Runtime Analysis for Self-adaptive Mutation Rates

Benjamin Doerr[1] · Carsten Witt[2] · Jing Yang[1]

Abstract

We propose and analyze a self-adaptive version of the $(1, \lambda)$ evolutionary algorithm in which the current mutation rate is encoded within the individual and thus also subject to mutation. A rigorous runtime analysis on the OneMax benchmark function reveals that a simple local mutation scheme for the rate leads to an expected optimization time (number of fitness evaluations) of $O(n\lambda/\log\lambda + n\log n)$ when λ is at least $C \ln n$ for some constant $C > 0$. For all values of $\lambda \geq C \ln n$, this performance is asymptotically best possible among all λ-parallel mutation-based unbiased black-box algorithms. Our result rigorously proves for the first time that self-adaptation in evolutionary computation can find complex optimal parameter settings on the fly. In particular, it gives asymptotically the same performance as the relatively complicated self-adjusting scheme for the mutation rate proposed by Doerr, Gießen, Witt, and Yang (Algorithmica 2019). On the technical side, the paper contributes new tools for the analysis of two-dimensional drift processes arising in the analysis of dynamic parameter choices in EAs, including bounds on occupation probabilities in processes with non-constant drift.

Keywords Evolutionary algorithms · Self-adaptive · Runtime analysis

1 Introduction

Evolutionary algorithms are a class of heuristic algorithms that can be applied to solve optimization problems if no problem-specific algorithm is available. For example, this may be the case if the structure of the underlying problem is poorly understood or one

---

✉ Carsten Witt
  cawi@imm.dtu.dk

1   Laboratoire d'Informatique (LIX), CNRS, École Polytechnique, Institut Polytechnique de Paris, Palaiseau, France

2   DTU Compute, Technical University of Denmark, Kongens Lyngby, Denmark

is faced with a so-called black-box scenario, in which the quality of a solution can only be determined by calling an implementation of the objective function. This implementation may be implicitly given by, e. g., the outcome of a simulation without revealing structural relationships between the search point and the function value.

An approach to understand the working principles of evolutionary algorithms is to analyze the underlying stochastic process and its first hitting time of the set of optimal or approximate solutions. The runtime analysis community in evolutionary computation (see, e. g., [2, 31, 41, 54] for introductions to the subject) follows this approach by partly using methods known from the analysis of classical randomized algorithms and, more recently and increasingly often, using and adapting tools from the theory of stochastic processes to obtain bounds on the hitting time of optimal solutions for different classes of evolutionary algorithms and problems. Such bounds will typically depend on the problem size, the problem type, the evolutionary algorithm, and the values of the parameters that these heuristic algorithms come with.

One of the core difficulties when using evolutionary algorithms is in fact finding suitable values for its parameters. It is well known and supported by ample experimental (e. g., [8, 58]) and theoretical evidence (e. g., [24, 28, 44, 48, 61]) that already small changes of the parameters can have a crucial influence on the efficiency of the algorithm.

One elegant way to overcome this difficulty, and in addition the difficulty that the optimal parameter values may change during a run of the algorithm, is to let the algorithm optimize the parameters *on the fly* (we give more details and references on dynamic parameter choices in Sect. 1.2). Formally speaking, this is an even more complicated task, because instead of a single good parameter value now a suitable functional dependence of the parameter on the search history needs to be provided. Fortunately, a number of natural heuristics like the 1/5-th rule has proven to be effective in certain cases. In a sense, these are all *exogenous* parameter control mechanisms which are added to the evolutionary system.

An even more elegant way is to incorporate the parameter control mechanism into the evolutionary process, that is, to attach the parameter value to the individual, to modify it via (extended) variation operators, and to use the fitness-based selection mechanisms of the algorithm to ensure that good parameter values become dominant in the population. This *self-adaptation* of the parameter values has two main advantages: (1) It is generic, that is, the adaptation mechanism is provided by the algorithm, only the representation of the parameter in the individual and the extension of the variation operators have to be provided by the user. (2) It allows to re-use existing algorithms and much of the existing code.

Despite these advantages, self-adaptation is not used a lot in discrete evolutionary optimization. From the theory side, there exists some advice on how to set up such a self-adaptive system, but a real proof for its usefulness is still missing. This is the point we aim to make some progress on.

1.1 Our Results

The main result of this work is that we propose a version of the $(1, \lambda)$ evolutionary algorithm (EA) with a natural self-adaptive choice of the mutation rate. For

$\lambda \geq C \ln n$, C a sufficiently large constant, we prove that it optimizes the classic ONE-MAX benchmark problem in a runtime that is asymptotically optimal among all λ-parallel black-box optimization algorithms and that is better than the known runtimes of the $(1,\lambda)$ EA and the $(1+\lambda)$ EA for all static choices of the mutation rate. Compared to the (also asymptotically optimal) $(1+\lambda)$ EA with fitness-dependent mutation rate of Badkobeh, Lehre, and Sudholt [9] and the $(1+\lambda)$ EA with self-adjusting (exogenous) mutation rate of Doerr, Gießen, Witt, and Yang [23] the good news of our result is that this optimal runtime could be obtained in a generic manner. Note that both the fitness-dependent mutation rate of [9] and the self-adjusting rate of [23] with its mix of random and greedy rate adjustments would have been hard to find without a deeper understanding of the mathematics of these algorithms.

Not surprisingly, the proof of our main result has some similarity to the analysis of the self-adjusting $(1+\lambda)$ EA of [23]. In particular, we also estimate the expected progress in one iteration and use variable drift analysis. Also, we need a careful probabilistic analysis of the progress obtained from different mutation rates to estimate which rate is encoded in the new parent individual (unfortunately, we cannot reuse the analysis of [23] since it is not always strong enough for our purposes). The reason, and this is also the main technical challenge in this work, is that the $(1,\lambda)$ EA can lose fitness in one iteration. This happens almost surely when the mutation rate is too high. For this reason, we need to argue more carefully that such events do not happen regularly. To do so, among several new arguments, we also need a stronger version of the occupation probability result [47, Theorem 7] since (1) we need sharper probability estimates for the case that movements away from the target are highly unlikely and (2) for our process, the changes per time step cannot be bounded by a small constant. We expect our new results (Lemmas 6 and 7) to find other applications in the theory of evolutionary algorithms in the future. Note that for the $(1+\lambda)$ EA, an excursion into unfavorable rate regions is less a problem as long as one can show that the mutation rate returns into the good region after a reasonable time. The fact that the $(1,\lambda)$ EA can lose fitness also makes it more difficult to cut the analysis into regimes defined by fitness levels since it is now possible that the EA returns into a previous regime.

In this work, we also gained two insights which might be useful in the design of future self-adaptive algorithms.

Need for non-elitism Given the previous works, it would be natural to try a self-adaptive version of the $(1+\lambda)$ EA. However, this is risky. While the self-adjusting EA of [23] copes well with the situation that the current mutation rate is far from the ideal one and then provably quickly changes the rate to an efficient setting, a self-adaptive algorithm cannot do so. Since the mutation rate is encoded in the individual, a change of the rate can only occur if an offspring is accepted. For an elitist algorithm like the $(1+\lambda)$ EA, this is only possible when an offspring is generated that is good enough to compete with the parent(s). Consequently, if the parent individual in a self-adaptive $(1+\lambda)$ EA has a high fitness, but a detrimental (that is, large) mutation rate, then the algorithm is stuck with this individual for a long time. Already for the simple ONEMAX function, such a situation can lead to an exponential runtime.

Needless to say, when using a comma strategy we have to choose λ sufficiently large to avoid losing the current-best solution too quickly. This phenomenon has

been observed earlier, e.g., in [57] it is shown that $\lambda \geq (1 - o(1)) \log_{(e-1)/e}(n)$ is necessary for the $(1,\lambda)$ EA with mutation rate $1/n$ to have a polynomial runtime on any function with unique optimum. We shall not specify a precise leading constant for our setting, but also require that $\lambda \geq C \ln(n)$ for a sufficiently large constant C.

Tie-breaking towards lower mutation rates To prove our result, we need that the algorithm in case of many offspring of equal fitness prefers those with the smaller mutation rate. Given that the usual recommendation for the mutation rate is small, namely $\frac{1}{n}$, and that it is well-known that large rates can be very detrimental, it is natural to prefer smaller rates in case of ties (where, loosely speaking, the offspring population gives no hint which rate is preferable).

This choice is similar to the classic tie-breaking rule of preferring offspring over parents in case of equal fitness. Here, again, the fitness indicates no preference, but the simple fact that one is maybe working already for quite some time with this parent suggests to rather prefer the new individual.

1.2 Previous Works

This being a theoretical paper, for reasons of space we shall mostly review the relevant theory literature, and also this with a certain brevity. For a broader account of previous works, we refer to the survey [46]. For a detailed description of the state of the art in theory of dynamic parameter choices, we refer to the survey [14]. We note that the use of self-adaptation in genetic algorithms was proposed in the seminal paper [5] by Bäck. Also, we completely disregard evolutionary optimization in continuous search spaces due to the very different nature of optimization there (visible, e.g., from the fact that dynamic parameter changes, including self-adaptive choices, are very common and in fact necessary to allow the algorithms to approach the optimum with arbitrary precision).

The theoretical analysis of dynamic parameter choices started slow. A first paper [45] on this topic in 2006 demonstrated the theoretical superiority of dynamic parameter choices by giving an artificial example problem for which any static choice of the mutation rate leads to an exponential runtime, whereas a suitable time-dependent choice leads to a polynomial runtime. Four years later [6], it was shown that a fitness-dependent choice of the mutation rate can give a constant-factor speed-up when optimizing the LEADINGONES benchmark function (see [33, Sect. 2.3] for a simplified proof giving a more general result). The first super-constant speed-up on a classic benchmark function obtained from a fitness-dependent parameter choice was shown in [15], soon to be followed by the paper [9] which is highly relevant for this work. In [9], the $(1+\lambda)$ EA with fitness-dependent mutation rate was analyzed. For a slightly complicated fitness-dependent mutation rate, an optimization time of $O(n\lambda/ \log \lambda + n \log n)$ was obtained. Also, it was shown that no λ-parallel mutation-based unbiased black-box algorithm can have an asymptotically better optimization time.

Around that time, several successful self-adjusting ("on-the-fly") parameter choices were found and analyzed with mathematical means. In [49], a success-based multiplicative update of the population size λ in the $(1+\lambda)$ EA is proposed and it

is shown that this can lead to a reduction of the parallel runtime. A multiplicative update inspired by the 1/5-th success rule from evolution strategies automatically finds parameter settings [12] leading to the same performance as the fitness-dependent choice in [15]. Similar multiplicative update rules have been used to control the mutation strength for multi-valued decision variables [17] and the time interval for which a selected heuristic is used in [30]. A learning-based approach was used in [18] to automatically adjust the mutation strength and obtain the performance of the fitness-dependent choice of [19]. Again a different approach was proposed in [23], where the mutation rate for the $(1+\lambda)$ EA was determined on the fly by creating half the offspring with a smaller and half the offspring with a larger mutation rate than the value currently thought to be optimal. As new mutation rate, with probability $\frac{1}{2}$ the rate which produced the best offspring was chosen, with probability $\frac{1}{2}$ a random of the two rates was chosen. The three different exogenous approaches used in these works indicate that a generic approach towards self-adjusting parameter choices, such as self-adaptation, would ease the design of such algorithms significantly.

Surprisingly, prior to this work only a single runtime analysis paper for self-adapting parameter choices appeared. In [29], Dang and Lehre show several positive and negative results on the performance of a simple class of self-adapting evolutionary algorithms having the choice between several mutation rates. Among them, they show that such an algorithm having the choice between an appropriate and a destructively high mutation rate can optimize the LEADINGONES benchmark function in the usual quadratic time, whereas the analogous algorithm using a random of the two mutation rates (and hence in half the cases the right rate) fails badly and needs an exponential time. As a second remarkable result, they give an example setting where any constant mutation rate leads to an exponential runtime, whereas the self-adapting algorithm succeeds in polynomial time. As for almost all such examples, also this one is slightly artificial and needs quite some assumptions, for example, that all λ individuals are initialized with the 1-point local optimum. Nevertheless, this result makes clear that self-adaptation can outperform static parameter choices. In the light of this result, the main value of our results is showing that asymptotic runtime advantages from self-adaptation can also be obtained in less constructed examples (of course, at the price that the runtime gap is not exponential).

To complete the picture on previous work relevant to ours, we finally quickly describe what is known on the performance of the most common mutation-based algorithms for the ONEMAX benchmark function. For the simple $(1+1)$ EA, the expected runtime of $\Theta(n \log n)$ was determined in [53] (upper bound) and [25] (lower bound, this result was announced already in 1998). For the $(1+\lambda)$ EA with $\lambda \le n^{1-\varepsilon}$, $\varepsilon > 0$ a constant, an expected runtime (number of fitness evaluations) of

$$\Theta\left(\frac{n\lambda \log \log \lambda}{\log \lambda} + n \log n \right)$$

was shown in [27, 42]. These two problems are also the only problems discussed in this work for which more precise bounds than asymptotic orders of magnitude have been obtained, however, with significant technical effort [20, 21, 36, 38, 39, 50, 59].

For the (μ+1) EA with polynomially bounded μ, the expected runtime is $\Theta(\mu n + n \log n)$, see [61]. Finally, the expected runtime of the (μ+λ) EA was recently [3] determined as

$$\Theta\left(\frac{n \log n}{\lambda} + \frac{n}{\lambda/\mu} + \frac{n \log^+ \log^+ \lambda/\mu}{\log^+ \lambda/\mu} \right),$$

where $\log^+ x := \max\{1, \log x\}$ for all $x > 0$.

For non-elitist algorithms, unsuitable choices of the parameters or selection mechanisms can lead to prohibitively large runtimes. The earliest runtime analysis of the (1,λ) EA with mutation rate $1/n$ on ONEMAX is due to Jägersküpper and Storch [44], who proved a phase transition from exponential to polynomial runtime in the regime $\lambda = \Theta(\log n)$, leaving a gap of at least 21 between the largest λ in the exponential regime and the smallest in the polynomial regime. This result was improved by Rowe and Sudholt [57], who determined the phase transition point to be the above-mentioned function $\log_{(e-1)/e}(n)$, up to lower order terms. Jägersküpper and Storch [44] also obtain a useful coupling result: if $\lambda \geq c \ln n$ for a sufficiently large constant $c > 0$, the stochastic behavior of the (1+λ) EA and (1,λ) EA with high probability are identical for a certain polynomial (with degree depending on c) number of steps, allowing the above-mentioned results about the (1+λ) EA to be transferred to the (1,λ) EA.

For the (μ,λ) EA with $\lambda \geq (1 + \varepsilon)e\mu$, $\varepsilon > 0$ a constant, and $\mu = \Omega(\log n)$ sufficiently large (as well as for many other non-elitist algorithms under suitable parameter settings), a runtime of $O(n\lambda \log \lambda)$ was shown in [28], which was improved to $O(n\lambda)$ in [10]. When $\lambda \leq (1 - \varepsilon)\mu$ and λ is at most polynomial in n, then the runtime of the (μ,λ) EA is exponential in n [48, Corollary 1]. More precise results on this threshold phenomenon were given in [4].

1.3 Techniques

One of the technical difficulties in our analysis is that our self-adaptive (1,λ) EA can easily lose fitness when the rate parameter is set to an unsuitable value. For this reason, we cannot use the general approach of the analysis of the self-adjusting (1+λ) EA in [23], which separated the analysis of the rate and the fitness by, in very simple words, first waiting until the rate is in the desired range and then waiting for a fitness improvement (of course, taking care of the fact that the rate could leave the desired range). To analyze the joint process of fitness and rate with its intricate interactions, we in particular use drift analysis with a two-dimensional distance function, that is, we map (e.g., in Lemma 22) the joint space of fitness and rate suitably into the non-negative integers in a way that the expected value of this mapping decreases in each iteration. This allows to use well-known drift theorems.

The use of two-dimensional potential functions is not new in the analysis of evolutionary algorithms. However, so far only very few analyses exist that use this technique with dynamic parameter values and among these results, we feel that ours, in particular, Lemma 22, are relatively easy to use. Again in very simple words, the distance function g defined in the proof of Lemma 22 is the fitness distance plus a

pessimistic estimate for the fitness loss that could be caused from the current rate if this is maladjusted). We thus hope that this work eases future analyses of dynamic parameter choices by suggesting ways to measure suitably the progress in the joint space of solution quality and parameter value.

To allow the reader to compare our two-dimensional drift approach with existing works using similar arguments, we briefly review the main works that use two- or more-dimensional potential functions. Ignoring that the artificial fitness functions used in [22, 25, 26, 62] could also be interpreted as n-dimensional potential functions, the possibly first explicit use of a two-dimensional potential function in the runtime analysis of randomized search heuristics can be found in [60, proof of Theorem 4], a work analyzing how simulated annealing and the Metropolis algorithm compute minimum spanning trees in a line of connected triangles. In such optimization processes, a solution candidate (which is a subset of the edge set of the graph) can have two undesirable properties. (1) The solution contains a *complete triangle*, so one of these three edges has to be removed on the way to the optimal solution. (2) The solution contains two edges of a triangle, but not the two with smallest weight. This case, called *bad triangle*, is the less desirable one as here one edge of the solution has to be replaced by the missing edge and hence the status of two edges has to be changed. It turns out that a simple potential function can take care of these two issues, namely twice the number of bad triangles plus the number of complete triangles.

When analyzing non-trivial parent populations, then often it does not suffice to measure the quality of the current state via the maximum fitness in the population, but also the number of individuals having this best fitness has to be taken into account. This was first done in the analysis of the $(\mu+1)$ EA in [61]. Since in a run of this algorithm the population never worsens (in a strong sense), the progress could be analyzed conveniently via arguments similar to the fitness level method. Consequently, it was not necessary to define an explicit potential function. In a similar fashion, the $(N + N)$ EA [11] and the $(\mu+\lambda)$ EA [3] were analyzed by regarding the maximum fitness and the number of individuals having this fitness.

