The linear hidden subset problem for the (1+1) EA with scheduled and adaptive mutation rates

Extended Abstract

Hafsteinn Einarsson
Department of Computer Science, ETH Zürich

Johannes Lengler
Department of Computer Science, ETH Zürich

Marcelo Matheus Gauy
Department of Computer Science, ETH Zürich

Florian Meier
Department of Computer Science, ETH Zürich

Asier Mujika
Department of Computer Science, ETH Zürich

Angelika Steger
Department of Computer Science, ETH Zürich

Felix Weissenberger
Department of Computer Science, ETH Zürich

ABSTRACT

We study unbiased (1 + 1) evolutionary algorithms on linear functions with an unknown number $n$ of bits with non-zero weight. Static algorithms achieve an optimal runtime of $O(n(\ln n)^{2+\epsilon})$, however, it remained unclear whether more dynamic parameter policies could yield better runtime guarantees. We consider two setups: one where the mutation rate follows a fixed schedule, and one where it may be adapted depending on the history of the run. For the first setup, we give a schedule that achieves a runtime of $(1 \pm o(1))\beta n \ln n$, where $\beta \approx 3.552$, which is an asymptotic improvement over the runtime of the static setup. Moreover, we show that no schedule admits a better runtime guarantee and that the optimal schedule is essentially unique. For the second setup, we show that the runtime can be further improved to $(1 \pm o(1))\ln n$, which matches the performance of algorithms that know $n$.

Finally, we study the related model of initial segment uncertainty with static position-dependent mutation rates, and derive asymptotically optimal lower bounds. This answers a question by Doerr, Doerr, and Kötzing.

KEYWORDS

Evolutionary Algorithm, Mutation-Based, Linear Functions, Hidden Subset Problem, Unknown Problem Length, Adaptive Parameters, Parameter Control

ACM Reference Format:

Hafsteinn Einarsson, Johannes Lengler, Marcelo Matheus Gauy, Florian Meier, Asier Mujika, Angelika Steger, and Felix Weissenberger. 2018. The linear hidden subset problem for the (1+1) EA with scheduled and adaptive mutation rates: Extended Abstract. In GECCO '18: Genetic and Evolutionary Computation Conference, July 15–19, 2018, Kyoto, Japan. ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3205455.3205519

1 INTRODUCTION

In mutation-based evolutionary algorithms (EAs) the mutation rate is a critical parameter. For example, for linear pseudo-boolean fitness functions $f : \{0, 1\}^n \rightarrow \mathbb{R}$, Witt has shown in [13] that the optimal static mutation rate is $1/n$, which leads to a runtime (the number of function evaluations before a global optimum is found) of $(1 \pm o(1))\ln n$. Interestingly, for any other mutation rate $c/n$, where $c$ is a constant, Witt proved a strictly larger runtime of $(1 \pm o(1))\ln n$. This runtime is worse by roughly a factor of $1/c$ if $c < 1$, and it becomes exponentially worse as $c > 1$ grows. Thus, finding the optimal mutation rate may not only be difficult but also paramount.

Crucially, even for a simple function like OneMax, the optimal mutation rate $1/n$ can only be used if the problem size $n$ is known. However, consider the following hidden subset problem: the search space is $[0, 1]^N$, but only a small subset of $n < N$ positions is fitness-relevant. We call this hidden set the support of the fitness function, and we study fitness functions that depend linearly on the supporting bits. In this case, since $n$ is unknown, the optimal mutation rate is also unknown. This problem was proposed by Cathabard, Lehre, and Yao [1] and has been studied by Doerr, Doerr,
and Kötzing [3, 4] in the case of OneMax and LeadingOnes instead of linear functions.

Situations in which the fitness is a function of a small hidden subset of parameters occurs naturally in many practical applications, particularly in the context of big data. For example, complex models like a biospheric model or a neural network may come with an immense number of parameters, and the choice of parameters (which is feasible with sufficient data) often leads to high-dimensional optimization problems. However, it often turns out in hindsight that only a small subset of parameters are relevant, which is exactly the situation captured by the hidden subset problem.

