, U. Schaller, The (1+1)-EA on noisy linear functions with random positive weights, in: Foundations of Computational Intelligence, in IEEE Symposium Series on Computational Intelligence, IEEE, 2018, pp. 712–719.
[18] B. Doerr, Probabilistic tools for the analysis of randomized optimization heuristics, in: Theory of Evolutionary Computation, Springer, pp. 1–87.
[19] C. Witt, Fitness levels with tail bounds for the analysis of randomized search heuristics, Inf. Process. Lett. 114 (1–2) (2014) 38–41.
[20] T. Kötzing, Concentration of first hitting times under additive drift, Algorithmica 75 (3) (2016) 490–506.
[21] P.K. Lehre, X. Yao, On the impact of mutation-selection balance on the runtime of evolutionary algorithms, IEEE Trans. Evol. Comput. 16 (2) (2012) 225–241.
[22] D. Sudholt, The impact of parametrization in memetic evolutionary algorithms, Theor. Comput. Sci. 410 (26) (2009) 2511–2528.
[23] B. Doerr, Probabilistic tools for the analysis of randomized optimization heuristics, in: Theory of Evolutionary Computation, Springer, 2020, pp. 1–87.
[24] J. Lässig, D. Sudholt, Design and analysis of migration in parallel evolutionary algorithms, Soft Comput. 17 (7) (2013) 1121–1144.