In [51], a vaguely similar approach was taken for non-elitist population-based algorithms. However, the fact that these algorithms may lose the current-best solution required a number of non-trivial modifications, most notably, (1) that the potential is based on the maximum fitness such that at least a proportion of γ of the individuals have at least this fitness (for a suitable constant $0 < \gamma < 1$) instead of the maximum fitness among all individuals, and (2) that the arguments resembling the fitness level method had to be replaced by a true drift argument. This approach was extended in [28] to give a general "level-based" runtime analysis result. A simplified version of this level theorem was recently given in [10].

What comes closest to our work with respect to the use of two-dimensional potential functions is [17], where a self-adjusting bit-wise mutation strength for functions defined not over bit strings, but over $\{0, \dots, r-1\}^n$ for some $r > 2$ is discussed. The potential function defined in (6) in [17, Sect. 7] is too complicated to be described here in detail, but it also follows the pattern used in this work, namely that the potential (to be minimized) is the sum of the fitness distance and a penalty for mutation strengths deviating from their currently ideal value. This potential

function, however, does not admit an easy interpretation of the type "fitness distance plus expected damage from improper mutation strength" as in our work. Consequently, the proof that indeed the desired progress is obtained with respect to this potential function is a lengthy (more than 4 pages) case distinction. Apparently unaware of the conference version [16], a similar approach, also with a slightly complicated potential function, was developed in [1] to analyze the (1+1) ES with 1/5 success rule.

A very general approach was recently published in [56]. When a process $X_0, X_1, \ldots$ admits several distance functions $d_1, \ldots, d_m$ such that, for all $i \in [1 \ldots m]$, the i-th distance satisfies $E[d_i(X_{t+1}) \mid X_t] \leq A(d_1(X_t), \ldots, d_m(X_t))^\top$ for a given matrix A, then under some natural conditions the first time until all distances are zero can be bounded in terms of a suitable eigenvalue of A. The assumptions on the distance functions and the matrix A are non-trivial, but [56] provides a broad selection of applications of this method. For our problem, we would expect that this method can be employed as well, however, this would also need an insight similar to the main insight of our approach, namely that the expected new fitness can be estimated in a linear fashion from the current fitness and the distance of the current rate from the ideal value.

1.4 Organization of This Work

This paper is structured as follows. In Sect. 2, we define the self-adaptive (1,λ) EA proposed in this work. In Sect. 3 we provide the technical tools needed on our analysis, among them two new results on occupation probabilities. Sect. 4 presents the main theorem. Its proof considers two main regions of different fitness, which are dealt with in separate subsections. We present some experimental results in Sect. 5 and finish with some conclusions in the last section.

2 The (1,λ) EA With Self-adapting Mutation Rate

We now define precisely the (1,λ) EA with self-adaptive mutation rate proposed in this work. This algorithm, formulated for the *minimization* of pseudo-boolean functions $f : \{0, 1\}^n \to \mathbb{R}$, is stated in Algorithm 1. We repeat that the main point of such a self-adaptive algorithm, as compared to other ways of modifying parameter values during a run, is that the parameter is encoded with the individuals, and is thus modified via the existing variation-selection cycle of the EA.

To encode the mutation rate into the individual, we extend the individual representation by adding the rate parameter. Hence the extended individuals are pairs (x, r) consisting of a search point $x \in \{0, 1\}^n$ and the rate parameter r, which shall indicate that r/n is the mutation rate this individual was created with.

The extended mutation operator first changes the rate to either r/F or Fr with equal probability ($F > 1$). It then performs standard bit mutation with the new rate.

In the selection step, we choose from the offspring population an individual with best fitness. If there are several such individuals, we prefer individuals having the

smaller rate r/F, breaking still existing ties randomly. In this winning individual, we replace the rate by F if it was smaller than F to ensure that in the next iterations, the lower of the two rates is at least 1. We replace the rate by $r_{max} = F^{\lfloor \log_F(n/(2F)) \rfloor}$, that is, the largest power of F not exceeding $n/(2F)$, if it was larger than this number. This ensures that in the next iteration, the larger of the two rates is not larger than $n/2$ and that the rate remains a power of F despite the cap.

We formulate the algorithm to start with any initial mutation rate r^{init} such that $F \leq r^{init} \leq n/(2F)$ and r^{init} is a power of F. For the result we shall show in this work, the initial rate is not important, but without this prior knowledge we would strongly recommend to start with the smallest possible rate $r^{init} = F$. Due to the multiplicative rate adaptation, the rate can quickly grow if this is profitable. On the other hand, a too large initial rate might lead to an erratic initial behavior of the algorithm.

For the adaptation parameter, we shall use $F = 32$ in our runtime analysis. Having such a large adaptation parameter eases the already technical analysis, because now the two competing rates r/F and Fr are different enough to lead to a significantly different performance (this is helpful, e.g., in the proof of Lemma 10). For a practical application, we suspect that a smaller value of F is preferable as it leads to a more stable optimization process. The choice of the offspring population size depends mostly on the degree of parallelism one wants to obtain. Clearly, λ should be at least logarithmic in n to prevent a too quick loss of the current-best solution. For our theoretical analysis, we require $\lambda \geq C \ln n$ for a sufficiently large constant C.

Algorithm 1 The $(1,\lambda)$ EA with self-adapting mutation rate, adaptation parameter $F > 1$, and initial mutation rate r^{init}/n such that $r^{init} \in [F, n/(2F)]$ and $r^{init} = F^i$ for some $i \in \mathbb{N}$.

Select x_0 uniformly at random from $\{0,1\}^n$.
Set $r_0 \leftarrow r^{init}$.
for $t \leftarrow 1, 2, \ldots$ **do**
    for $i \leftarrow 1, \ldots, \lambda$ **do**
        Choose $r_{t,i} \in \{r_{t-1}/F, Fr_{t-1}\}$ uniformly at random.
        Create $x_{t,i}$ by flipping each bit in x independently with probability $r_{t,i}/n$.
    Choose $i \in [1..\lambda]$ such that $f(x_{t,i}) = \min_{j \in [1..\lambda]} f(x_{t,j})$; in case of a tie, prefer an i with $r_{t,i} = r_{t-1}/F$; break remaining ties randomly.
    $(x_t, r_t) \leftarrow (x_{t,i}, r_{t,i})$.
    Replace r_t with $\min\{\max\{F, r_t\}, F^{\lfloor \log_F(n/(2F)) \rfloor}\}$.

The main result of this work is a mathematical runtime analysis of the performance of the algorithm proposed above on the classic benchmark function ONEMAX : $\{0,1\}^n \to \mathbb{R}$ defined by ONEMAX$(x) = \sum_{i=1}^{n} x_i$ for all $x = (x_1, \ldots, x_n) \in \{0,1\}^n$. Since such runtime analyses are by now a well-established way of understanding the performance of evolutionary algorithms, we only briefly give the most important details and refer the reader to the textbook [41].

The aim of runtime analysis is predicting how long an evolutionary algorithm takes to find the optimum or a solution of sufficient quality. As implementation-independent performance measure usually the number of fitness evaluations performed in a run of the algorithm is taken. More precisely, the *optimization time* of an algorithm on some problem is the number of fitness evaluations performed until for the first time an optimal solution is evaluated. Obviously, for a $(1,\lambda)$ EA, the optimization time is essentially λ times the number of iterations performed until an optimum is generated.

As in classic algorithms analysis, our main goal is an asymptotic understanding of how the optimization time depends on the problems size n. Hence all asymptotic notation in the paper will be with respect to n tending to infinity.

3 Technical Tools

In this section, we list several tools which are used in our work. Most of them are standard tools in the runtime analysis of evolutionary algorithms, however, we also prove two new results on occupation probabilities at the end of this section.

3.1 Elementary Estimates

We shall frequently use the following estimates.

Lemma 1

(a) *For all $x \in \mathbb{R}, 1 + x \leq e^x$.*
(b) *For all $x \in [0, \frac{2}{3}], e^{-x-x^2} \leq 1 - x$. Moreover, for all $x \in [0, \frac{1}{2}], e^{-3x/2} \leq 1 - x$.*
(c) *Weierstrass product inequality: For all $p_1, \ldots, p_n \in [0, 1]$,*

$$1 - \sum_{i=1}^{n} p_i \leq \prod_{i=1}^{n} (1 - p_i).$$

All these estimates can be proven via elementary means. We note that the second estimate was proven in [23, Lemma 8(c)]. The third is usually proven via induction, a possibly more elegant proof via the union bound was given in [34].

3.2 Probabilistic Tools

In our analysis, we use several standard probabilistic tools including Chernoff bounds. All these can be found in many textbook or the book chapter [34]. We mention the following variance-based Chernoff bound due to Bernstein [7], which is less common in this field (but can be found as well in [34]).

Theorem 2 *Let $X_1, \ldots, X_n$ be independent random variables. Let b be such that $E(X_i) - b \leq X_i \leq E(X_i) + b$ for all $i = 1, \ldots, n$. Let $X = \sum_{i=1}^{n} X_i$. Let $\sigma^2 = \sum_{i=1}^{n} \text{Var}(X_i) = \text{Var}(X)$. Then for all $\lambda \geq 0$,*

$$\Pr(X \geq E(X) + \lambda) \leq \exp\left(-\frac{\lambda^2}{2(\sigma^2 + \frac{1}{3}b\lambda)}\right),$$

$$\Pr(X \leq E(X) - \lambda) \leq \exp\left(-\frac{\lambda^2}{2(\sigma^2 + \frac{1}{3}b\lambda)}\right).$$

We shall follow the common approach of estimating the expected progress and translating this via so-called drift theorems into an estimate for the expected optimization time. We use the variable drift theorem independently found in [43, 52] in a slightly generalized form.

Theorem 3 (Variable Drift, Upper Bound) *Given a stochastic process, let $(X_t)_{t\geq 0}$ be a sequence of random variables obtained from mapping the random state at time t to a finite set $S \subseteq \{0\} \cup [x_{\min}, x_{\max}]$, where $x_{\min} > 0$. Let T be the random variable that denotes the earliest point in time $t \geq 0$ such that $X_t = 0$. If there exists a monotone increasing function $h(x) : [x_{\min}, x_{\max}] \to \mathbb{R}^+$ such that for all $x \in S$ with $\Pr(X_t = x) > 0$ we have*

$$E(X_t - X_{t+1} \mid X_t = x) \geq h(x)$$

then for all $x' \in S$ with $\Pr(X_0 = x') > 0$

$$E(T \mid X_0 = x') \leq \frac{x_{\min}}{h(x_{\min})} + \int_{x_{\min}}^{x'} \frac{1}{h(x)} \, dx.$$

Finally, we mention an elementary fact which we shall use as well. See [13, Lemma 1] for a proof.

Lemma 4 *Let $X \sim \text{Bin}(n, p)$ and $k \in [0 \dots n]$. Then $E(X \mid X \geq k) \leq E(X) + k$.*

3.3 Occupation Probabilities

To analyze the combined process of fitness and rate in the parent individual, we need a tool that translates a local statement, that is, how the process changes from one time step to the next, into a global statement on the occupation probabilities of the process. Since in our application the local process has a strong drift to the target, Theorem 7 from [47] is too weak. Also, we cannot assume that the process in each step moves at most some constant distance. For that reason, we need the following stronger statement.

Theorem 5 (Theorem 2.3 in [37]) *Suppose that $(\mathcal{F}_k)_{k\geq 0}$ is an increasing family of sub-σ-fields of $\mathcal{F}$ and $(Y_k)_{k\geq 0}$ is adapted to $(\mathcal{F}_k)$. If*

$$E\left(e^{\eta(Y_{k+1}-Y_k)}; Y_k > a \,\Big|\, \mathcal{F}_k\right) \leq \rho \text{ and } E\left(e^{\eta(Y_{k+1}-1)}; Y_k \leq a \,\Big|\, \mathcal{F}_k\right) \leq D,$$

then

$$\Pr(Y_k \geq b \mid \mathcal{F}_0) \leq \rho^k e^{\eta(Y_0-b)} + \frac{1-\rho^k}{1-\rho} D e^{\eta(a-b)}.$$

We apply this theorem in the following lemma that fits into the case in this paper.

Lemma 6 *Consider a stochastic process X_t, $t \geq 0$, on $\mathbb{R}$ such that for some $p \leq 1/25$ the transition probabilities for all $t \geq 0$ satisfy $\Pr(X_{t+1} \geq X_t + a \mid X_t > 1) \leq p^{a+1}$ for all $a \geq -1/2$ as well as $\Pr(X_{t+1} \geq a + 1 \mid X_t \leq 1) \leq p^{a+1}$ for all $a \geq 0$. If $X_0 \leq 1$ then for all $t \geq 1$ and $k > 1$ it holds that*

$$\Pr(X_t \geq 1 + k) \leq 11(ep)^k.$$

Proof We aim at applying Theorem 5.

There are two cases depending on X_t: for $X_t \leq 1$, using the monotonicity of $e^{\lambda(X_{t+1}-1)}$ with respect to $X_{t+1} - 1$, we obtain

$$D(p, \lambda) := E(e^{\lambda(X_{t+1}-1)} \mid X_t \leq 1) \leq E(e^{\lambda \max\{\lceil X_{t+1}-1\rceil, 0\}} \mid X_t \leq 1)$$

$$= e^0 \Pr(X_{t+1} \leq 1 \mid X_t \leq 1) + \sum_{a=1}^{\infty} e^{\lambda a} \Pr(a < X_{t+1} \leq a + 1 \mid X_t \leq 1)$$

$$\leq e^0 + \sum_{a=1}^{\infty} e^{\lambda a} \Pr(X_t > a \mid X_t \leq 1),$$

using the assumption that $\Pr(X_{t+1} \geq a + 1 \mid X_t \leq 1) \leq p^{a+1}$ for all $a \geq 0$ then

$$D(p, \lambda) \leq 1 + \sum_{a=1}^{\infty} e^{\lambda a} p^a = 1 + \frac{e^\lambda p}{1 - e^\lambda p};$$

and for $X_t > 1$, using the monotonicity of $e^{\lambda(X_{t+1}-X_t)}$ with respect to $X_{t+1} - X_t$, we have

$$\rho(p, \lambda) := E(e^{\lambda(X_{t+1}-X_t)} \mid X_t > 1) \leq E(e^{\lambda \max\{\lceil 2(X_{t+1}-X_t)\rceil/2, -1/2\}} \mid X_t > 1)$$

$$= e^{-\lambda/2} \Pr\left(X_{t+1} - X_t \leq -\frac{1}{2} \,\Big|\, X_t > 1\right)$$

$$+ \sum_{a=0}^{\infty} e^{\lambda a/2} \Pr\left(\frac{a-1}{2} < X_{t+1} - X_t \leq \frac{a}{2} \,\Big|\, X_t > 1\right)$$

$$\leq e^{-\lambda/2} + \sum_{a=0}^{\infty} e^{\lambda a/2} \Pr\left(X_{t+1} - X_t > \frac{a-1}{2} \,\Big|\, X_t > 1\right),$$

using the assumption that $\Pr(X_{t+1} \geq X_t + a \mid X_t > 1) \leq p^{a+1}$ for all $a \geq -1/2$ then

$$\rho(p, \lambda) = e^{-\lambda/2} + \sum_{a=0}^{\infty} e^{\lambda a/2} p^{(a+1)/2} = \frac{p^{1/2}}{(e^\lambda p)^{1/2}} + \frac{p^{1/2}}{1 - (e^\lambda p)^{1/2}}.$$

Using $\lambda := \ln(1/(ep))$ such that $e^\lambda p = 1/e$, we have

$$\rho := \rho(p, \lambda) \leq e^{1/2} p^{1/2} + \frac{p^{1/2}}{1 - e^{-1/2}} \leq \frac{e^{1/2}}{5} + \frac{1/5}{1 - e^{-1/2}} < 0.84,$$
$$D := D(p, \lambda) \leq 1 + (1/e)/(1 - 1/e) < 1.6.$$

Theorem 5 yields with $a := 1$ and $b := 1 + k$ that

$$\Pr(X_t \geq 1 + k \mid X_0) \leq \rho^t e^{-\lambda(1+k-X_0)} + \frac{1}{1-\rho} D e^{-\lambda k}$$

$$\leq (ep)^k + \frac{1.6}{1 - 0.84} (ep)^k = 11(ep)^k.$$

$\square$

For the simpler case of a random process that runs on the positive integers and that has a strong negative drift, we have the following estimate for the occupation probabilities.

Lemma 7 *Consider a random process defined on the positive integers $1, 2, \ldots$. Assume that from each state i different from 1, only the two neighboring states $i - 1$ and $i + 1$ can be reached (and there is no self-loop on state i). From state 1, only state 2 can be reached and the process can stay on state 1. Let $p_i < 1$ be an upper bound for the transition probability from state i to state $i + 1$ (valid in each iteration regardless of the past). Assume that*

$$p_{i-1} \geq \frac{p_i}{1 - p_i}$$

holds for all $i \geq 2$. Assume that the process starts in state 1. Then at all times, the probability to be in state i is at most

$$q_i := \prod_{j=1}^{i-1} \frac{p_j}{1 - p_j},$$

where as usual we read the empty product as $q_1 = 1$.

Proof The claimed bound on the occupation probabilities is clearly true at the start of the process. Assume that it is true at some time. By this assumption and the assumptions on the process, the probability to be in state $i \geq 2$ after one step is at most

$$q_{i-1} p_{i-1} + q_{i+1} = q_{i-1} \left(p_{i-1} + \frac{p_{i-1}}{1 - p_{i-1}} \frac{p_i}{1 - p_i} \right)$$

$$\leq q_{i-1} \left(p_{i-1} + \frac{p_{i-1}}{1 - p_{i-1}} p_{i-1} \right)$$

$$= q_{i-1} \frac{p_{i-1}}{1 - p_{i-1}} = q_i.$$

Trivially, the probability to be in state 1 after one step is at most $q_1 = 1$. Hence, by induction over time, we see that q_i is an upper bound for the probability to be in state i at all times. $\square$

4 Main Result and Proof

We can now state precisely our main result and prove it.

Theorem 8 *Let $\lambda \geq C \ln n$ for a sufficiently large constant $C > 0$ and $\lambda = n^{O(1)}$. Let $F = 32$. Then the expected number of generations the self-adapting $(1,\lambda)$ EA takes to optimize* ONEMAX *is*

$$O\left(\frac{n}{\log \lambda} + \frac{n \log n}{\lambda}\right).$$

This corresponds to an expected number of fitness evaluations of $O(n\lambda / \log \lambda + n \log n)$.