In the aforementioned work [1, 3, 4], the problems were analyzed for a static choice of mutation rates (cf. below). However, when faced with unknown problem characteristics, it is natural to consider more dynamic parameter handling, either scheduled or adaptive ones. In [4] it was speculated that dynamic parameter handling could improve the runtime compared to the static setup. In this paper, we quantify the gain or loss of either method. We restrict ourselves to mutation-based $(1 + 1)$ EAs with standard bit mutation, and we distinguish three different types of parameter handling.

1. In the static setup, a probability distribution $D$ is fixed before the algorithm starts, and in each round the mutation rate is drawn from $D$.
2. In the scheduled setup, a sequence $D_t$ of probability distributions is fixed before the algorithm starts. Then, in the $t$-th round of the algorithm the mutation rate is drawn from $D_t$.
3. In the adaptive setup, the mutation rate at time $t$ may be chosen depending on the history of the run up to time $t - 1$.

### 1.1 Previous work and our contribution

As mentioned before, Witt has shown in [13] that for known parameters, the optimal runtime for any linear function is $1/n$, yielding a runtime of $(1 + o(1))en \ln n$. Strictly speaking, Witt only considered static mutation rates. However, his proof is based on a drift argument, and he shows that for a suitable potential function, the drift towards the optimum is strongest for mutation rate $1/n$. Thus, his proof also shows that no adaptive policy for the mutation rate can beat the runtime of $(1 + o(1))en \ln n$. Therefore, in our more difficult setting where $1/n$ is unknown, the bound $(1 + o(1))en \ln n$ is also a lower bound on the runtime with any parameter handling policy. The question is thus: how much do we lose compared to this lower bound, depending on the parameter handling.

**Static Mutation Rate.** The static setup has been studied (for OneMax and the non-linear LeadingOnes function) in [3, 4]. For OneMax, it turned out that even with the best static setup, the runtime is asymptotically slower if $n$ is unknown. More precisely, for any static setup the runtime is at least $Ω(n \ln^2 n)$ [4], and this bound is tight up to $\ln \ln n$ factors. Since OneMax is the easiest linear function by [12], the lower bound holds for every linear function.

**Scheduled Mutation Rate.** For the scheduled setup, we show that there is an asymptotic improvement of the runtime over the runtime in the static setup. Moreover, the runtime is only by the factor $β/e ≈ 1.307$ larger than in the case where $n$ is known. More precisely, we show that the scheduling policy $D_{opt}^n$ that sets the mutation rate in the $t$-th step deterministically to $a \ln(t)/t$ for $a \approx 1.545$ leads whp to a runtime of $(1 + o(1))en \ln n$ for every linear function with support of size $n$. This policy is optimal, that is, for every other schedule deterministic or randomized, there are infinitely many $n$ such that the runtime on every linear function with support of size $n$ is whp at least $(1 + o(1))β^n \ln \ln n$ for some $β > β$.

**Adaptive Mutation Rate.** Finally, we show that there is no significant price for the unknown $n$ if adaptive schedules are used: there is an adaptive scheduling scheme that achieves whp runtime $(1 + o(1))en \ln n$, thus matching the lower bound from the setting with known $n$.

There are two ways to interpret the results. Firstly, we may define the black-box complexity (BBC) of a function with respect to unbiased $(1 + 1)$ EAs with the respective updating scheme as the best runtime achievable by algorithms of this kind. In this sense, the result in [4] says that the BBC of linear functions for static mutation rate is $Ω(n \ln^2 n)$, while we show that the BBC for scheduled and adaptive mutation rates is $(1 + o(1))β^n \ln n$ and $(1 + o(1))en \ln n$, respectively.

Secondly, we may consider this result as an analogue to the price of anarchy [9, 11] in game theory. We may define the price of non-adaptiveness of a family $F = (f_n)_{n \geq 1}$ of functions, where $f_n$ has support of size $n$, to be

$$\text{PoNA}(F) := \lim_{n \to \infty} \frac{\text{runtime of best scheduled algorithm on } f_n}{\text{runtime of best adaptive algorithm on } f_n}.$$  

Then we show in this paper for every family $F = (f_n)$ of linear functions, we have $\text{PoNA}(F) = β/e ≈ 1.307$. Note that this definition also makes the somewhat ambiguous concept of “best” algorithm precise. For the adaptive case, there is a single algorithm which achieves, up to lower order terms, for all $n$ simultaneously the optimal runtime, so it is clear that this algorithm is best. For the scheduled setup, this is not the case, so we define the “best” algorithm as the algorithm which minimizes the PoNA.