The proof of this theorem is based on a careful, technically demanding drift analysis of both the current ONEMAX-value k_t (which is also the fitness distance, recall that our goal is the minimization of the objective function) and the current rate r_t of the parent. In very rough terms, a similar division of the run as in [23] into regions of large ONEMAX-value, the far region (Sect. 4.1), and of small ONEMAX-value, the near region (Sect. 4.2) is made. The middle region considered in [23] is subsumed under the far region here.

In the remainder of our analysis, we assume that n is sufficiently large, that $\lambda \geq C \ln n$ with a sufficiently large constant C, that $\lambda = n^{O(1)}$, and that $F = 32$.

4.1 The Far Region

In this section, we analyze the optimization behavior of our self-adaptive $(1,\lambda)$ EA in the regime where the fitness distance k is larger than $n/\ln \lambda$. Due to our assumption $\lambda \geq C \ln n$, it is very likely to have at least one copy of the parent among λ offspring when $r = O(\ln \lambda)$. Thus the $(1,\lambda)$ EA works almost the same as the $(1 + \lambda)$ EA when r is small, but can lose fitness in general. The following lemma is crucial in order to analyze the drift of the rate depending on k, which follows a similar scheme as with the $(1 + \lambda)$ EA proposed in [23].

Roughly speaking, the rate leading to optimal fitness progress is n for $k \geq n/2 + \omega(\sqrt{n \ln(\lambda)})$, $n/2$ for $k = n/2 \pm o(\sqrt{n \log(\lambda)})$, and then the optimal rate quickly drops to $r = \Theta(\log \lambda)$ when $k \leq n/2 - \varepsilon n$.

To ease statement of the following lemma, we define two fitness dependent bounds $L(k)$ and $R(k)$.

Definition 9 Let $n/\ln \lambda < k < n/2$. We define $L(k) := (F \ln(en/k))^{-1}$ and $U(k) := n(2n - k)/(22(n - 2k)^2)$.

According to the definition, both $L(k)$ and $R(k)$ monotonically increase when k increases.

Lemma 10 *Consider an iteration of the self-adaptive* $(1,\lambda)$ *EA with current fitness distance k and current rate r. Then*:

(a) *If* $n/\ln\lambda < k$ *and* $F \le r \le L(k)\ln\lambda$, *the probability that all best offspring have been created with rate Fr is at least* $1 - O(\ln^3(\lambda)/\lambda^{1/(4\ln\ln\lambda)})$.

(b) *If* $k < n/2$ *and* $n/(2F) \ge r \ge U(k)\ln\lambda$, *then the probability that all best offspring have been created with rate r/F is at least* $1 - \lambda^{1-(23/22)r/(U(k)\ln\lambda)}$.

Proof of Lemma 10 part (a) Let $q(k, i, r)$ and $Q(k, i, r)$ be the probability that standard bit mutation with mutation rate $p = r/n$ creates from a parent with fitness distance k an offspring with fitness distance exactly $k - i$ and at most $k - i$, respectively. Then

$$q(k, i, r) = \sum_{j=0}^{k-i} \binom{k}{i+j}\binom{n-k}{j} p^{i+2j}(1-p)^{n-i-2j} \tag{1}$$

and $Q(k, i, r) = \sum_{j=i}^{k} q(k, j, r)$. We aim at finding i such that $Q(k, i, Fr) \ge \ln(\lambda)/\lambda$ while $Q(k, i, r/F) = O(\ln^3(\lambda)/\lambda^{1+1/4(\ln\ln\lambda)})$. Then we use these to bound the probability that at least one offspring using rate Fr obtains a progress of i or more while at the same time all offspring using rate r/F obtains less than i progress. Let i^* be the largest i such that $Q(k, i, Fr) \ge \ln(\lambda)/\lambda$. Using the fact that $\ln(1 - x) \ge -x - x^2$ for all $0 < x \le 1/2$, note that our algorithm by definition never uses mutation rates larger than $1/2$, we notice that $(1 - Fp)^{n-i} \ge (1 - Fp)^n \ge e^{-Fr-(Fr)^2/n}$. By the assumption that $r \le L(k)\ln\lambda \le \ln\lambda$, we obtain $(Fr)^2/n = O(\ln^2\lambda/n) = o(1)$. Thus $(1 - Fp)^{n-i} = (1 - o(1))e^{-Fr}$. We also notice that $\binom{k}{i} = (k/i)((k-1)/(i-1))\cdots(k-i+1) > (k/i)^{i-1}(k-i) = (k/i)^i((k-i)i/k) > 2(k/i)^i$ for $2 < i < k-2$. Thus for $i > 2$ we can bound $Q(k, i, Fr)$ by

$$Q(k, i, Fr) \ge q(k, i, Fr) \ge \binom{k}{i}(Fp)^i(1-Fp)^{n-i} \ge \left(\frac{k}{i} \cdot \frac{Fr}{n}\right)^i e^{-Fr}, \tag{2}$$

where the second inquality follows from (1) by regarding the term for $j = 0$ only. Let $i = \max\{(F-1)r, \ln\lambda/(8\ln\ln\lambda)\}$. We prove $i^* \ge i$ by distinguishing between two cases according to which argument maximizes i.

If $i = \ln\lambda/(8\ln\ln\lambda)$, then $r \le i/(F-1)$ and $Fr \le 2i$. Referring to inequality (2) and using the fact that $k/n \ge 1/\ln\lambda$, $i < \ln\lambda$, and $\ln\ln(\lambda) > 1$, we obtain

$$\ln(Q(k, i, Fr)) \ge i\ln\left(\frac{k}{in}\right) - Fr \ge -i\ln(\ln^2\lambda) - 2i$$

$$= -2i\ln\ln\lambda - 2i > -4i\ln\ln\lambda = -\frac{\ln\lambda}{2} \ge \ln\left(\frac{\ln\lambda}{\lambda}\right)$$

and thus $Q(k, i, Fr) \ge \ln(\lambda)/\lambda$.

If $i = (F-1)r$, then $r \ge \ln\lambda/(8(F-1)\ln\ln\lambda)$ since F is a constant. Using $r \le L(k)\ln\lambda$, we obtain $\ln\lambda \ge \ln(en/k)Fr$ which is equivalent to $(k/en)^{Fr} \ge 1/\lambda$. Furthermore, $(k/n)^i > (k/n)^{Fr}$ since $i = (F-1)r < Fr$. Thus

$$Q(k, i, Fr) \geq \left(\frac{k}{i} \cdot \frac{Fr}{n}\right)^i e^{-Fr} \geq \left(\frac{F}{F-1}\right)^{(F-1)r} \left(\frac{k}{en}\right)^{Fr} \geq 2^r \left(\frac{k}{en}\right)^{Fr} \geq \frac{\ln \lambda}{\lambda}.$$

Since $Q(k, i, r)$ is decreasing in i, we obtain $i^* \geq \max\{(F-1)r, \ln \lambda/(8 \ln \ln \lambda)\}$. Using a Chernoff bound and recalling that the expected number of flipped bits is bounded by $FL(k) \ln \lambda \leq \ln \lambda/\ln(2e)$, we notice that $i^* \leq \ln \lambda$. This upper bound will be used to estimate $Q(k, i^*, Fr)/Q(k, i^*+1, Fr)$ in the following part of the proof.

We now prove that $Q(k, i^*, r/F) = o(1/\lambda)$. By comparing each component in $q(k, i, r/F)$ and $q(k, i, Fr)$, and applying Lemma 1 (b) to estimate $(1 - Fr/n)^n$ and $(1 - r/(Fn))^n$ with $r = O(\ln \lambda) = o(n^{1/2})$ for large enough n, we obtain

$$\frac{q(k, i, Fr)}{q(k, i, r/F)} \geq F^{2i} \frac{(1 - Fr/n)^n}{(1 - r/(Fn))^n}$$
$$\geq (1 - o(1))F^{2i} e^{-(F-1/F)r} > F^{2i} e^{-Fr}.$$

Therefore $Q(k, i^*, Fr)/Q(k, i^*, r/F) \geq F^{2i^*} e^{-Fr} = \exp(2i^* \ln F - Fr) \geq \exp(2i^* \ln F - Fi^*/(F-1)) > \exp(3i^*) > \lambda^{1/(4 \ln \ln \lambda)}$, where in the first inequality, we use the fact that $i^* \geq (F-1)r$. To prove $Q(k, i^*, r/F) = O(\ln^3(\lambda)/\lambda^{1+1/4(\ln \ln \lambda)})$, we first show $Q(k, i^*, Fr)/Q(k, i^*+1, Fr) = O(\ln^2 \lambda)$. Then we use this to bound $Q(k, i^*, Fr) = O(\ln^3(\lambda)/\lambda)$ according to the definition of i^*. Finally we obtain $Q(k, i^*, r/F) \leq Q(k, i^*, Fr)/\lambda^{1/(4 \ln \ln \lambda)} = O(\ln^3(\lambda)/\lambda^{1+1/4(\ln \ln \lambda)})$. It remains to bound $Q(k, i^*, Fr)/Q(k, i^*+1, Fr)$. We show that the majority of $q(k, i, r)$ are from the first $3r$ terms in the summation of equation (1). Let $q(k, i, r)_j$ denote the j-th item $\binom{k}{i+j}\binom{n-k}{j} p^{i+2j}(1-p)^{n-i-2j}$ in equation (1). Then

$$\frac{q(k, i, r)_{j+1}}{q(k, i, r)_j} = \frac{k-i-j}{i+j+1} \cdot \frac{n-k-j}{j+1} \cdot p^2 \cdot (1-p)^{-2} \leq \frac{r^2}{(i+j+1)(j+1)}.$$

If $j > 3r$, then $r^2/((i+j+1)(j+1)) < 1/9$, and thus

$$q(k, i, r) \leq \left(\sum_{j=0}^{3r} q(k, i, r)_j\right) + q(k, i, r)_{3r} \left(\sum_{j=3r+1}^{k-i} (1/9)^{j-3r}\right)$$
$$\leq \left(\sum_{j=0}^{3r} q(k, i, r)_j\right) + q(k, i, r)_{3r} \cdot \frac{1/9}{1-1/9}$$
$$= \left(\sum_{j=0}^{3r-1} q(k, i, r)_j\right) + \frac{9}{8} \cdot q(k, i, r)_{3r} \leq \frac{9}{8} \sum_{j=0}^{3r} q(k, i, r)_j.$$

We notice that

$$\frac{q(k, i+1, r)_j}{q(k, i, r)_j} = \frac{\binom{k}{i+j+1}\binom{n-k}{j} p^{i+2j+1}(1-p)^{n-i-2j-1}}{\binom{k}{i+j}\binom{n-k}{j} p^{i+2j}(1-p)^{n-i-2j}} = \frac{(k-i-j)p}{(i+j+1)(1-p)},$$

using the fact that $\sum_{j=0}^{3r} q(k,i,r)_j \le q(k,i,r) \le (9/8)\sum_{j=0}^{3r} q(k,i,r)_j$ for all (k,i,r), we compute

$$\frac{q(k,i^*+1,Fr)}{q(k,i^*,Fr)} \ge \frac{\sum_{j=0}^{3Fr} q(k,i^*+1,Fr)_j}{(9/8)\sum_{j=0}^{3Fr} q(k,i^*,Fr)_j} \ge \frac{8}{9} \cdot \frac{k-i^*-3Fr}{i^*+3Fr+1} \cdot \frac{p}{1-p}.$$

Since $i^* \ge (F-1)r$, $i^* \le \ln\lambda$, and $k \ge n/\ln\lambda = \omega(\ln\lambda)$, we obtain

$$\frac{q(k,i^*+1,Fr)}{q(k,i^*,Fr)} = \Omega\left(\frac{kp}{i^*}\right) = \Omega\left(\frac{kr}{i^*n}\right) = \Omega\left(\frac{1}{\ln^2\lambda}\right).$$

Consequently we have $q(k,i^*,Fr)/Q(k,i^*+1,Fr) \le q(k,i^*,Fr)/q(k,i^*+1,Fr) = O(\ln^2\lambda)$ and

$$\frac{Q(k,i^*,Fr)}{Q(k,i^*+1,Fr)} = 1 + \frac{q(k,i^*,Fr)}{Q(k,i^*+1,Fr)} = O(\ln^2\lambda).$$

So finally $Q(k,i^*,Fr) = O(\ln^3(\lambda)/\lambda)$ due to the definition of i^*, and

$$Q(k,i^*,r/F) \le \frac{Q(k,i^*,Fr)}{F^{2i^*}e^{-Fr}} = O\left(\frac{\ln^3\lambda}{\lambda \cdot \lambda^{1/(4\ln\ln\lambda)}}\right).$$

A simple union bound shows that with probability $1 - O(\ln^3(\lambda)/\lambda^{1/(4\ln\ln\lambda)})$, no offspring of rate r/F manages to obtain a progress of i^* or more. However, the probability that an offspring has rate Fr and obtains at least i^* progress is $\ln(\lambda)/(2\lambda)$. Thus the probability that no offspring generated with rate Fr achieves a progress of at least i^* is at most $(1 - \ln(\lambda)/(2\lambda))^\lambda \le \lambda^{-1/2} = o(\ln^3(\lambda)/\lambda^{1/(4\ln\ln\lambda)})$. This proves the first statement of the lemma. $\square$

Proof of Lemma 10 part (b) For $\tilde{r} \in \{r/F, Fr\}$ let the random variable $X(k,\tilde{r})$ denote the number of flipped bits in k ones and $Y(k,\tilde{r})$ denote the number of flipped bits in $n-k$ zeros when applying standard bit mutation with probability $p = \tilde{r}/n$. Let $Z(k,\tilde{r}) := Y(k,\tilde{r}) - X(k,\tilde{r})$ denote the improvement in fitness. Let $Z^*(k,\tilde{r})$ denote the minimal $Z(k,\tilde{r})$ among all offspring which apply rate $\tilde{r}$. $E(Z(k,\tilde{r})) = (n-k)\tilde{r}/n - k\tilde{r}/n = (n-2k)\tilde{r}/n$. Our aim is to find a β such that $\Pr(Z(k,r/F) \le \beta) = \Theta(1)$ while $\Pr(Z(k,Fr) \le \beta) = o(1/\lambda)$, and use this to obtain a high value for $\Pr(Z^*(k,r/F) < Z^*(k,Fr))$.

Let $\beta := E(Z(k,r/F))$. We notice that $\Pr(X(k,r/F) > E(X(k,r/F)) - 1) \ge 1/2$ since the median of binomial distribution $X(k,\ r/F)$ is $\lfloor E(X(k,r/F)) \rfloor$ or $\lceil E(X(k,r/F)) \rceil$. Applying Lemma 8 in [32] to $\Pr(Y(k,r/F) < E(Y(k,r/F)) - 1)$ with $E(Y(k,r/F)) = \Omega(\ln\lambda) = \omega(1)$ by assumption $r \ge U(k)\ln\lambda$ and $E(Y(k,r/F)) < (n-k)/2$, we obtain for n sufficiently large that

$$\Pr\Big(Y(k, r/F) < E(Y(k, r/F)) - 1 \Big)$$
$$\geq \frac{1}{2} - \sqrt{\frac{n-k}{2\pi \lfloor (n-k)p \rfloor (n-k - \lfloor (n-k)p \rfloor)}} > \frac{2}{5}. \tag{3}$$

Thus $\Pr(Z(k, r/F) \leq \beta) > (1/2)(2/5) = 1/5$. We use Bernstein's inequality (Theorem 2) to bound $\Pr(Z(k, Fr) \leq \beta)$ and obtain

$$\Pr\Big(Z(k, Fr) \leq E(Z(k, Fr)) - \Delta \Big) \leq \exp\left(-\frac{\Delta^2}{2(\mathrm{Var}(Z(k, Fr)) + \Delta/3)} \right) \text{ for all } \Delta > 0.$$

With $\Delta = E(Z(k, Fr)) - \beta = (n - 2k)(Fr/n - r/(Fn)) = (n - 2k)(F^2 - 1)r/(Fn)$ and $\mathrm{Var}(Z(k, Fr)) = Fr(1 - Fr/n) < Fr$, we compute

$$\Pr(Z(k, Fr) \leq \beta) \leq \exp\left(-\frac{1}{2} \cdot \frac{(F^2 - 1)^2 (n - 2k)^2 r^2}{F^2 n^2 (Fr + (n - 2k)(F^2 - 1)r/(3Fn))} \right)$$
$$= \exp\left(-\frac{1}{2} \cdot \frac{(F^2 - 1)^2 (n - 2k)^2 r}{F^3 n^2 + Fn(n - 2k)(F^2 - 1)/3} \right)$$
$$\leq \exp\left(-\frac{3}{2} \cdot \frac{(F^2 - 1)^2 (n - 2k)^2 r}{3F^3 n^2 + F^3 n(n - 2k)} \right)$$
$$= \exp\left(-\frac{3}{4} \cdot \frac{(F^2 - 1)^2 (n - 2k)^2 r}{F^3 n(2n - k)} \right).$$

Given $F = 32$ and $r \geq U(k) \ln \lambda$ then

$$\Pr(Z(k, Fr) \leq \beta) < \exp\left(-\frac{23.9(n - 2k)^2 r}{n(2n - k)} \right) < \lambda^{-\frac{23.9r}{22U(k)\ln\lambda}}.$$

With a simple union bound, we obtain $\Pr(Z^*(k, Fr) \leq \beta) < \lambda \Pr(Z(k, Fr) \leq \beta) < \lambda^{1-23.9r/(22U(k)\ln\lambda)}$. The probability that an offspring has rate r/F and obtains β is at least $(1/2)(1/5) = 1/10$. Thus the probability that no offspring generated with r/F has a Z-value of at least β is at most $(1 - 1/10)^\lambda = \exp(-\Theta(\lambda))$. Therefore $\Pr(Z^*(k, Fr) < Z^*(k, r/F)) < \lambda^{1-23.9r/(22U(k)\ln\lambda)}(1 - \exp(-\Theta(\lambda))) = o(\lambda^{1-23r/(22U(k)\ln\lambda)})$ which means with probability at least $1 - \lambda^{1-(23r/(22U(k)\ln\lambda))}$ all best offspring have been created with rate r/F. □

Lemma 10 will be crucial in order to bound the expected progress on fitness in the far region. We notice that $\ln \lambda = o(\sqrt{n})$ in the lemma we may allow $r > \ln \lambda$ when k is large and $r = \Theta(n)$ when $k = n/2 - \Theta(\sqrt{n \ln \lambda})$. It is easy to show a positive progress on fitness for $r < \ln \lambda$ since there will be sufficiently many offspring that do not flip zeros. When $r \geq \ln \lambda$ we expect all offspring to flip zeros, but we can still show a positive drift when $k > 7n/20$, as stated in the following lemma. The idea is that the standard variation of the number of flipping ones is $\sqrt{kr/n(1 - r/n)} = \Theta(\sqrt{r})$. This makes a deviation compensating bad flips among the remaining $n - 2k$ zeros likely enough.