**Initial Segment Uncertainty.** Doerr, Doerr, and Kötzing showed in [4] that for LeadingOnes there is an intimate connection between the (static) hidden-subset problem (HSP) considered in this paper, and the following problem with initial segment uncertainty (ISU). The support is an initial segment $\{0, \ldots, n\}$ of unknown length $n$, and for each bit $i$ the algorithm may choose a probability $p_i$. In each round the offspring is generated by flipping the $i$-th bit with probability $p_i$. This ISU variant was historically the first to be studied and was motivated in [1] by the study of finite...
We consider a large search space $\mathbb{N}$. In [4] it was conjectured that there is also a connection between the ISU model and the HSP for other problems than LeadingOnes, specifically for OneMax.

It was proved in [4] that for every monotonically decreasing, summable sequence $(d_i)_{i \geq 1}$ of positive reals there is an algorithm in the ISU model with runtime $O(\ln(n)/d_n)$ on OneMax. As an open problem, the authors asked for matching lower bounds. In this paper we provide such bounds, in the following sense. For every non-summable monotonically decreasing sequence $(d_i)_{i \geq 1}$ of positive reals there is a constant $c > 0$ such that every algorithm in the ISU model has runtime at least $c \ln(n)/d_n$ on OneMax. Interestingly, both the upper and lower bound in the ISU model match the upper and lower bound in the HSP, which were derived in [4]. Although this result is less tight than the connection for LeadingOnes (where the distributions of runtimes exactly coincide with each other), this gives further indication for a fundamental connection between the ISU model and the HSP.

2 NOTATION, TOOLS, ALGORITHMS

2.1 Models of uncertainty

We consider a large search space $[0,1]^N$. In contrast, the function $f$ to be optimized only depends on a small subspace. More precisely, there is a set $I \subseteq \{1, \ldots, N\}$ and a function $f$ on $[0,1]^I$, where $n := |I|$, such that $f(x) = f_I(x_I)$ for all $x \in [0,1]^N$. Here, $x_I$ denotes the bit string consisting of the bits $x_i$ of $x$ with $i \in I$. The dimension $N$ of the search space does not affect the results in this paper. Therefore, we assume that the search space is $[0,1]^N$. We call the positions $I \subseteq \mathbb{N}$ that $f$ depends on the relevant bits (or support) of $f$. To ease notation, we also use the symbol $f$ for $f_I$.

We consider two models of uncertainty. In the unrestricted uncertainty model, the relevant bits $I$ and the number of relevant bits $n$ are unknown. In the initial segment problem, $I$ is the initial segment $[n] := \{1, \ldots, n\}$, and the number of relevant bits $n$ is unknown.

2.2 Algorithmic setup

The $(1+1)$ EA has the goal of finding a search point that minimizes a function $f$. First, it draws u.a.r. a search point $x \in [0,1]^I$. Then, an offspring $y$ of the current search point $x$ is created in every round by flipping each bit independently with probability $p$. The parameter $p$ is called mutation strength, mutation rate or mutation parameter. In this paper we stick with mutation rate.

For compact descriptions of Algorithms 1-4, we define the operator $\text{Mutate}(x,p)$, which generates a mutation $y$ of $x$ by flipping each bit independently with probability $p$. If $p$ is a sequence, then each bit $x_i$ is flipped with probability $p_i$. In the static setup (see Algorithm 1), for each time $t$ the mutation rate $p_t$ is drawn from a fixed probability distribution $D$ over the interval $[0,1]$, which is identical for all $t$. In the scheduled setup, a sequence of such distributions $D_t$ is fixed in advance, and the mutation rates $p_t$ at time $t$ is drawn from $D_t$, see Algorithm 2. In the adaptive setup, the distributions $D_t$ can depend on the history of the process. However, we assume that the algorithm is comparison-based, i.e., whenever the fitness values of the search point $x$ and offspring $y$ are compared, the algorithm receives from an oracle the information whether the offspring is accepted or not. Then, it may choose $D_t$, depending on all bits received from the oracle before time $t$. We also note that the definition implies all versions of the $(1+1)$ EA are unbiased, i.e., the mutation operator is invariant under the automorphisms of the search space. For more background on comparison-based and unbiased algorithms, see [6].