Lemma 11 *Let $7n/20 \le k < n/2$ and $\alpha = 10^{-4}$. Assume $r \le \min\{n^2 \ln \lambda/(12 (n - 2k)^2), n/(2F)\}$. Assume that from a parent with fitness distance k we generate an offspring using standard bit mutation with mutation rate $p = r/n$. Then the probability that this offspring has a fitness distance of at most $k - s$ with $s := \alpha(\min\{\ln \lambda, r\} + (n - 2k)r/n)$, is at least $\lambda^{-0.98}$.*

Proof We first look at the case when $r < 1/(2\alpha)$. In this case $s \le \alpha(r + (n - 2k)r/n) \le \alpha(2r) < 1$. Then the probability that this offspring has a fitness distance of $k - 1 > k - s$ is at least

$$\binom{k}{1}\left(\frac{r}{n}\right)^1\left(1 - \frac{r}{n}\right)^{n-1} = \Theta(e^{-r}) = \omega(\lambda^{-0.98}).$$

Therefore it remains to consider $r \ge 1/(2\alpha)$.

Let random variables X and Y denote the number of flips in k one-bits and $(n - k)$ zero-bits, respectively, in an offspring using rate $p = r/n$. Then $X - Y$ is the decrease of fitness distance. X and Y follow binomial distributions $\mathrm{Bin}(k, p)$ and $\mathrm{Bin}(n - k, p)$, respectively. Let

$$B(x) := \Pr(X = x) = \binom{k}{x}p^x(1 - p)^{k-x} \text{ for all } x \in \{0, 1, \ldots, k\},$$

$$F(x) := \Pr(X \ge x) = \sum_{i=\lceil x \rceil}^{k} B(i) \text{ for all } x \in [0, k].$$

Since $r \ge 1/(2\alpha) \ge 5000$ and $n \le 20k/7$, then $p = r/n \ge 5000 \cdot 7/(20k) = 1750/k$. Using this and the fact that $p \le 1/(2F)$, we apply Lemma 8 in [32] and obtain

$$\Pr(X > E(X)) > \frac{1}{2} - \sqrt{\frac{k}{2\pi\lfloor kp \rfloor(k - \lfloor kp \rfloor)}} > \frac{1}{2} - \sqrt{\frac{1}{2kp}} > \frac{2}{5}$$

Similarly $\Pr(Y \le E(Y)) = 2/5$. Since $E(X - Y) = kp - (n - k)p = -(n - 2k)p$, we bound

$$\Pr(X - Y \ge s) \ge \Pr(X \ge E(X) + (n - 2k)p + s)\Pr(Y \le E(Y))$$

$$\ge \frac{2}{5}F(kp + (n - 2k)p + s).$$

Let $\delta := \lceil (n - 2k)p + s \rceil$, $u := kp$ and $\tilde{u} := \lceil u \rceil$. We notice that $u = rk/n \ge (1/(2\alpha))(7/20) = 1750$. Furthermore, we have $\delta < \tilde{u} - 2 < u$ since

$$\delta = \lceil (n-2k)p + s \rceil < (1+\alpha)(n-2k)p + \alpha \min\{\ln\lambda, r\} + 1$$

$$\leq (1+\alpha)\frac{n-2k}{n}r + \alpha r + 1 \leq \left((1+\alpha)\frac{3}{10} + \alpha\right)r + 1$$

$$= \frac{3+13\alpha}{10} \cdot \frac{n}{k} \cdot u + 1 < \frac{3+13\alpha}{10}(3u) + 1 < 0.91u + 1$$

$$= u - 0.09u + 1 \leq u - (0.09 \cdot 1750 - 1) = u - 156.5.$$

We aim at proving $F(u+\delta) = \omega(\lambda^{-0.98})$ to obtain this lemma. If $F(u+\delta) = \Theta(1)$ then the conclusion holds. It remains to consider $F(u+\delta) = o(1)$ while $F(u) - F(u+\delta) \geq 2/5 - o(1)$ as stated in equation (3). For any $x \in \mathbb{Z}_{\geq u}$ we have

$$\frac{B(x+1)}{B(x)} = \frac{k-x}{x+1} \cdot \frac{p}{1-p} \leq \frac{u-up}{u-up+1-p} < 1.$$

Since $\tilde{u} = \lceil u \rceil$ then $B(\tilde{u}) > B(\tilde{u}+1) > \cdots > B(k)$, and thus $F(u+\delta) \geq \delta B(\tilde{u}+2\delta)$ as well as $F(u) - F(u+\delta) \leq \delta B(\tilde{u})$. Using the fact that $p/(1-p) = u/(k-u)$ and $\tilde{u} - 1 < u$, we see that

$$\frac{B(\tilde{u}+2\delta)}{B(\tilde{u})} = \frac{(k-\tilde{u})\cdots(k-(\tilde{u}+2\delta)+1)}{(\tilde{u}+1)\cdots(\tilde{u}+2\delta)} \cdot \frac{p^{2\delta}}{(1-p)^{2\delta}}$$

$$\geq \frac{(k-(\tilde{u}-1)-2\delta)^{2\delta}}{(\tilde{u}+1)\cdots(\tilde{u}+2\delta)} \cdot \frac{u^{2\delta}}{(k-u)^{2\delta}} \geq \left(1 - \frac{2\delta}{k-u}\right)^{2\delta} \frac{u^{2\delta}}{(\tilde{u}+1)\cdots(\tilde{u}+2\delta)}.$$

We compute the following factorials using Robbins's Stirling's approximation in [55]

$$(\tilde{u}+2\delta)! \leq \sqrt{2\pi(\tilde{u}+2\delta)}\left(\frac{\tilde{u}+2\delta}{e}\right)^{\tilde{u}+2\delta} \exp\left(\frac{1}{12(\tilde{u}+2\delta)}\right),$$

$$\tilde{u}! \geq \sqrt{2\pi\tilde{u}}\left(\frac{\tilde{u}}{e}\right)^{\tilde{u}} \exp\left(\frac{1}{12\tilde{u}+1}\right).$$

Notice that $12\tilde{u} + 1 < 12(\tilde{u}+2\delta)$, we obtain

$$\frac{1}{(\tilde{u}+1)\cdots(\tilde{u}+2\delta)} = \frac{\tilde{u}!}{(\tilde{u}+2\delta)!} \geq \sqrt{\frac{\tilde{u}}{\tilde{u}+2\delta}}\frac{\tilde{u}^{\tilde{u}}e^{2\delta}}{(\tilde{u}+2\delta)^{\tilde{u}+2\delta}} \geq \sqrt{\frac{\tilde{u}}{\tilde{u}+2\delta}}\frac{u^{\tilde{u}}e^{2\delta}}{(\tilde{u}+2\delta)^{\tilde{u}+2\delta}}.$$

Therefore

$$\frac{B(\tilde{u}+2\delta)}{B(\tilde{u})} \geq \left(1 - \frac{2\delta}{k-u}\right)^{2\delta} \sqrt{\frac{\tilde{u}}{\tilde{u}+2\delta}}\frac{u^{\tilde{u}+2\delta}e^{2\delta}}{(\tilde{u}+2\delta)^{\tilde{u}+2\delta}}$$

$$= \sqrt{\frac{\tilde{u}}{\tilde{u}+2\delta}} \exp\left(2\delta\ln\left(1 - \frac{2\delta}{k-u}\right) + (\tilde{u}+2\delta)\ln\left(\frac{u}{\tilde{u}+2\delta}\right) + 2\delta\right)$$

$$\geq \sqrt{\frac{\tilde{u}}{\tilde{u}+2\delta}} \exp\left(2\delta\ln\left(1 - \frac{2\delta}{k-u}\right) + (\tilde{u}+2\delta)\ln\left(1 - \frac{2\delta+1}{\tilde{u}+2\delta}\right) + 2\delta\right).$$

We notice that $2\delta/(k-u) \leq 2\delta/(2Fu-u) = 2\delta/(63u) < 2/63 < 1/2$ and $(2\delta+1)/(\tilde{u}+2\delta) < (2\delta+1)/(3\delta+2) < 2/3$. Referring to Lemma 1, we compute

$$2\delta \ln\left(1 - \frac{2\delta}{k-u}\right) \geq -\frac{3}{2} \cdot \frac{4\delta^2}{k-u} = -\frac{6\delta^2}{u/p - u} \geq -\frac{6\delta^2}{2Fu - u} \geq -\frac{\delta^2}{10u},$$

$$(\tilde{u} + 2\delta)\ln\left(1 - \frac{2\delta + 1}{\tilde{u} + 2\delta}\right) \geq -(2\delta + 1) - \frac{(2\delta + 1)^2}{\tilde{u} + 2\delta} \geq -2\delta - \frac{4\delta^2}{u} - 3,$$

$$\frac{B(\tilde{u} + 2\delta)}{B(\tilde{u})} \geq \sqrt{\frac{\tilde{u}}{\tilde{u} + 2\delta}} \exp\left(-\frac{41\delta^2}{10u} - 3\right) \geq \sqrt{\frac{1}{3}} e^{-3} \exp\left(-\frac{41\delta^2}{10u}\right). \tag{4}$$

where the last inequality used that $\tilde{u}/(\tilde{u} + 2\delta) \geq 1/3$ since $\delta \leq \tilde{u}$. Using $n/k \leq 20/7$ and $r \leq n^2 \ln\lambda/(12(n-2k)^2)$, we obtain

$$\frac{41\delta^2}{10u} = \frac{41n}{10k} \cdot \frac{[(1+\alpha)((n-2k)/n)r + \alpha \min\{\ln\lambda, r\}]^2}{r}$$

$$\leq \frac{41n}{10k} \cdot \frac{((1+\alpha)((n-2k)/n)r + \alpha \min\{\ln\lambda, r\} + 1)^2}{r}$$

$$\leq \frac{41n}{10k} \cdot \left(\frac{((1+\alpha)((n-2k)/n)r + \alpha \min\{\ln\lambda, r\})^2}{r} + 2(1+\alpha)\frac{n-2k}{n} + 2\alpha + \frac{1}{r}\right)$$

$$\leq \frac{82}{7} \cdot \left((1+\alpha)^2\left(\frac{n-2k}{n}\right)^2 r + 2(1+\alpha)\frac{n-2k}{n}\alpha \ln\lambda + \alpha^2 \ln\lambda + 1\right)$$

$$\leq \frac{82}{7} \cdot \left(\frac{(1+\alpha)^2 \ln\lambda}{12} + \frac{3(1+\alpha)\alpha}{5}\ln\lambda + \alpha^2\ln\lambda + 1\right) < 0.978\ln\lambda + \frac{82}{7}.$$

Plugging the last estimate into inequality (4), we obtain $B(\tilde{u} + 2\delta)/B(\tilde{u}) = \omega(\lambda^{-0.98})$. Thus $F(u + \delta)/(F(u) - F(u + \delta)) = \omega(\lambda^{-0.98})$ and $F(u + \delta) = \omega(\lambda^{-0.98})$ which proves the statement in this lemma. □

For $k < 7n/20$, we need a more careful analysis, where we will estimate the expected progress on fitness averaged over the random rates the algorithm may have at a time. Hence, we assume a fixed current fitness but a random current rate and compute the average drift of fitness with respect to the distribution on the rates. This approach is similar to the one by Jägersküpper [40], who computes the average drift of the Hamming distance to the optimum when the (1+1) EA is optimizing a linear function, where the average is taken with respect to a distribution on all search points with a certain Hamming distance.

Of course, we want to exploit that a rate yielding near-optimal fitness progress is used most of the time such that too high (or too low) rates do not have a significant impact. To this end, Lemma 6 about occupation probabilities will be crucial.

We now define two fitness dependent bounds $r_l(k)$ and $r_u(k)$. We show in Lemma 13 that for any rate, if r/F or Fr is within the bounds, then the algorithm has logarithmic drift on fitness.

Definition 12 Let $n/\ln\lambda < k < n/2$. We define

$$r_u(k) := \begin{cases} n^2 \ln(\lambda)/(12(n-2k)^2) & \text{if } 7n/20 \leq k < n/2, \\ 10U(k)\ln(\lambda)/9 & \text{if } n/\ln\lambda < k < 7n/20. \end{cases}$$

$$r_l(k) := \begin{cases} L(k)\ln(\lambda)/2 & \text{if } n/\ln\lambda \leq k < n/2, \\ F & \text{if } n/\lambda < k < n/\ln\lambda. \end{cases}$$

where $L(k)$ and $U(k)$ are defined as in Definition 9.

We notice that Lemma 10 can be applied to all $r > r_u$ or $r < r_l$ because for all $7n/20 \leq k < n/2$, we have $r_u/(U(k)\ln\lambda) = 22n/(12(2n-k)) \geq 22/(12(2-0.35)) = 10/9$. For $k < n/\ln\lambda$, we set r_l to the minimal possible value of r. Finally note that r_u is non-decreasing in k due to the monotonicity of $n^2/(n-2k)^2$ and $U(k)$.

Lemma 13 *Let $n/\lambda < k < n/2$.*
Let $\Delta(k,r)$ denote the fitness gain of the best offspring using rate in $\{r/F, Fr\}$.

(a) *The negative drift of fitness for too high rates $r \geq Fr_u$ is bounded by*

$$E(\Delta(k,r)) \geq -\big(1+o(1)\big)\frac{n-2k}{n}\frac{r}{F}.$$

(b) *When $k \geq 7n/20$ the positive drift of fitness for good rate $r \leq Fr_u$ is bounded by*

$$E(\Delta(k,r)) \geq \big(1-o(1)\big)\cdot 10^{-4}\left(\frac{n-2k}{n}\cdot\frac{r}{F}+\min\left\{\ln\lambda, \frac{r}{F}\right\}\right).$$

(c) *When $n/\lambda < k < 7n/20$ the positive drift of fitness for good rate $r \leq Fr_u$ is bounded by*

$$E(\Delta(k,r)) \geq \big(1-o(1)\big)\min\left\{\frac{r}{F}, \frac{\ln\lambda}{F\ln(en/k)}\right\}.$$

Proof The probability of using rate r/F is 1/2. Thus with probability at least $1-(1/2)^\lambda = 1-o(1/n^3)$, at least one offspring uses rate r/F. For this offspring, the expected loss is $(n-2k)r/(Fn)$. If the complementary event (hereinafter called failure) of probability $o(1/n^3)$ happens, we estimate $\Delta(k,r)$ pessimistically by $-n$. This proves the first statement.

To prove the second item, we take $i = 10^{-4}((n-2k)r/(Fn)+\min\{\ln\lambda, r/F\})$. According to Lemma 11, the probability that an offspring uses rate r/F and achieves progress of i or more is at least $\lambda^{-0.98}/2$. Thus for λ offspring, we obtain $\Pr(\Delta(k,r) \geq i) \geq 1-(1-\lambda^{-0.98}/2)^\lambda = 1-O(\exp(-\lambda^{0.02}/2)) = 1-o(1)$. If the failure event happens, we estimate $\Delta(k,r)$ pessimistically by $-(n-2k)r/(Fn) = O(i)$. Thus the statement holds.

For the third item, we take $i := \min\{r, \ln(\lambda)/\ln(en/k)\}/F$. Notice that for $k < 7n/20$ we have $r_u(k) < r_u(7n/20) = (25/27)\ln\lambda < 0.93\ln\lambda$. Applying

Lemma 1(b) with $r/F \leq r_u(k) = o(\sqrt{n})$ we obtain $(1 - r/(Fn))^n \geq (1 - o(1))e^{-r/F}$. Therefore the probability that one offspring using rate $r/F < 0.93 \ln \lambda$ makes a progress of at least i is lower bounded by (assuming n large enough)

$$\binom{k}{i}\left(\frac{r}{Fn}\right)^i \left(1 - \frac{r}{Fn}\right)^n \geq \left(\frac{k}{i} \cdot \frac{r}{Fn}\right)^i \left((1 - o(1))e^{-\frac{r}{F}}\right)$$

$$> \left(\frac{k}{en}\right)^i e^{-0.94 \ln \lambda} \geq \lambda^{-1/F - 0.94} > \lambda^{-0.98}.$$

Thus for λ offspring, we obtain $\Pr(\Delta(k, r) \geq i) \geq 1 - (1 - \lambda^{-0.98}/2)^\lambda = 1 - o(1/\ln(\lambda))$. If the failure event happens we estimate $\Delta(k, r)$ pessimistically by $-(n - 2k)r/(Fn) = O(\ln \lambda)$. The contribution of failure events is $o(1)$ which is also $o(i)$. Therefore the third statement holds.   □

As discussed, our aim is to show that r_t/F or Fr_t stays in the right range frequently enough such that the overall average drift is still logarithmic. We notice that small rates $r_t < r_l$ intuitively do not have a negative effect, therefore we focus on the probability that $r_t < Fr_u$. Since r_u decreases monotonically in k, we need to analyze whether r still stays in the right range if there are large jumps in fitness distance k. Intuitively, the speed at which the mutation rate is decreased is much higher than than the decrease of fitness distance. To make this rigorous, we first look at the probability of large jumps, as detailed in the following lemma.

Lemma 14 *Assume $r \leq n/2$ and let $Z(k, r)$ denote the fitness-distance increase when applying standard bit mutation with probability $p = r/n$ to an individual with k ones. Then*

$$\Pr(Z(k, r) \leq (n - 2k)r/n - \Delta) \leq \exp\left(\frac{-\Delta^2}{2(1 - p)(r + \Delta/3)}\right),$$

$$\Pr(Z(k, r) \geq (n - 2k)r/n + \Delta) \leq \exp\left(\frac{-\Delta^2}{2(1 - p)(r + \Delta/3)}\right)$$

for all $\Delta \geq 0$.

Proof Without loss of generality, we assume that the individual has k leading ones and $n - k$ trailing zeros. Let random variables $Z_1, \ldots, Z_n$ be the contribution to fitness distance increase in each position after standard bit mutation. Then

$$\Pr(Z_i = -1) = p \text{ and } \Pr(Z_i = 0) = 1 - p \text{ for all } 1 \leq i \leq k;$$
$$\Pr(Z_i = 1) = p \text{ and } \Pr(Z_i = 0) = 1 - p \text{ for all } k < i \leq n.$$

The random variables $Z_1, \ldots, Z_n$ are independent and $Z(k, r) = \sum_{i=1}^n Z_i$. Similarly as in the proof of Lemma 10 (b), we have $E(Z(k, r)) = -kp + (n - 2k)p = (n - 2k)p$ and $\text{Var}(Z(k, r)) = \sum_{i=1}^n \text{Var}(Z_i) = np(1 - p) = (1 - p)r$. To apply Bernstein's inequality (Theorem 2), we construct $\tilde{Z}_i$ such that $\tilde{Z}_i = Z_i + p$ for all $1 \leq i \leq k$ and $\tilde{Z}_i = Z_i - p$ for all $k < i \leq n$. Therefore $E(\tilde{Z}_i) = 0$ and $\text{Var}(\tilde{Z}_i) = \text{Var}(Z_i)$.

$$\Pr(\tilde{Z}_i = -1 + p) = p \text{ and } \Pr(\tilde{Z}_i = p) = 1 - p \text{ for all } 1 \le i \le k;$$
$$\Pr(\tilde{Z}_i = 1 - p) = p \text{ and } \Pr(\tilde{Z}_i = -p) = 1 - p \text{ for all } k < i \le n.$$

By assuming $r \le n/2$, we have $p \le 1/2$ and thus $p - 1 \le \tilde{Z}_i \le 1 - p$ for all $1 \le i \le n$. Using the fact that $\sum_{i=1}^n \tilde{Z}_i = Z(k, r) - E(Z(k, r))$, Theorem 2 yields with $b := 1 - p$ and $\sigma^2 := (1 - p)pn = (1 - p)r$ that

$$\Pr\left(\sum_{i=1}^n Z(k, r) - E(Z(k, r)) \ge \Delta \right) \le \exp\left(\frac{-\Delta^2}{2(1 - p)(r + \Delta/3)} \right).$$

Similarly the lower tail bound holds. $\qquad\square$

We now use Lemma 14 to show that once $r_t \ge F r_u(k_t)$, there will be a strong drift for $r_t/r_u(k_t)$ to decrease down to 1.

Lemma 15 *Let* $k_t < n/2$. *Let* $\tau := \log_F(3/\sqrt{10})$ *and* $X_t := \log_F(r_t/r_u(k_t)) - \tau$ *with* $r_u(k_t)$ *defined in Definition* 12, *we have*

$$\Pr(X_{t+1} - X_t \ge a \mid X_t > 1) \le \lambda^{-\Omega(a+1)} \text{ for all } a \ge -1/2,$$
$$\Pr(X_{t+1} - 1 \ge a \mid X_t \le 1) \le \lambda^{-\Omega(a+1)} \text{ for all } a > 0.$$

Proof Using the fact that $r_{t+1} \in \{F r_t, r_t/F\}$, we see that

$$X_{t+1} - X_t \in \left\{ 1 + \log_F\left(\frac{r_u(k_t)}{r_u(k_{t+1})} \right), -1 + \log_F\left(\frac{r_u(k_t)}{r_u(k_{t+1})} \right) \right\}.$$

According to the monotonicity that $r_u(k)$ increases with respect to k, we notice that $k_t \ge k_{t+1}$ is a necessary condition for $X_{t+1} - X_t \ge 1$. We also notice that $X_t \ge \tau$ is equivalent to $r_t/r_u(k_t) \ge 3/\sqrt{10}$, which is sufficient to apply Lemma 10(b) since $r_u(k) \ge (10/9)U(k)\ln\lambda$ as defined in Definition 12.

We first consider the case $k_{t+1} \ge k_t$ (equivalent to $r_u(k_{t+1}) \ge r_u(k_t)$). In this case $X_{t+1} - X_t \le 1$ thus $\Pr(X_{t+1} \ge 1 \cap k_{t+1} \ge k_t \mid X_t < 0) = 0$ and $\Pr(X_{t+1} - 1 \ge 1 \cap k_{t+1} \ge k_t \mid X_t \le 1) = 0$. It remains to consider

$$\Pr(X_{t+1} - X_t \ge a \cap k_{t+1} \ge k_t \mid X_t > 1) \text{ with } -1/2 \le a \le 1, \text{ and}$$
$$\Pr(X_{t+1} - 1 \ge a \cap k_{t+1} \ge k_t \mid 0 \le X_t \le 1) \text{ with } 0 < a < 1.$$

If $r_{t+1} = r_t/F$ then $X_{t+1} - X_t \le -1$. Clearly $X_{t+1} - X_t \ge a \ge -1/2$ is impossible. It also makes $X_{t+1} \ge 1$ with $0 \le X_t \le 1$ impossible. Thus, the two probabilities above are bounded by $\Pr(r_{t+1} = F r_t \cap k_{t+1} \ge k_t \mid X_t \ge \tau) \le \Pr(r_{t+1} = F r_t \mid X_t \ge \tau) = \lambda^{-\Omega(1)}$ according to Lemma 10(b).

It remains to consider $k_{t+1} < k_t$ (equivalent to $r_u(k_{t+1}) < r_u(k_t)$). We make a case distinction based on the value of $(n - 2k_t)^2$.

Case 1: $(n - 2k_t)^2 < 2Fn \ln \lambda$. In this case, $r_u(k_t) = n^2 \ln \lambda / (12(n - 2k_t)^2) \geq n/(24F)$ which means that $X_t < 1$ for all rates $r \leq n/(2F)$. Thus $\Pr(X_{t+1} - X_t < a \cap k_{t+1} < k_t \mid X_t > 1) = 0$. When computing $\Pr(X_{t+1} - 1 \geq a \cap k_{t+1} < k_t \mid X_t \leq 1)$, we notice that $X_{t+1} \geq 1 + a$ implies $\log_F((n/2F)/r_u(k_{t+1})) \geq 1 + a + \tau$. Furthermore,

$$\frac{n/2F}{r_u(k_{t+1})} = \frac{12(n - k_{t+1})^2}{(2F)n \ln \lambda} \geq F^{1+a+\tau} = \frac{3F^{1+a}}{\sqrt{10}} \text{ if and only if } (n - k_{t+1})^2 \geq \frac{16F^{1+a}n \ln \lambda}{\sqrt{10}}.$$

Therefore a necessary condition for $X_{t+1} \geq 1 + a$ while $X_t \leq 1$ and $(n - k_t)^2 \leq 2Fn \ln \lambda$ is $k_t - k_{t+1} \geq ((4F^{(1+a)/2}/10^{1/4} - \sqrt{2F})/2)\sqrt{n \ln \lambda} > (6F^{a/2} - 4)\sqrt{n \ln \lambda}$. We notice that $E(k_{t+1} - k_t) > 0$, applying Lemma 14 and using a union bound we obtain for $\Delta := (6F^{a/2} - 4)\sqrt{n \ln \lambda} > 2\sqrt{n \ln \lambda}$ that

$$\begin{aligned}
\Pr(k_t - k_{t+1} > \Delta \mid X_t \leq 1) &= \Pr(k_{t+1} - k_t < -\Delta \mid X_t \leq 1) \\
&< \Pr(k_{t+1} - k_t < E(k_{t+1} - k_t) - \Delta \mid X_t \leq 1) \\
&< \lambda \exp\left(\frac{-\Delta^2}{2(n/2 + \Delta/3)}\right) < \lambda \exp\left(\frac{-\Delta^2}{n + \Delta}\right) = \lambda^{-\Omega(1+a)}.
\end{aligned}$$

Therefore $\Pr(X_{t+1} - 1 \geq a \cap k_{t+1} < k_t \mid X_t \leq 1) = \lambda^{-\Omega(1+a)}$.

Case 2: $(n - 2k_t)^2 \geq 2Fn \ln \lambda$. Let

$$\sigma_t^2 := r_u(k_t)/r_u(k_{t+1}) = (n - 2k_{t+1})^2/(n - 2k_t)^2,$$

then $X_{t+1} - X_t \in \{1 + \log_F(\sigma_t^2), -1 + \log_F(\sigma_t^2)\}$. We rewrite for $X_t > 1$ and $a \geq -1/2$

$$\begin{aligned}
&\Pr(X_{t+1} - X_t \geq a \cap k_{t+1} < k_t \mid X_t) \\
&\leq \Pr(r_{t+1} = r_t/F \cap \sigma_t^2 \geq F^{a+1} \mid X_t) + \Pr(r_{t+1} = Fr_t \cap \sigma_t^2 \geq F^{a-1} \mid X_t) \quad (5) \\
&\leq \Pr(\sigma_t^2 \geq F^{a+1} \mid X_t) + \Pr(\sigma_t^2 \geq F^{a-1} \mid X_t)\mathbb{1}_{a>2} + \Pr(r_{t+1} = Fr_t \mid X_t)\mathbb{1}_{a\leq2},
\end{aligned}$$

as well as for $X_t \leq 1$ and $a > 0$

$$\begin{aligned}
&\Pr(X_{t+1} - 1 \geq a \cap k_{t+1} < k_t \mid X_t) = \Pr(X_{t+1} - X_t \geq 1 + a - X_t \cap k_{t+1} < k_t \mid X_t) \\
&\leq \Pr(r_{t+1} = r_t/F \cap \sigma_t^2 \geq F^{a+2-X_t} \mid X_t) + \Pr(r_{t+1} = Fr_t \cap \sigma_t^2 \geq F^{a-X_t} \mid X_t) \\
&\leq \Pr(\sigma_t^2 \geq F^{a+1} \mid X_t) + \Pr(r_{t+1} = Fr_t \cap \sigma_t^2 \geq F^{a-X_t} \mid X_t),
\end{aligned}$$

$$(6)$$

where the second item in the above inequality (6) is furthermore bounded in (7) by making a distinction between $X_t \geq \tau \wedge a \leq 2$ and the remaining cases.

$$\begin{aligned}
&\Pr(r_{t+1} = Fr_t \cap \sigma_t^2 \geq F^{a-X_t} \mid X_t) \\
&\leq \Pr(\sigma_t^2 \geq F^{a-X_t} \mid X_t)\mathbb{1}_{X_t<\tau\vee a>2} + \Pr(r_{t+1} = Fr_t \mid X_t)\mathbb{1}_{X_t\geq\tau\wedge a\leq2}.
\end{aligned} \quad (7)$$

Applying Lemma 10(b) we see that both $\Pr(r_{t+1} = Fr_t \mid X_t)\mathbb{1}_{a\leq2}$ from (5) and $\Pr(r_{t+1} = Fr_t \mid X_t)\mathbb{1}_{X_t\geq\tau\wedge a\leq2}$ from (7) are of order $\lambda^{-\Omega(1)}$. This $\Omega(1)$ exponent is

sufficient to prove the lemma for $a \leq 2$. We also notice that the event $\sigma_t^2 \geq F^{a-\tau}$ subsumes all the other remaining events in inequalities (5), (6), and (7). Therefore it remains to validate $\Pr\left(\sigma_t^2 \geq F^{a-\tau} \mid X_t\right) \leq \lambda^{-\Omega(a+1)}$ for $a \geq 0$. To ease representation, let $s := F^{(a-\tau)/2} - 1 \geq F^{-\tau/2} - 1 = (10/9)^{1/4} - 1 > 1/40$. Since $s = \Omega(1+a)$, proving $\Pr\left(\sigma_t \geq 1 + s \mid X_t\right) = O\left(\lambda^{-\Omega(s)}\right)$ is sufficient to conclude the analysis of this case and therefore the lemma. We rewrite

$$\Pr(\sigma_t \geq 1 + s \mid X_t) = \Pr\left(\frac{n - 2k_{t+1}}{n - 2k_t} \geq 1 + s \mid X_t\right)$$

$$= \Pr(k_t - k_{t+1} \geq s(n - 2k_t)/2 \mid X_t).$$

Let $\Delta := (s/2 + p)(n - 2k_t)$ for $0 < p \leq 1/2$. Applying Lemma 14 and using a union bound we obtain

$$\Pr(\sigma_t \geq 1 + s \mid X_t) < \lambda \exp\left(\max_{0<p\leq1/2}\left\{\frac{-\Delta^2}{2(1-p)(pn+\Delta/3)}\right\}\right)$$

$$< \lambda \exp\left(\max_{0<p\leq1/2}\left\{\frac{-\Delta}{2(1+1/3)}1_{pn\leq\Delta} + \frac{-\Delta^2}{2(1-p)(pn)(1+1/3)}1_{pn>\Delta}\right\}\right)$$

$$< \lambda \exp\left(-\min_{0<p\leq1/2}\left\{\frac{\Delta}{3}1_{pn\leq\Delta} + \frac{\Delta^2}{3(1-p)(pn)}1_{pn>\Delta}\right\}\right)$$

We notice that $\Delta \geq (s/2)\sqrt{2Fn\ln(\lambda)} = 4s\sqrt{n\ln(\lambda)}$ and $(s/2 + p)^2/((1-p)p)$ attains the minimal value $s(2 + s) > 2s$ when $p = s/(2(s+1))$. Using the fact that $(n - 2k_t)^2/n \geq 2F\ln(\lambda)$ and $s > 1/40$,

$$\Pr(\sigma_t \geq 1 + s \mid X_t) < \lambda \exp\left(-\min\left\{s\sqrt{n\ln\lambda}1_{pn\leq\Delta} + \frac{(2s)2F\ln\lambda}{3}1_{pn>\Delta}\right\}\right)$$

$$< \lambda \exp\left(-\min\left\{s\sqrt{n\ln(\lambda)}1_{pn\leq\Delta} + 42s\ln(\lambda)1_{pn>\Delta}\right\}\right) = \lambda^{-\Omega(s)}.$$

$\square$

We finally use Lemmas 15 and 6 to obtain a logarithmic drift on average. After this major effort, it is a matter of a relatively straightforward drift analysis of fitness distance to obtain the following bound on the time to leave the far region.

Theorem 16 *The $(1,\lambda)$ EA with self-adapting mutation rate reaches a* ONEMAX-*value of $k \leq n/\lambda$ within an expected number of $O(n/\log \lambda)$ iterations, regardless of the initial mutation rate. Furthermore, with probability at least $1 - o(1)$, it holds $k_{t'} \leq 2n/\lambda$ and $r_{t'} \leq (7/9)\ln \lambda$ for some $t' = O(n/\log \lambda)$.*

Proof We first argue that within an expected number of $O(\sqrt{n})$ generations we will have $k_t < n/2$. Consider the case that $k_t \geq n/2$ and let the independent random variables X and Y denote the number of flips in k_t one-bits and $(n - k_t)$ zero-bits, respectively, in an offspring using rate $p = r/n$. Referring to [34] for $p \in [2/n, 1/2]$ we obtain, using similar arguments in the

proof Lemma 10(b) that $\Pr(X \geq E(X) + 1) = \Theta(1)$ and $\Pr(Y \leq E(Y)) = \Theta(1)$. Then $\Pr(X - Y \geq E(X) - E(Y) + 1) = \Theta(1)$. Since $E(X) \geq E(Y)$, the probability that an offspring choose rates $\tilde{r} \in \{r_t/F, Fr_t\}$ with $2 \leq \tilde{r} \leq n/2$ and have $X - Y \geq 1$ is at least $1/2 \cdot \Theta(1) = \Theta(1)$. Since the best of $\lambda = \Omega(\ln n)$ offspring is selected, the probability that $k_{t+1} \leq k_t - 1$ holds is at least $1 - \exp(-\Theta(\lambda)) = 1 - o(1/n^2)$. By an additive drift theorem, it takes $O(\max\{k_0 - n/2, 0\}) = O(\sqrt{n})$ iterations from the initial random search point to reach a parent with fitness distance less than $n/2$.

Without loss of generality, we can now assume $k_0 < n/2$. Consider the number of one-bits flips X and zero-bits flips Y in a parent with fitness distance $k_t < n/2$ and rate $2 \leq r < n/2$. As argued above $\Pr(X - Y \geq E(X) - E(Y) + 1) = \Theta(1)$. Since $k_t - (E(X) - E(Y)) = k_t - (k_t - (n - k_t))r/n = k_t(1 - r/n) + (n - k_t)(r/n) < n/2$ for all $r < n/2$, the probability that an offspring has fitness distance at most $n/2 - 1$ is $\Theta(1)$. Thus for $\lambda = \Omega(\ln n)$ offspring, we have $\Pr(k_{t+1} < n/2) \geq 1 - \exp(-\Theta(\lambda)) = 1 - o(1/n^2)$. Since that we aim at proving a hitting time of $O(n/\ln \lambda)$ and only consider phases of this length, we may furthermore assume $k_t < n/2$ for all $t \geq 0$, which only introduces an $o(1)$ error term by a union bound.

Define random variables $X_t := \log_F(r_t/r_u(k_t)) - \tau$ with $\tau = \log_F(3/\sqrt{10}) < 0$. We notice that when $(n - 2k_t)^2 \leq 2Fn \ln \lambda$ we have $X_t < 1$. If $r_t \geq Fr_u(k_t)$, according to Lemma 10(b), with probability $1 - o(1)$ we have $r_{t-1} = r_t/F$. Therefore within $O(\ln n)$ iterations we will obtain $X_t \leq 1$.