Finally, for the ISU model, we consider position dependent mutation rates $\tilde{p}_i$, where an offspring $y$ of $x$ is created by flipping the bit at position $i$ with probability $p_i$, see Algorithm 3. The $p_i$ are fixed over time.

As in previous work, we consider the number of fitness evaluations as the complexity measure. We define the runtime (or optimization time) as the number of $f$-evaluations until the search point with minimal $f$-value is reached.

2.3 Basic Notation

We denote sequences $(p_t)_{t \in \mathbb{N}}$ by $\tilde{p}$. In this paper, we consider the OneMax function and the class of linear functions $f$ to be minimized. The OneMax function with support $I$ is defined by $f(x) = \sum_{i \in I} x_i$ for any $x \in [0,1]^I$. A linear function $f$ with support $I$ depends linearly on the bits in $I$, that is, $f(x) = \sum_{i \in I} w_i x_i$ for some $w_i \in \mathbb{R}$. Since $f(x)$ can be written as $\sum_{i \in I, w_i > 0} w_i x_i + \sum_{i \in I, w_i < 0} |w_i|(1-x_i) - \sum_{i \in I, w_i < 0} |w_i|$, without loss of generality we can assume that $w_i \geq 0$ for all $i \in I$. Therefore, our target search point is the all 0 string from now on.

Further, we denote by $x^1$ the search point at time $t$. We say bit $i$ flips at time $t$ if $y_i$ is set to 1 at $x_i^1$ in the mutation step of the algorithm. We say that there is a single bit flip in round $t$ if exactly one relevant bit flips, and there is a multi bit flip if at least two relevant bits flip. Further, we say bit $i$ changes at time $t$ if $x_i^t \neq x_i^{t-1}$, which happens if bit $i$ flips at time $t$ and the offspring $y$ is accepted. We say that bit $i$ is optimized at time $t$ if $x_i^t = 0$.

Algorithm 1: The static $(1+1)$ EA with mutation rate distribution $D$ minimizing a pseudo-Boolean function $f$: $[0,1]^N \to \mathbb{R}$

1. Initialization: Sample $x \in [0,1]^N$ uniformly at random;
2. Optimization: for $t = 1, 2, 3, \ldots$
   3. $p_t \sim D$;
   4. $y \leftarrow \text{Mutate}(x,p_t)$;
   5. if $f(y) \leq f(x)$ then $x \leftarrow y$; //selection step

2.4 ONEMAX is the easiest linear function

Let $A$ and $B$ be two random variables that take values in $\mathbb{N}$. A stochastically dominates $B$ if $\Pr(A \geq i) \geq \Pr(B \geq i)$ for all $i \in \mathbb{N}$.
The following theorem was proven in [13] slightly more general, that is, for mutation based EAs with arbitrary population size.

**Theorem 2.1.** (Theorem 6.2 in [13]) Consider the static (1 + 1) EA $A$ with mutation rate $p \leq 1/2$. Then, the optimization time of $A$ on any function with a unique global optimum stochastically dominates the optimization time of $A$ on ONE-MAX.

In [13], Witt proved this theorem by induction over time $t$. The proof requires that $p \leq 1/2$ for every time step $t$, but does not require that $p$ is fixed. Thus, the theorem can be extended to the setup where the mutation rates $p$ are scheduled or adaptively chosen. Therefore, Witt’s proof also applies the following theorem.

**Theorem 2.2.** (Adaptation of Theorem 6.2 in [13]) Let $A$ be a (1+1) EA with scheduled or adaptively chosen mutation rates $p$ satisfying $p_t \leq 1/2$. Then, the optimization time of $A$ on any function with a unique global optimum stochastically dominates the optimization time of Algorithm $A$ on ONE-MAX.