The idea of the remaining proof is to compute an average drift for any fixed distance using the distribution of mutation rates, and then to apply the variable drift theorem to obtain a runtime bound. Applying Lemmas 15 and 6 to the X_t, we see that

$$\Pr\left(r_t \geq F^{1+a+\tau} r_u(k_t)\right) \leq \lambda^{-\Omega(a)} \text{ for all } a > 0.$$

Let $r^{(i)}, i \in \mathbb{Z}$, denote the rate between $(F^i r_u(k), F^{i+1} r_u(k)]$ corresponding to fitness distance k. Thus, for all $i \geq 1$, we obtain

$$\Pr\left(r^{(i)}\right) \leq \Pr\left(r_t > F^i r_u(k_t)\right) \leq \Pr\left(r_t \geq F^{1+(i-1-\tau)+\tau} r_u(k_t)\right) \leq \lambda^{-\Omega(i-1-\tau)}.$$

According to Lemma 13, $E(\Delta(k, r^{(i)})) \geq -(1 + o(1))(n - 2k)r^{(i)}/n$ for $i \geq 1$ and $E(\Delta(k, r^{(0)})) \geq \Omega((n - 2k)r^{(0)}/n)$. The contribution of the negative drift is a lower order term compared to the contribution of the positive drift. Let $\Delta^{(k)}$ denote the average drift at distance k. We obtain

$$\Delta^{(k)} = \sum_{i \in \mathbb{Z}} E(\Delta(k, r^{(i)})) \Pr(r^{(i)}) \geq (1 - o(1)) \sum_{i \leq 0} E(\Delta(k, r^{(i)})) \Pr(r^{(i)}).$$

We notice that $\sum_{i \leq 0} \Pr(r^{(i)}) = 1 - o(1)$ and $E(\Delta(k, r^{(i)})) > 0$ for all $i \leq 0$. According to Lemma 10(a), with at least constant probability $r_t = \Omega(r_l(k_t))$. Since for any rate $r = \Omega(r_l(k))$ and $r \leq Fr_u$ the drift is $E(\Delta(k, r)) \geq \Theta(\ln(\lambda)/\ln(n/k_t))$ according to Lemma 13, the average drift satisfies

$$\Delta^{(k)} \geq \Theta(\ln(\lambda)/\ln(n/k)).$$

Using the variable drift theorem (Theorem 3) and the fact that

$$\int_{n/\lambda}^{n/2} \frac{\ln(n/k)}{\ln(\lambda)} \, dk = \frac{(k\ln(n) - k\ln(k) + k)\big|_{n/\lambda}^{n/2}}{\ln \lambda} = \frac{\Theta(n)}{\ln \lambda},$$

the expected time to reduce the fitness distance to at most n/λ conditioning on the assumption that $k_t < n/2$ for some $t = O(\sqrt{n})$ and $k_{t'} < n/2$ for all $t \leq t' = O(n/\log \lambda)$ is then $\Theta(n/\log \lambda)$. Thus the runtime bound of $O(n/\log \lambda)$ holds with probability $\Omega(1)$ due to Markov's inequality. Using a restart argument we then obtain the claimed expected runtime since the expected number of repetition of a phase of length $O(n/\log \lambda)$ is $O(1)$.

To prove the second statement of the theorem, we notice that the corresponding upper bound on the rate for $k_t = o(n)$ is $r_u(k_t) \leq (10/9)(U(k_t)) \ln \lambda = ((10/9)(2/22) + o(1)) \ln \lambda < (1/9) \ln \lambda$ and the occupation probability satisfies $\Pr(r_t \leq (7/9)F \ln \lambda \mid k_t = o(n)) \geq 1 - \lambda^{-\Omega(1)} = 1 - o(1)$. Therefore with probability $1 - o(1)$, the first iteration such that $k_t \leq 2n/\lambda$ has rate $r_t \leq (7/9) \ln \lambda$. We then argue for this iteration that with high probability it satisfies $r_{t+1} = r_t/F$ and $k_{t+1} \leq 2n/\lambda$. The probability of being no worse than parent using mutation probability $p \leq (7/9) \ln(\lambda)/n$ is at least $(1-p)^n \geq (1 - o(1))\lambda^{-7/9} > \lambda^{-8/9}$. Therefore,

$$\Pr(k_{t+1} \leq k_t \mid r_t \leq (7/9)F \ln \lambda) \geq 1 - \left(1 - \lambda^{-8/9}/2\right)^\lambda = 1 - o(1).$$

Furthermore $\Pr(r_{t+1} = Fr_t \mid k_t = o(n), r_t \geq (7/9) \ln \lambda) \leq \lambda^{1-(23/22)7} = o(1)$. Then we obtain an iteration with $k_t \leq 2n/\lambda$ and $r_t \leq (7/9) \ln \lambda$ with probability $1 - o(1)$. $\qquad\square$

4.2 The Near Region

We now analyze the regime in which the fitness distance satisfies $k = k_t = O(n/\lambda)$, the so-called near region. In this region, the probability that a fixed offspring created with rate r is better than its parent is $\Theta(\frac{1}{\lambda} \frac{r}{e^r})$, see Lemma 17. Consequently, the probability to make progress is only $\Theta(\frac{r}{e^r})$. This implies that the optimal rate r is constant (and by taking care of the constants, we shall see that the optimal rate value for the parent is $r = F$, the minimal possible value).

The superiority of small rate values is sufficiently strong to show that the rate drifts towards these values (Lemmas 19 and 20), however, for small values of λ we cannot show that in this regime, which takes at least an expected number of $\Omega(n/\lambda)$ iterations, it never happens that the rate increases to a value which lets all offspring be worse than the parent (this happens from $r \geq C\lambda$ for a suitable constant C on). Consequently, we cannot exclude the possibility that the algorithms loses fitness occasionally.

To analyze the progress of the algorithm (proof of Lemma 22), we devise a potential function based on the current fitness and rate and show that the expected progress with respect to this potential is high enough. This allows to use the multiplicative drift theorem to argue that within a desired time, we reach the optimum.

Naturally, we also have to argue that the process does not leave the near region except with small probability. This is done in Lemma 21.

We start with determining the probability of making progress in one mutation and similar events.

Lemma 17 *Let $0 < k \le 3n/\lambda$, and $r = o(\lambda^{1/4})$. Let $x \in \{0, 1\}^n$ with fitness distance $f(x) = k$. Let $y \in \{0, 1\}$ be obtained from x by flipping each bit independently with probability r/n. Consider the probabilities*

$$p_-(r) := \Pr(f(y) < f(x)),$$
$$p_0(r) := \Pr(f(y) = f(x)),$$
$$p'(r) := \Pr(\forall i \in [1 \ldots n] : x_i = 0 \implies y_i = 0),$$

that is, the probabilities that the offspring is better than the parent, that is is equally good, and that none of the 0-bits of the parent were flipped in the generation of the offspring.

Then

$$(1 - o(1))\frac{kr}{n}e^{-r} < p_-(r) < (1 + o(1))\frac{kr}{n}e^{-r},$$
$$(1 - o(1))e^{-r} < p_0(r) < (1 + o(1))e^{-r},$$
$$(1 - o(1))e^{-r} < p'(r) < (1 + o(1))e^{-r}.$$

Proof We regard the number X of flips in the k one-bits ("good flips" which reduce the fitness distance) and the number Y of flips in the $(n - k)$ zero-bits of the parent ("bad flips" which increase the fitness distance). Then $p_-(r)$ is at least

$$p_-(r) \ge \Pr(X = 1, Y = 0) = \frac{kr}{n}\left(1 - \frac{r}{n}\right)^{n-1} \ge (1 - o(1))\frac{kr}{n}e^{-r},$$

where the last estimate uses Lemma 1 (b).

Since $r = o(\lambda^{1/4})$, we have $kr/n = o(1)$, $kr^2/n = o(1)$, and $(kr^2/n)^{1.5} = o(kr/n)$. This allows to bound $p_-(r)$ from above by

$$p_-(r) < \Pr(X \in \{1, 2\}, Y = 0) + \sum_{i=3}^{2k-1} \Pr(X + Y = i, X > Y)$$

$$< \frac{kr}{n}\left(1 - \frac{r}{n}\right)^{n-1} + \frac{k^2r^2}{2n^2}\left(1 - \frac{r}{n}\right)^{n-2}$$

$$+ \sum_{i=3}^{2k-1}(i - 1)\left(\frac{r}{n}\right)^i\left(1 - \frac{r}{n}\right)^{n-i}\binom{k}{\lceil i/2 \rceil}\binom{n-k}{\lfloor i/2 \rfloor}$$

$$< \frac{kr}{n}\left(1 - \frac{r}{n}\right)^{n-2}\left(1 - \frac{r}{n} + \frac{kr}{2n}\right) + \sum_{i=3}^{2k-1}\left(\frac{r}{n}\right)^i\left(1 - \frac{r}{n}\right)^{n-i}(kn)^{i/2}$$

$$< (1 + o(1))\frac{kr}{n}\left(1 - \frac{r}{n}\right)^n + \sum_{i=3}^{2k-1}\left(\frac{kr^2}{n}\right)^{i/2}\left(1 - \frac{r}{n}\right)^{n-i}$$

$$< (1 + o(1))\frac{kr}{n}e^{-r}.$$

Similarly for $p_0(r)$ we have

$$p_0(r) > \Pr(X = Y = 0) = \left(1 - \frac{r}{n}\right)^n \geq (1 - o(1))e^{-r}.$$

Using again the fact that $kr^2/n = o(1)$, we have

$$p_0(r) = \Pr(X = Y = 0) + \sum_{i=1}^{k} \Pr(X = Y = i)$$

$$= \left(1 - \frac{r}{n}\right)^n + \sum_{i=1}^{k} \binom{k}{i} \binom{n-k}{i} \left(\frac{r}{n}\right)^{2i} \left(1 - \frac{r}{n}\right)^{n-2i}$$

$$< e^{-r} + \sum_{i=1}^{k} \left(\frac{kr^2}{n}\right)^i e^{-r} < (1 + o(1))e^{-r}.$$

Finally, for $p'(r)$ we compute $p'(r) = \Pr(Y = 0) = (1 - \frac{r}{n})^{n-k} = (1 \pm o(1))e^{-r}.$ □

Lemma 18 *Consider one iteration of the self-adaptive $(1,\lambda)$ EA starting with an individual of fitness distance k and rate $r = o(\lambda^{-1/4})$. Then the probability that there is an offspring which uses rate r/F and which inherits all 0-bits from the parent (and thus is at least as good as the parent), is at least $1 - \exp(-\frac{1}{2}\lambda(1 - o(1))e^{-r/F})$.*

Proof We compute

$$1 - \left(1 - \frac{1}{2}p'\left(\frac{r}{F}\right)\right)^{\lambda} \geq 1 - \left(1 - \frac{1}{2}(1 - o(1))e^{-r/F}\right)^{\lambda} \geq 1 - \exp\left(-\frac{1}{2}\lambda(1 - o(1))e^{-r/F}\right).$$

□

The following lemma is the counterpart of Lemma 10 (b), where now the optimal rate is the smallest possible value F. Again, we regard the event that all best offspring are created with the higher rate, since—due to our tie-breaking rule—only this leads to an increase of the rate. Different from Lemma 10 (b), now the probability of making a rate-increasing step is no $o(1)$ in general. If $k_t = \Theta(n/\lambda)$ and $r_t = O(1)$, we still have a small constant probability of increasing the rate.

Lemma 19 *Let $0 < k \leq 3n/\lambda$. The probability that all best offspring have been created with rate Fr is at most $(1 + o(1))\frac{\lambda k Fr}{n}e^{-Fr}$ when $r < \ln \lambda$. This probability is at most $\exp(-9r)$ for all r.*

Proof Let first $r < \ln \lambda$. According to Lemma 17,

$$p_-(Fr) \leq (1 + o(1))\frac{Fkr}{n}e^{-Fr} \quad \text{and} \quad p_0(r/F) \geq (1 - o(1))e^{-r/F}.$$

Therefore with probability at least $1 - \lambda p_-(Fr) = 1 - (1 + o(1))\frac{\lambda Fkr}{n}e^{-Fr}$, no offspring of rate Fr is better than its parent. Furthermore, by Lemma 18, with

probability at most $\exp(-(1 - o(1))\frac{1}{2}\lambda \exp(-r/F)) \leq \exp(-(1 - o(1))\frac{1}{2}\lambda^{1-1/F})$ there is no offspring using rate r/F and being equally good as its parent. Hence, the probability that a best offspring has been created with rate r/F is more than

$$1 - (1 + o(1))\frac{\lambda Fkr}{n}e^{-Fr} - \exp(-(1 - o(1))\tfrac{1}{2}\lambda^{1-1/F}) > 1 - (1 + o(1))\frac{\lambda Fkr}{n}e^{-Fr}.$$

Note that for $r < \ln \lambda$, the second bound follows from the first. If $r \geq \ln \lambda$, then the second bound follows from applying Lemma 10 to $U(k) = 1/11 + o(1)$. $\square$

We shall use the lemma above twice, first to bound the probability to have a certain rate (which will be needed to estimate the negative fitness drift) and second to estimate that a suitable two-dimensional drift is of the right order. We start with the occupation probability argument for the rate values.

Lemma 20 *Consider a run of the self-adaptive $(1,\lambda)$ EA started with some search point of fitness distance $k_0 \leq 2n/\lambda$ and rate $r_0 = F$. While the current search point of the algorithm has a fitness distance of at most $3n/\lambda$, the probability that the current rate is F^i is at most $\exp(-8F^{i-1})$ for all $i \in \mathbb{N}_{\geq 2}$.*

Proof If the current search point has fitness distance at most $3n/\lambda$ and the current rate is r, then by Lemma 19 the rate in the next iteration is Fr with probability at most $\exp(-9r)$; note that this estimate is not affected by a possible cap of the rate at $r_{\max}$.

Consequently, the random process describing the rates is such that from rate F^i, $i \in [1 \dots \log_F(r_{\max})]$, we go to rate F^{i+1} with probability at most $p_i = \exp(-9F^i)$. Otherwise, we go to rate F^{i-1} if $i \geq 2$ and stay at rate F if $i = 1$. By Lemma 7, note that we obviously have $p_i/(1 - p_i) \leq p_{i-1}$, in each iteration (such that the fitness distance has never gone above $3n/\lambda$) and for each $i \geq 2$ the probability q_i that the current rate is F^i is at most

$$q_i \leq \prod_{j=1}^{i-1} \frac{p_j}{1 - p_j} \leq \frac{p_{i-1}}{1 - p_{i-1}} \leq \exp(-8F^{i-1}).$$

$\square$

We use these occupation probabilities to estimate the drift away from the optimum ("negative drift"). From this we derive the statement that with high probability, the fitness distance does not increase to above $3n/\lambda$ in $n\lambda$ iterations.

Lemma 21 *Consider a run of the self-adaptive $(1,\lambda)$ EA started with some search point of fitness distance $k_0 \leq 2n/\lambda$ and rate $r_0 = F$. Then the probability that the process within the first $n\lambda$ iterations reaches a search point (as parent individual) with fitness distance more than $3n/\lambda$, is $o(1)$.*

Proof Denote by X_t the fitness distance at time t. We start by bounding the negative drift $E(\max\{0, X_t - X_{t-1}\})$ of the X process while it is at most $3n/\lambda$. If the current rate is r, then by Lemma 18 with probability at least $1 - \exp(-\frac{1}{2}\lambda(1 - o(1))e^{-r/F})$ there is an individual that used rate r/F and that did not flip any zero-bit into a one-bit. Let us call this event "A" and note that, naturally, under this event the drift cannot be negative as the individual without flipped zeros has at an least as good fitness as the parent.

We now analyze the case that A does not hold. Consider an individual conditional on that it uses rate r/F and at least one zero-bit was flipped into a one-bit. The number of such bad bits follows a distribution $(X \mid X \geq 1)$ with $X \sim \text{Bin}(n - X_{t-1}, r/Fn)$ and has expectation at most $1 + r/F$ by Lemma 4. For an individual using rate rF, the expected number of bad flips is $(n - k)\frac{rF}{n} \leq rF$. Consequently, noting that $1 + r/F \leq rF$ when $r \geq F$ and $F \geq \sqrt{2}$, the expected number of bad flips in all individuals (conditional on not A) is at most λrF and this is an upper bound on the negative drift.

In summary, in an iteration starting with rate r, the negative drift is at most

$$\lambda rF \exp(-\tfrac{1}{2}\lambda(1 - o(1))e^{-r/F}). \tag{8}$$

With Lemma 20, we can estimate the probability to have a certain rate. Hence the expected negative drift is

$$E(\max\{0, X_t - X_{t-1}\}) \leq \sum_{i=1}^{\log_F r_{\max}} \Pr(r = F^i)\lambda F^i F \exp(-\tfrac{1}{2}\lambda(1 - o(1))e^{-F^i/F})$$

$$\leq \sum_{i=2}^{\infty} \exp(-8F^{i-1})\lambda F^{i+1} \exp(-\tfrac{1}{2}\lambda(1 - o(1))e^{-F^{i-1}})$$

$$+ \lambda F^2 \exp(-\tfrac{1}{2}\lambda(1 - o(1))e^{-1}).$$

Note that[1] for $i \geq \lceil \log_F(\ln \lambda) + 1 - \frac{1}{5} \rceil = i^*$, we have $\lambda \leq \exp(2F^{i-1})$ and thus $\exp(-8F^{i-1})\lambda F^{i+1} = \exp(-(1 - o(1))8F^{i-1})\lambda \leq \exp(-(1 - o(1))6F^{i-1})$. Naturally, $\exp(-\tfrac{1}{2}\lambda(1 - o(1))e^{-F^{i-1}}) \leq 1$. Hence

$$\sum_{i=i^*}^{\infty} \exp(-8F^{i-1})\lambda F^{i+1} \exp(-\tfrac{1}{2}\lambda(1 - o(1))e^{-F^{i-1}}) \leq \sum_{i=i^*}^{\infty} \exp(-(1 - o(1))6F^{i-1})$$

$$\leq \exp(-(1 - o(1))6F^{i^*-1}) \leq \lambda^{-3(1-o(1))}.$$

For $i < \log_F(\ln \lambda) + 1 - \frac{1}{5}$, we have $\exp(-\tfrac{1}{2}\lambda(1 - o(1))e^{-F^{i-1}}) \leq \exp(-\tfrac{1}{2}(1 - o(1))\lambda^{1/2})$ and $\exp(-8F^{i-1})\lambda F^{i+1} = O(\lambda)$. Hence

---

[1] In this part of the proof, we use the fact that $F = 32$. This does not mean that for other not too small values of F we would not obtain similar results, but it increases the readability to work with this concrete value.

$$\sum_{i=1}^{i^*-1} \exp(-8F^{i-1})\lambda F^{i+1} \exp(-\tfrac{1}{2}\lambda(1-o(1))e^{-F^{i-1}})$$

$$\leq O(\log\log\lambda)O(\lambda)\exp(-\tfrac{1}{2}(1-o(1))\lambda^{1/2}) = o(\lambda^{-3}).$$

Consequently, $E(\max\{0, X_t - X_{t-1}\}) \leq \lambda^{-3(1-o(1))}$.