### 3 SCHEDULED SETUP

First we give some intuition on how the mutation rates $\tilde{p}$ should be chosen. It turns out that nearly all time of the optimization process is spent to optimize the last $n^\epsilon$ non optimized bits (for some $\epsilon > 0$). In the regime where only very few 1-bits are left the probability that the number of 1-bits decreases given a multi bit flip is much smaller than the same probability given a single bit flip. If there were only single bit flips, then by a coupon collector type argument $(1 + o(1))/(n \ln n)$ of them are needed to optimize a function with $n$ relevant bits. If there are $n$ relevant bits, the probability of a single bit flip is maximized by $p = 1/n$. Since $n$ is unknown, we need to solve the problem for all $n$ simultaneously. If we fix $p_t$, then round $t$ contributes substantially to optimizing functions $f$ that have support size $n = \Theta(p_t^{-1})$, because for these $n$ the probability of a single bit flip is $np_t(1 - p_t)^{n-1} = \Theta(1)$. We wish to optimize functions $f$ with support size $n$ in time $T_n = \Theta(n \ln n)$. Since there is more time to optimize functions with large support $n$, for small $t$ the $p_t$ should contribute to solving functions with small support. More precisely, for any $n$ a significant number of $p_t$’s with $t \leq T_n$ needs to be chosen of order $\Theta(1/n)$. This suggests to choose $p_t = \Theta(\ln(t)/t)$. As we will see, the optimal choice for the hidden factor will be the following constant $\alpha$.

**Definition 3.1 ($\alpha, \beta$).** Let $\alpha$ be the unique solution of the equation

$$
\int_0^1 \frac{1}{u} \alpha^{-1/2} \, du = 1,
$$

and let $\beta = \frac{\alpha}{\ln \alpha}$. The numerical values are approximately $\alpha \approx 1.54468$ and $\beta \approx 3.55248$.

**Remark 1.** Since the left hand side of Equation 1 is monotone in $\alpha$ (mind $u < 1$) it is easy to see that there is a unique solution to Equation 1. Further, the variable transformation $z = \alpha \ln \alpha u$ transforms the integral into $\int_0^1 \frac{\beta}{z} e^{-\frac{z}{2}} \, dz = 1$. Define $h(\alpha, \beta) = \int_0^1 \frac{\beta}{z} e^{-\frac{z}{2}} \, dz$. Fix $\alpha$ as in Definition 3.1, then $\alpha$ such that $\alpha = \alpha \ln \alpha$ maximizes $h(\alpha, \beta)$ (this can be seen by solving $\frac{\partial h(\alpha, \beta)}{\partial \alpha} = 0$). By continuity arguments, for $\beta' < \beta$ there is a $\delta > 0$ such that $h(\alpha', \beta') \leq (1 - \delta)$ for any $\alpha'$. Further, for $\alpha' \neq \alpha$ there is a $\alpha > \beta$ and $\delta > 0$ such that $h(\alpha', \beta') \leq (1 - \delta)$.

Now we are ready to state matching upper and lower bounds on the optimization time of the (1+1) EA with scheduled mutation rates on linear functions (see Algorithm 2).

**Theorem 3.2 (Lower bound).** Let $(D_t)_{t \in \mathbb{N}}$ be any scheduling policy and let $\beta$ be as in Definition 3.1. For infinitely many $n$, the optimization time of the (1+1) EA with scheduling policy $(D_t)_{t \in \mathbb{N}}$ on any linear function with $n$ relevant bits is whp at least $(1 - o(1))\beta n \ln n$.

It turns out that there is an optimal deterministic scheduling policy. Define $D^{\text{opt}}_n$ to be the distribution that sets $p_t = \alpha \ln(t)/t$ with probability 1, where $\alpha$ is defined in Definition 3.1.

**Theorem 3.3 (Upper Bound).** Let $\beta$ be defined as in Definition 3.1. Then the optimization time of the (1+1) EA with scheduling policy $D^{\text{opt}}_n$ is whp at most $(1 + o(1))\beta n \ln n$ for any linear function $f$ with $n$ relevant bits.

As mentioned in the introduction the lower bound can be strengthened in the sense that Theorem 3.2 holds for a subset of $n$ with positive density, and that the scheduling policy $D^{\text{opt}}_n$ is essentially unique. In order to make this precise, define the following measure $\mu$ on $\mathbb{N}$. For any $N \in \mathbb{N}$ define $\mu(N) = \sum_{t \in N} \ln(t)/t$. The density of a set $N$ is defined as $\liminf_{n \to \infty} \mu(N \cap [n])/\mu([n])$. For example, if a set contains for all $n$ at least $\alpha n$ elements of $[n, 2n]$ for some $\alpha > 0$, then it has positive density with respect to $\mu$. The proof of Theorem 3.2 also shows the following two remarks.