Define inductively $Y_0 = 0$ and $Y_t = Y_{t-1} + \max\{0, X_t - X_{t-1}\}$, if $\max\{X_s \mid s \in [0 \ldots t-1]\} \leq 3n/\lambda$ and $Y_t = Y_{t-1}$ otherwise. In other words, the Y process collects all the moves of the X process that go away from the optimum until the X process goes above $3n/\lambda$.

By our above computation, we have $E(Y_t) \leq t\lambda^{-3(1-o(1))}$. Consequently, by Markov's inequality, we have

$$\Pr(Y_t \geq t\lambda^{-2}) \leq \lambda^{-1+o(1)}$$

for all $t \in \mathbb{N}$. In particular, for $t = n\lambda$, we have $\Pr(Y_t \geq n/\lambda) \leq \lambda^{-1+o(1)}$. Note that $Y_t \leq n/\lambda$ implies $X_s \leq 3n/\lambda$ for all $s \leq t$.

Lemma 22 *Consider a run of the self-adaptive $(1,\lambda)$ EA started with some search point of fitness distance $k_0 \leq 2n/\lambda$ and rate $r_0 = F$. Then with probability at least $\frac{3}{4}$ there is a $T^* = O(n\ln(n/\lambda + 2)/\lambda)$ such that $k_{T^*} = 0$.*

Proof Since we are proving an asymptotic statement, we can assume that n is as large as we find convenient. Consider a run of the self-adjusting $(1,\lambda)$ EA from our starting position. Let T be the first time that the fitness distance is larger than $3n/\lambda$, if such a time exists, and $T = \infty$ otherwise. Let k_t denote the fitness distance at time t and r_t the rate used in iteration t, if $t \leq T$, and $(k_t, r_t) := (0, F)$ otherwise. We show that the process (k_t, r_t) reaches $(0, F)$ in time T^* with probability at least $1 - 1/e^2$.

We use a two-dimensional drift argument. Let $\gamma = 2F$ and define $g : \mathbb{N} \times \mathbb{N} \to \mathbb{R}$ by $g(k, r) = k + \gamma(r - F)$ for all k and r. We show that if for some t we have $(k, r) = (k_t, r_t)$, then $(k', r') := (k_{t+1}, r_{t+1})$ satisfies

$$E(g(k', r')) \leq g(k, r)(1 - \tfrac{\lambda}{10n}) \tag{9}$$

when assuming n to be sufficiently large.

There is nothing to show in the artificial case when $k > 3n/\lambda$ as we have, by definition, $g(k', r') = 0$ in this case. Among the interesting cases, we consider first that $r = F$. We obtain an improvement in fitness in particular if there is an offspring that uses rate $r/F = 1$, flips exactly one of the k missing bits, and flips no other bit. Hence the probability to make a positive fitness progress is at least

$$1 - (1 - \tfrac{1}{2}(1-\tfrac{1}{n})^{n-1}\tfrac{k}{n})^\lambda \geq 1 - (1 - \tfrac{k}{2en})^\lambda \geq 1 - \exp(-\tfrac{k\lambda}{2en}) \geq \tfrac{k\lambda}{3en},$$

where we used $(1-\tfrac{1}{n})^{n-1} \geq \tfrac{1}{e}$, $\tfrac{k\lambda}{2en} \leq \tfrac{3}{2e} < \tfrac{3}{2} \cdot \tfrac{1}{2}$ and Lemma 1 (b). The expected negative progress is at most $\lambda F^2 \exp(-(1+o(1))\tfrac{1}{2e}\lambda)$ as shown in (8). This negative

drift can be assumed to be $O(n^{-2})$ by taking the implicit constant in the assumption $\lambda = \Omega(\log n)$ large enough.

Consequently, $E(k') \leq k - \frac{1}{3e}\frac{\lambda k}{n} + O(n^{-2})$.

Regarding r', we note that by Lemma 19 we have $\Pr(r' = F^2) \leq (1 + o(1))\frac{\lambda k}{n}F^2 \exp(-F^2)$ and $r' = F$ otherwise. Hence $E(r') = F + (1 + o(1))(F - 1)F^3\frac{\lambda k}{n}\exp(-F^2)$. Consequently,

$$E(g(k,r) - g(k',r')) \geq \frac{1}{3e}\frac{\lambda k}{n} - O(n^{-2}) - \gamma(1 + o(1))(F - 1)F^3\frac{\lambda k}{n}\exp(-F^2)$$
$$= \frac{\lambda k}{n}(\frac{1}{3e} - \gamma(F - 1)F^3\exp(-F^2) - o(1))$$
$$\geq \frac{\lambda k}{n}\frac{1}{10} = g(k,r)\frac{\lambda}{10n}.$$

Let now be $r > F$. Note that the minimum fitness loss among the offspring is at most the minimum number of bits flipped, which in expectation is at most the number of bits flipped in the first offspring, which is exactly Fr. Consequently, we have $E(k') \leq k + Fr$. For r', we note that by Lemma 19, we have $r' = Fr$ with probability at most $\exp(-9r)$ and we have $r' = \frac{r}{F}$ otherwise. Consequently, $E(r') \leq Fr\exp(-9r) + \frac{r}{F}$. This yields

$$E(g(k,r) - g(k',r')) \geq -Fr + \gamma(r - Fr\exp(-9r) - \frac{r}{F})$$
$$\geq r(-F + \gamma - F\exp(-9F^2) - \frac{1}{F})$$
$$\geq r(-F + \gamma - \frac{2}{F}) \geq 31r = r + 30r \geq F^2 + 30r$$
$$\geq 32^2 + 30r \geq \frac{\lambda}{10n}(k + \gamma r) \geq g(k,r)\frac{\lambda}{10n},$$

where we used that $\lambda \leq 2n$; note that $\lambda > 2n$ gives $k_0 = 0$.

We have thus shown (9) for all (k, r). Since we start the process with a g-potential of at most $g(2n/\lambda, F) = \frac{2n}{\lambda}$, the multiplicative drift theorem with tail bounds [22, Theorem 5] gives that after $t = \lceil \frac{10n}{\lambda}(2 + \ln(\frac{2n}{\lambda})) \rceil$ iterations, we have $\Pr(g(k_t, r_t) > 0) \leq \frac{1}{e^2}$. Consequently, with probability $1 - \frac{1}{e^2}$, the potential is zero at time t, which implies $k_t = 0$ or $k_t > \frac{3n}{\lambda}$. By Lemma 21, note that $\lambda = \Omega(\log n)$ implies $t = O(n) = o(n\lambda)$, the probability that $k_t > \frac{3n}{\lambda}$ is $o(1)$, hence with probability at least $\frac{3}{4}$, we have indeed $k_t = 0$. □

Theorem 23 *Assume $k_0 \leq \frac{2n}{\lambda}$ and $r_0 \leq \frac{7}{9}\ln\lambda$. Then there is a $t = O(n\ln(n/\lambda + 2)/\lambda)$ such that with probability at least $\frac{1}{2}$, we have $k_t = 0$.*

Proof We first show that with good probability we quickly reach the initial situation of Lemma 22. The probability of observing $R^* - 1 := \log_F(r_0) - 1 \leq \log_F(\frac{7}{9}\ln\lambda) - 1$ rate-decreasing steps in a row by Lemma 19 is at least

$$\prod_{i=2}^{R^*}\left(1 - \exp(-9F^i)\right) \geq 1 - \sum_{i=2}^{R^*}\exp(-9F^i) \geq 1 - 0.001$$

by the Weierstrass product inequality (Lemma 1 (c)).

The probability of not flipping any zero-bits in at least one offspring, resulting in not increasing fitness distance, is for rate $r \leq \frac{7}{9} \ln \lambda$ at least $1 - \exp(-\frac{1}{2}\lambda(1 - o(1))e^{-r/F})$ by Lemma 18. By a union bound over $R^* - 1$ iterations, the probability of decreasing the initial rate to F in $O(\log \log \lambda)$ iterations without losing fitness is at least $1 - 0.001 - (R^* - 1)\exp(-\frac{1}{2}\lambda(1 - o(1))e^{-r/F}) \geq 5/6$ for sufficiently large n.

We can now apply Lemma 22 and obtain that with probability at least $\frac{3}{4}$ we have found the optimum within $t = O(n \ln(n/\lambda + 2)/\lambda)$ iterations. This shows the claim. □

4.3 Proof of Theorem 8

From the analyses of the two regimes in the previous two subsections, we now easily derive our main result, which is that the self-adaptive $(1,\lambda)$ EA optimizes ONEMAX within an expected number of $O(n\lambda/\log \lambda + n \log n)$ fitness evaluations when $\lambda = \Omega(\log n)$ is sufficiently large, λ is at most polynomial in n, and $F = 32$.

Proof Starting with an arbitrary initialization, Theorem 16 along with a Markov bound yield that with probability $\Omega(1)$ after $t = O(n/\log \lambda)$ iterations a search point is reached such that $k_t \leq 2n/\lambda$ and $r_t < 0.6(\ln \lambda)$. Assuming this to happen, the assumptions of Theorem 23 are satisfied. Hence, after another $O((n \log n)/\lambda)$ iterations the optimum is found with probability at least 1/2. Altogether, with probability $\Omega(1)$ the optimum is found from an arbitrary initial ONEMAX-value and rate within $T^* = O(n/\log \lambda + (n \log n)/\lambda)$ iterations. The claimed expected time now follows by a standard restart argument, more precisely by observing that after expected $O(1)$ repetitions of a phase of length T^* the optimum is found. □

5 Experiments

To gain some insight that cannot be derived from our asymptotic analysis, we performed a few numerical experiments. To this end we implemented the $(1,\lambda)$ EA in C++11 using the *default random engine* to generate pseudo-random numbers. The runtime is still measured via the number of generations until optimum is found.

We first see in Fig. 1 how fitness distance and mutation strength evolve in one run for $n = 100$, $\lambda = 12$ and $F = 1.2$. We used this small value of n to increase the readability of the figure, we used larger values for n in the remainder. Given the small value of n, we used a small mutation update factor of 1.2 instead of the value $F = 32$ used in our theoretical analysis. This run uses Algorithm 1 with $r^{\text{init}} = F$. We see that the algorithm prefers large mutation strengths at the beginning and small mutation strengths near the end of the optimization process. We also see that fitness distance can increase occasionally, in particular, when the rate is higher (in the plot, this happened in iteration 52 and iteration 88).
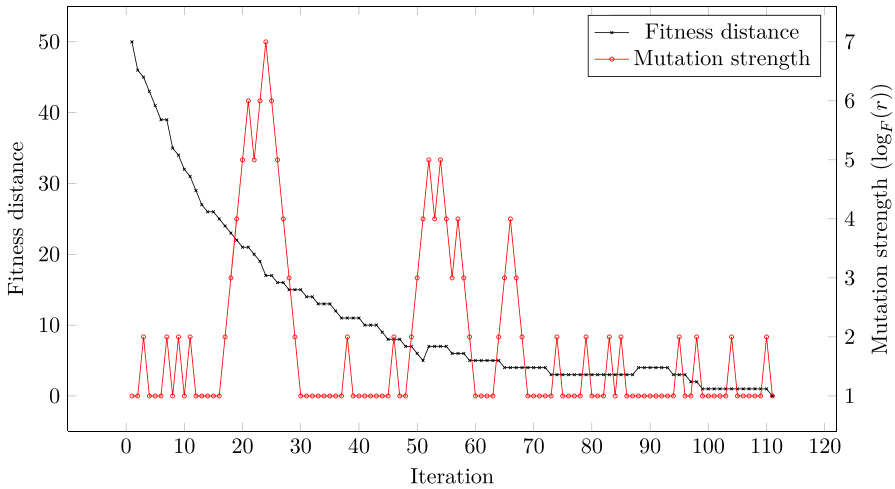
Fig. 1 Development of fitness distance and mutation strength in one run of self-adapting (1,λ) EA on ONEMAX ($n = 100$, $F = 1.2$, $\lambda = 12$)

In Fig. 2, we display the average runtime over 100 runs of different versions of the (1,λ) EA on ONEMAX for $n = 10^5$ and $\lambda = 100, 200, \ldots, 1000$ along with error bars for their standard deviation. For our self-adaptive (1,λ) EA (Algorithm 1), we used the update strengths $F \in \{1.2, 2, 32\}$. We did experiments also for $F = 1.05$, but the results were clearly inferior, so to avoid overloading this figure we do not visualize them. We always set the initial mutation strength to $r^{\text{init}} = F$. We further regard the classic (1,λ) EA using a static mutation rate of $\frac{1}{n}$ and the (1,λ) EA with fitness-dependent mutation rate $p = \max\{\frac{\ln \lambda}{n \ln(en/d)}, \frac{1}{n}\}$ as presented in [9].

The results clearly show that the update factor of $F = 32$ used in our mathematical analysis gives sub-optimal results for these values of λ and n. Recalling the working principle of the self-adaptive (1,λ) EA, this is not overly surprising. Even using the minimal possible rate $r = F$, the algorithm creates half of the offspring using an incredible large mutation probability of $F^2/n = 1024/n$. It is quite clear that this cannot be overly effective, but this can also be seen from the figure. The runtime of the self-adaptive (1,λ) EA with $F = 32$ is very close to the runtime of the static (1,λ) EA for half the λ-value, suggesting that half the offspring created by the self-adaptive (1,λ) EA, most likely the ones created with a mutation rate of F^2/n, had no impact on the process.

The results in Fig. 2 also show that the fitness-dependent mutation strength of [9] leads to a very good performance. In principle, of course, it is clear that the best fitness-dependent rate gives better results than any self-regulating rate since the latter needs to use also sub-optimal rates to find out what is the best rate. That the rate suggested in [9], a paper mostly concerned with asymptotic runtimes, shows such good results, is remarkable.

To ease the comparison of the algorithms having a similar performance, we plot in Fig. 3 these runtimes relative to the one of the classic (1,λ) EA, i. e., we divide the average runtime by the value of the classic (1,λ) EA. This shows that in most
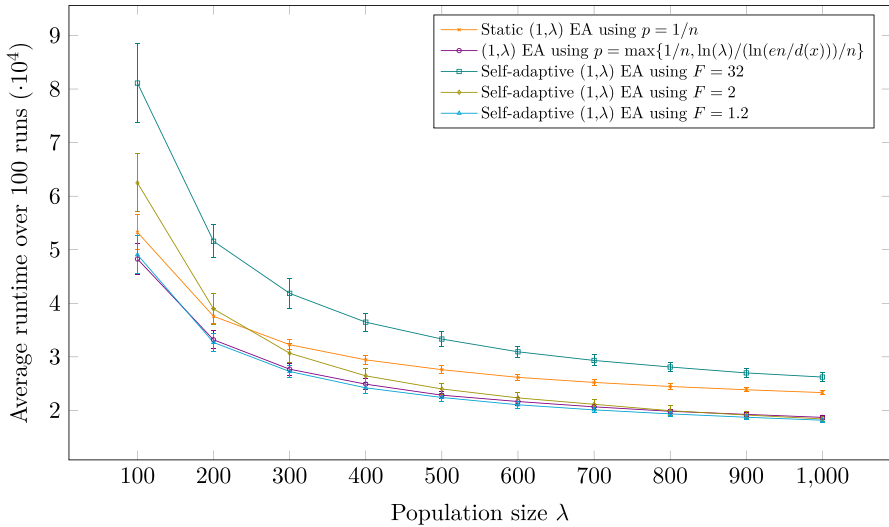
Fig. 2 Average runtime over 100 runs of five variants of the $(1,\lambda)$ EA on ONEMAX for $n = 10^5$
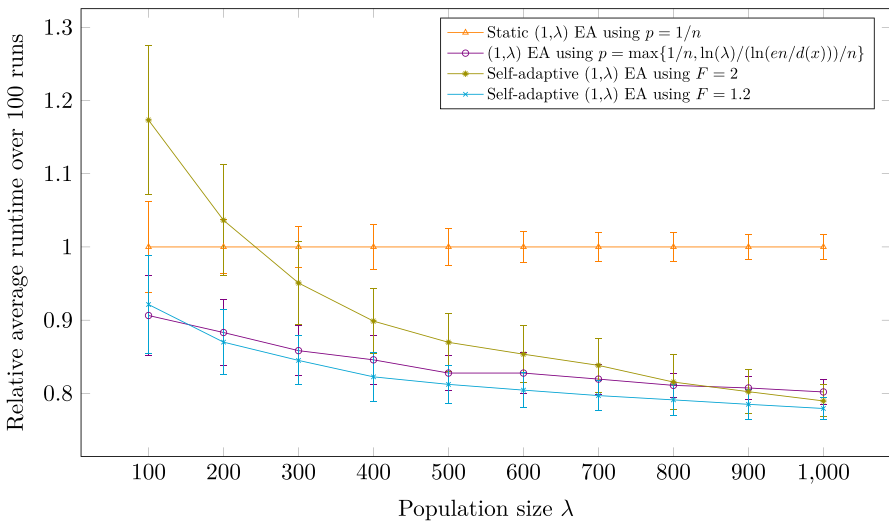

Fig. 3 Average runtime of three dynamic $(1,\lambda)$ EA s relative to the average runtime of the static $(1,\lambda)$ EA on ONEMAX ($n = 10^5$)

cases, the EAs using a dynamic mutation rate outperform the classic $(1,\lambda)$ EA. We also notice that the self-adaptive EA appears to outperform the one using the fitness-dependent rate for sufficiently large values of λ, e.g., for $\lambda \geq 200$ when $F = 1.2$.

To understand how our tie-breaking rule influences the performance, we also ran the self-adapting $(1,\lambda)$ EA without the bias towards smaller rates when breaking ties. In Fig. 4, we again plot the average runtimes over 100 runs relative to the results
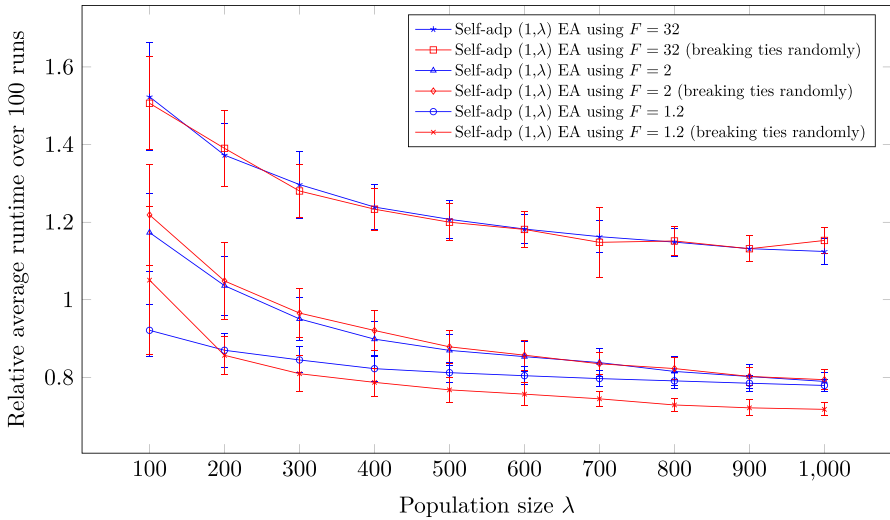
Fig. 4 Relative average runtime of self-adapting $(1,\lambda)$ EA s with different tie breaking rules on ONEMAX ($n = 10^5$)

of static $(1,\lambda)$ EA. We use the three update factors 1.2, 2, and 32 and the two tie-breaking rule of preferring the smaller rate in case of ties (as in our theoretical analysis) and random tie-breaking, that is, choosing uniformly at random an offspring with maximal fitness and taking its rate as the new rate of the algorithm. While for the two larger factors $F = 2$ and $F = 32$ no significant differences are visible, we see that for $F = 1.2$ random tie-breaking surpasses biased tie-breaking significantly when λ becomes larger than 200.

To understand how the tie-breaking rule influences the mutation strength chosen by the algorithm, we plot in Fig. 5 the mutation strength used at each fitness distance with a setting of $n = 10000$, $\lambda = 500$, and $F = 1.2$. We regarded one exemplary runs of our algorithm with each tie breaking rule. In each of these two experiments, we determined the set of all pairs (d_t, r_t) such that in iteration t, the fitness distance of the parent individual was d_t and its rate was r_t. We then plotted these sets, where to increase the readability we connected the points to polygonal curves. This visualization clearly shows that random tie breaking lets the algorithm pick larger rates more frequently. Together with the better runtimes, it appears that biased tie-breaking has a small negative effect on the choice of the mutation strength.

Finally, we regard the question of how to set the initial rate r^{init}. From the general experience that larger mutation rates are more profitable at the start of the search process, one could guess that it is a good idea to start with the largest possible rate $r_{\max} = F^{\lfloor \log_F(n/(2F)) \rfloor}$ instead of the smallest possible rate $r_{\min} = F$. For the settings used in Fig. 5, that is, $n = 10{,}000$, $\lambda = 500$, and $F = 1.2$, we obtain (as average of 100 runs) the runtimes given in Table 1. So indeed an initialization with a larger rate gives some improvement. Since it might be a particularity of the ONEMAX test function that huge rates are initially beneficial, we would not give out a general recommendation to start with the rate $r_{\max}$, but only state that we observed moderate
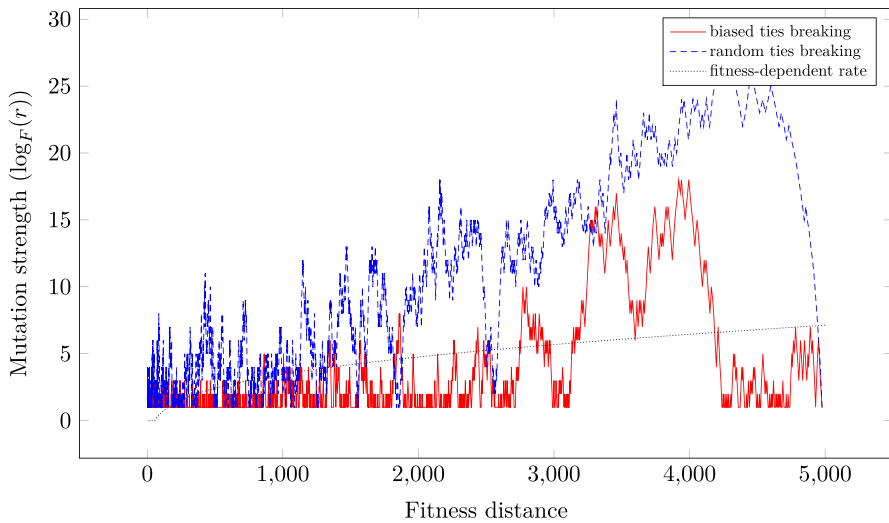
Fig. 5 Mutation strengths used at a certain fitness distance level in two example runs of the self-adapting $(1,\lambda)$ EA on ONEMAX ($n = 10000$, $\lambda = 500$, $F = 1.2$). For comparison, also the fitness-dependent rate proposed in [9] is plotted. Recall that a mutation strength of r in the self-adapting runs means that in average half the offspring use the rate r/F and half use the rate rF

Table 1 Comparison of the average runtime of 100 runs for different initial mutation rates ($n = 10000$, $\lambda = 500$, and $F = 1.2$)	Average runtime	Biased ties breaking	Random ties breaking
	$r^{\mathrm{init}} = r_{\min}$	2137	2011
	$r^{\mathrm{init}} = r_{\max}$	2080	1974

performance differences from using different initial rates, making the initial rate not the most critical parameter of the algorithm, but still one that can be worth optimizing.

6 Conclusions

In this work, we have designed and analyzed a self-adaptive $(1,\lambda)$ EA using a simple scheme for mutating the mutation rate. We have proven that for $\lambda = \Omega(\log n)$ it achieves an expected runtime (number of fitness evaluations) of $O(n\lambda/\log\lambda + n\log n)$ on ONEMAX, which is the best possible asymptotic runtime for λ-parallel mutation-based unbiased black-box algorithms. Hence, we have identified a simple and natural example where self-adaptation of strategy parameters in discrete EAs can lead to provably optimal runtimes that beat all known static parameter settings. Moreover, a relatively complicated and partly unintuitive self-adjusting scheme for the mutation rate proposed in [23] can be replaced by our simple

endogenous scheme. Experimental results confirm that our scheme achieves runtimes that are comparable with a $(1,\lambda)$ EA using a fitness-dependent and asymptotically optimal mutation rate.

The analysis of this $(1,\lambda)$ EA has revealed a non-trivial stochastic process in the cross product of fitness distances and mutation rates. We have advanced the techniques for the analysis of such two-dimensional processes, both via two new lemmas on occupation probabilities and by proposing suitable potential functions allowing to use classic drift theorems.

Altogether, we are optimistic that our research helps pave the ground for further uses and analyses of self-adaptive EAs.

References

1. Akimoto, Y., Auger, A., Glasmachers, T.: Drift theory in continuous search spaces: expected hitting time of the $(1 + 1)$-ES with 1/5 success rule. In: Proceedings of GECCO '18, pp. 801–808. ACM (2018)
2. Auger, A., Doerr, B. (eds.): Theory of Randomized Search Heuristics. World Scientific Publishing, Singapore (2011)
3. Antipov, D., Doerr, B., Fang, J., Hetet, T.: Runtime analysis for the $(\mu + \lambda)$ EA optimizing OneMax. In: Proceedings of GECCO '18, pp. 1459–1466. ACM (2018)
4. Antipov, D., Doerr, B., Yang, Q.: The efficiency threshold for the offspring population size of the (μ, λ) EA. In: Proceedings of GECCO '19, pp. 1461–1469. ACM (2019)
5. Bäck, T.: Self-adaptation in genetic algorithms. In: Proceedings of ECAL '92, pp. 263–271. MIT Press (1992)
6. Böttcher, S., Doerr, B., Neumann, F.: Optimal fixed and adaptive mutation rates for the LeadingOnes problem. In: Proceedings of PPSN '10, pp. 1–10. Springer (2010)
7. Bernstein, S.N.: On a modification of Chebyshev's inequality and of the error formula of Laplace. Ann. Sci. Inst. Sav. Ukraine, Sect. Math. 1 **4**, 38–49 (1924)
8. Birattari, M.: Tuning Metaheuristics: A Machine Learning Perspective. Springer, Berlin (2009)
9. Badkobeh, G., Lehre, P.K., Sudholt, D.: Unbiased black-box complexity of parallel search. In: Proceedings of PPSN '14, pp. 892–901. Springer (2014)
10. Corus, D., Dang, D.-C.D.-C., Eremeev, A.V., Lehre, P.K.: Level-based analysis of genetic algorithms and other search processes. IEEE Trans. Evol. Comput. **22**, 707–719 (2018)
11. Chen, T., He, J., Sun, G., Chen, G., Yao, X.: A new approach for analyzing average time complexity of population-based evolutionary algorithms on unimodal problems. IEEE Trans. Syst. Man Cybern. Part B **39**, 1092–1106 (2009)
12. Doerr, B., Doerr, C.: Optimal parameter choices through self-adjustment: applying the 1/5-th rule in discrete settings. In: Proceedings of GECCO '15, pp. 1335–1342. ACM (2015)
13. Doerr, B., Doerr, C.: Optimal static and self-adjusting parameter choices for the $(1 + (\lambda, \lambda))$ genetic algorithm. Algorithmica **80**, 1658–1709 (2018)
14. Doerr, B., Doerr, C.: Theory of parameter control for discrete black-box optimization: provable performance gains through dynamic parameter choices. In: Doerr, B., Neumann, F. (eds.) Theory of Evolutionary Computation: Recent Developments in Discrete Optimization, pp. 271–321. Springer, Berlin (2020)
15. Doerr, B., Doerr, C., Ebel, F.: Lessons from the black-box: fast crossover-based genetic algorithms. In: Proceedings of GECCO '13, pp. 781–788. ACM (2013)

16. Doerr, B., Doerr, C., Kötzing, T.: Provably optimal self-adjusting step sizes for multi-valued decision variables. In: Proceedings of PPSN '16, pp. 782–791. Springer (2016)
17. Doerr, B., Doerr, C., Kötzing, T.: Static and self-adjusting mutation strengths for multi-valued decision variables. Algorithmica **80**, 1732–1768 (2018)
18. Doerr, B., Doerr, C., Yang, J.: k-bit mutation with self-adjusting k outperforms standard bit mutation. In: Proceedings of PPSN '16, pp. 824–834. Springer (2016)
19. Doerr, B., Doerr, C., Yang, J.: Optimal parameter choices via precise black-box analysis. Theor. Comput. Sci. **801**, 1–34 (2020)
20. Doerr, B., Fouz, M., Witt, C.: Quasirandom evolutionary algorithms. In: Proceedings of GECCO '10, pp. 1457–1464. ACM (2010)
21. Doerr, B., Fouz, M., Witt, C.: Sharp bounds by probability-generating functions and variable drift. In: Proceedings of GECCO '11, pp. 2083–2090. ACM (2011)
22. Doerr, B., Goldberg, L.A.: Adaptive drift analysis. Algorithmica **65**, 224–250 (2013)
23. Doerr, B., Gießen, C., Witt, C., Yang, J.: The $(1 + \lambda)$ evolutionary algorithm with self-adjusting mutation rate. Algorithmica **81**, 593–631 (2019)
24. Doerr, B., Jansen, T., Sudholt, D., Winzen, C., Zarges, C.: Mutation rate matters even when optimizing monotone functions. Evol. Comput. **21**, 1–21 (2013)
25. Droste, S., Jansen, T., Wegener, I.: On the analysis of the (1+1) evolutionary algorithm. Theor. Comput. Sci. **276**, 51–81 (2002)
26. Doerr, B., Johannsen, D., Winzen, C.: Multiplicative drift analysis. Algorithmica **64**, 673–697 (2012)
27. Doerr, B., Künnemann, M.: Optimizing linear functions with the $(1+\lambda)$ evolutionary algorithm—different asymptotic runtimes for different instances. Theor. Comput. Sci. **561**, 3–23 (2015)
28. Dang, D.-C., Lehre, P.K.: Runtime analysis of non-elitist populations: From classical optimisation to partial information. Algorithmica **75**, 428–461 (2016)
29. Dang, D.-C., Lehre, P.K.: Self-adaptation of mutation rates in non-elitist populations. In: Proceedings of PPSN '16, pp. 803–813. Springer (2016)
30. Doerr, B., Lissovoi, A., Oliveto, P.S., Warwicker, J.A.: On the runtime analysis of selection hyper-heuristics with adaptive learning periods. In: Proceedings of GECCO '18, pp. 1015–1022. ACM (2018)
31. Doerr, B., Neumann, F.: Theory of Evolutionary Computation—Recent Developments in Discrete Optimization. Springer, Berlin (2019)
32. Doerr, B.: An elementary analysis of the probability that a binomial random variable exceeds its expectation. Stat. Probab. Lett. **139**, 67–74 (2018)
33. Doerr, B.: Analyzing randomized search heuristics via stochastic domination. Theor. Comput. Sci. **773**, 115–137 (2019)
34. Doerr, B.: Probabilistic tools for the analysis of randomized optimization heuristics. In: Doerr, B., Neumann, F. (eds.) Theory of Evolutionary Computation: Recent Developments in Discrete Optimization, pp. 1–87. Springer, Berlin (2020). https://arxiv.org/abs/1801.06733
35. Doerr, B., Witt, C., Yang, J.: Runtime analysis for self-adaptive mutation rates. In: Proceedings of GECCO '18, pp. 1475–1482. ACM (2018)
36. Gießen, C., Witt, C.: The interplay of population size and mutation probability in the $(1 + \lambda)$ EA on OneMax. Algorithmica **78**, 587–609 (2017)
37. Hajek, B.: Hitting-time and occupation-time bounds implied by drift analysis with applications. Adv. Appl. Probab. **13**, 502–525 (1982)
38. Hwang, H.-K., Panholzer, A., Rolin, N., Tsai, T.-H., Chen, W.-M.: Probabilistic analysis of the (1+1)-evolutionary algorithm. Evol. Comput. **26**, 299–345 (2018)
39. Hwang, H.-K., Witt, C.: Sharp bounds on the runtime of the (1+1) EA via drift analysis and analytic combinatorial tools. In: Proceedings of FOGA '19, pp. 1–12. ACM (2019)
40. Jägersküpper, J.: Combining Markov-chain analysis and drift analysis - the (1+1) evolutionary algorithm on linear functions reloaded. Algorithmica **59**, 409–424 (2011)
41. Jansen, T.: Analyzing Evolutionary Algorithms—The Computer Science Perspective. Springer, Berlin (2013)
42. Jansen, T., De Jong, K.A., Wegener, I.: On the choice of the offspring population size in evolutionary algorithms. Evol. Comput. **13**, 413–440 (2005)
43. Johannsen, D.: Random combinatorial structures and randomized search heuristics. PhD thesis, Saarland University (2010)
44. Jägersküpper, J., Storch, T.: When the plus strategy outperforms the comma strategy—and when not. In: Proceedings of FOCI '07, pp. 25–32. IEEE (2007)

45. Jansen, T., Wegener, I.: On the analysis of a dynamic evolutionary algorithm. J. Discret. Algorithms **4**, 181–199 (2006)
46. Karafotias, G., Hoogendoorn, M., Eiben, A.E.: Parameter control in evolutionary algorithms: trends and challenges. IEEE Trans. Evol. Comput. **19**, 167–187 (2015)
47. Kötzing, T., Lissovoi, A., Witt, C.: (1+1) EA on generalized dynamic OneMax. In: Proceedings of FOGA '15, pp. 40–51. ACM (2015)
48. Lehre, P.K.: Negative drift in populations. In: Proceedings of PPSN '10, pp. 244–253. Springer (2010)
49. Lässig, J., Sudholt, D.: Adaptive population models for offspring populations and parallel evolutionary algorithms. In: Proceedings of FOGA '11, pp. 181–192. ACM (2011)
50. Lehre, P.K., Witt, C.: Concentrated hitting times of randomized search heuristics with variable drift. In: Proceedings of ISAAC '14, pp. 686–697. Springer (2014)
51. Lehre, P.K., Yao, X.: On the impact of mutation–selection balance on the runtime of evolutionary algorithms. IEEE Trans. Evol. Comput. **16**, 225–241 (2012)
52. Mitavskiy, B., Rowe, J.E., Cannings, C.: Theoretical analysis of local search strategies to optimize network communication subject to preserving the total number of links. Int. J. Intell. Comput. Cybern. **2**, 243–284 (2009)
53. Mühlenbein, H.: How genetic algorithms really work: mutation and hillclimbing. In: Proceedings of PPSN '92, pp. 15–26. Elsevier (1992)
54. Neumann, F., Witt, C.: Bioinspired Computation in Combinatorial Optimization—Algorithms and Their Computational Complexity. Springer, Berlin (2010)
55. Robbins, H.: A remark on Stirling's formula. Am. Math. Mon. **62**, 26–29 (1955)
56. Rowe, J.E.: Linear multi-objective drift analysis. Theor. Comput. Sci. **736**, 25–40 (2018)
57. Rowe, J.E., Sudholt, D.: The choice of the offspring population size in the $(1, \lambda)$ evolutionary algorithm. Theor. Comput. Sci. **545**, 20–38 (2014)
58. Smit, S.K., Eiben, A.E.: Beating the 'world champion' evolutionary algorithm via REVAC tuning. In: Proceedings of CEC '10, pp. 1–8. IEEE Press (2010)
59. Sudholt, D.: A new method for lower bounds on the running time of evolutionary algorithms. IEEE Trans. Evol. Comput. **17**, 418–435 (2013)
60. Wegener, I.: Simulated annealing beats Metropolis in combinatorial optimization. In: Proceedings of ICALP '05, pp. 589–601. Springer (2005)
61. Witt, C.: Runtime analysis of the $(\mu + 1)$ EA on simple pseudo-Boolean functions. Evol. Comput. **14**, 65–86 (2006)
62. Witt, C.: Tight bounds on the optimization time of a randomized search heuristic on linear functions. Comb. Probab. Comput. **22**, 294–318 (2013)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.