**Remark 2.** Theorem 3.2 holds for a subset $N \subset \mathbb{N}$ with positive density with respect to $\mu$.

**Remark 3 (Uniqueness of $D^{\text{opt}}_n$).** Assume that a policy $(D_t)_{t \in \mathbb{N}}$ deviates from $D^{\text{opt}}_n$ on a set $N$ with positive density with respect to $\mu$, that is, for all $t \in N$ either $p_t \leq (1 - \epsilon)\alpha \ln(t)/t$ or $p_t \geq (1 + \epsilon)\alpha \ln(t)/t$ holds. Then, there is a $\beta' > \beta$ such that for infinitely many $n$, the optimization time of the (1+1) EA with scheduling policy $(D_t)_{t \in \mathbb{N}}$ on any linear function with $n$ relevant bits is whp at least $(1 - o(1))\beta' n \ln n$. 

1494
3.1 Proof of Lower Bound

As discussed in Section 2.4, the optimization time of Algorithm 2 on any function with unique global optimum stochastically dominates the optimization time of Algorithm 2 on OneMax, for any sequence $\beta$ of mutation rates. Therefore, in order to prove Theorem 3.2 it suffices to show the following lemma.

**Lemma 3.4.** Let $(D_t)_{t \in \mathbb{N}}$ be any scheduling policy and let $\beta$ be as in Definition 3.1. For infinitely many $n$, the optimization time of the $(1+1)$ EA with scheduling policy $(D_t)_{t \in \mathbb{N}}$ on the OneMax function with $n$ relevant bits is whp at least $(1-o(1))\beta' n \ln n$.

Due to the symmetry of the OneMax function, the relevant bits can be permuted arbitrarily. Therefore, we can assume from now on that the offspring $y$ of $x$ is only accepted if it has better fitness $f(y) < f(x)$. For the remainder of this section we let $T_n' = \beta' n \ln n$ for some arbitrary $\beta' < \beta$.

Before we prove Lemma 3.4, we first need some preparations. The following lemma follows easily from concentration inequalities. We omit the proof due to space limitation.

**Lemma 3.5.** It holds whp that the OneMax function with $n$ relevant bits is not optimized at time $T_n'$ or there was a point in time with $cn/\ln^2 n$ relevant non-optimized bits for some $1 \leq c \leq 2$.

To show that a statement holds with high probability like in Lemma 3.4, we may assume that other events of high probability do occur. In particular, by Lemma 3.5 we may assume from now on that the process starts with $\ell_0 = cn/\ln^2 n$ relevant non-optimized bits for some $1 \leq c \leq 2$. It will turn out in this situation that it is rather unlikely that the fitness improves by multi-bit flips. The number of single-bit flips is bounded by the following lemma.

**Lemma 3.6.** Given $n$ relevant bits, denote by $Z_n$ the number of single bit flips until time $T_n'$. There exists a $\delta > 0$ such that for infinitely many $n$ it holds $\mathbb{E}[Z_n] \leq (1-\delta/3) n \ln n$. For each of these $n$ it holds with probability $1-o(n^{-3})$ that $Z_n \leq (1-\delta/6) n \ln n$.

**Proof Sketch.** We would like to bound the number of single-bit flips $Z_n$ by $(1-\delta/3) n \ln n$ for certain $n$. This bound is clearly not true for all $n$, for example, if $p_t = 1/m$ for all $t$, then $\mathbb{E}[Z_m] = \beta'/e \cdot m \ln m$ is larger than $m \ln m$ for $\beta' > e$. Therefore, we consider instead a weighted average $B = \sum_n \rho(n) Z_n$ over many $n$. The choice of $\rho$ is delicate, but it turns out that $\rho(n) = 1/n^2$ is the right choice.\(^{15}\)

In the technical part of the proof, we derive an upper bound on $B = \sum_n Z_n/n^2$. Note that $Z_n$ counts the single bit flips until time $T_n'$. To bound $B$, we first change the order of summation and then approximate the contribution of each $p_t$ to $B$ by an integral. Using variable transformations, it turns out that the contribution of $p_t$ can be bounded by $\int_0^\delta \frac{\beta'}{x} e^{-\alpha/x} dx$, which is smaller than $1-\delta$ if $\beta' < \beta$. Finally, the upper bound on $B$ implies that $\mathbb{E}[Z_n] \leq (1-\delta/3) n \ln n$ for infinitely many $n$. In essence, our calculations go along the following lines, but we omit the details.

\(^{15}\)For example, any function $\rho(n) = \Theta(1/n^3)$ would give a non-tight result if the $\Theta$ hides a function that oscillates by at least a factor $1+\epsilon$ for a constant $\epsilon > 0$. The optimal scaling $\rho(n) = 1/n^2$ can be found by variational methods, but once it is known (or guessed), the derivation of $\rho$ is no longer required for a proof.

Denote by $Y_t$ the indicator random variable that indicates whether at time $t$ exactly one out of the $n$ bits flips. Then

$$\mathbb{E}[Z_n] \approx \sum_{n=N}^M \sum_{t=10^\beta}^{10^n} \mathbb{E}[Y_t] = \sum_{n=N}^M \sum_{t=10^\beta}^{10^n} \frac{p_t e^{-p_t n}}{n} \leq \sum_{t=10^\beta}^{10^n} \sum_{n=N}^M \frac{p_t e^{-p_t n}}{n}.$$

Then, we approximate the inner sum by an integral. For ease of notation we define $a_t' = a_t' n \ln(n/t)$.

$$\int_{N}^{M} \frac{p_t e^{-p_t n}}{n} dn = \int_{N}^{M} a_t' \ln(n/t) n e^{-a_t' \ln(n/t)} n / t \ dn,$$

Using the variable transformations $x = t/(n \ln t)$ yields

$$\int_{N}^{M} \frac{\alpha_t' \ln^2 t}{t^2} e^{-a_t' \ln(n/t)} dx \leq \frac{\ln t}{t^2} \int_{0}^{\beta'} \frac{\alpha_t' \ln^2 x}{x^2} - \frac{a_t'}{x} dx \leq \frac{\ln t}{t^2} (1-\delta),$$

where the last inequality follows by Remark 1 since $\beta' < \beta$. Note that if $a_t'$ deviates from $\alpha$ for a range of positive density, then the last inequality also follows for some $\beta' > \beta$, which yields Remark 3. This implies $\mathbb{E}[Z_n] \leq (1-\delta/3) N \ln n$ for infinitely many $n$, because otherwise $\mathbb{E}[Z_n] > (1-\delta/3) N \ln n$. For these $n$, standard concentration inequalities imply that $Z_n \leq (1-\delta/6) n \ln n$ holds with probability $1-o(n^{-3})$.

□

Now we can show Lemma 3.4.

**Proof of Lemma 3.4.** Let $n$ be such that the statement of Lemma 3.6 holds, that is, the event $A$ that $Z_n \leq (1-\delta/6) n \ln n$ holds with probability $1-o(n^{-3})$, and recall that we assume to start with $\ell_0 = cn/\ln^2 n$ relevant non-optimized bits, where $c$ is some constant between $1 \leq c \leq 2$.

Denote by $W_i$ the Bernoulli random variable that is $1$ if the $i$-th of the initial $\ell_0$ relevant 1-bits is at time $T_n'$ and let $W = \sum_{i=1}^{\ell_0} W_i$ be the number of such bits. Furthermore, denote by $V_i$ the Bernoulli random variable that is equal to $W_i$ if $Z_n > (1-\delta/6) n \ln n$. If $Z_n \leq (1-\delta/6) n \ln n$, then assume that after time $T_n'$ the optimization process continues with additional $(1-\delta/6) n \ln n$ random single bit flips (i.e., in every round the offspring $y$ is produced by flipping one random bit of $x$, and $y$ is accepted if $f(y) > f(x)$). In this case we set $V_i$ to be $1$ if the $i$-th bit is $1$ after these additional random single bit flips, and let $V_i$ be $0$ otherwise. The advantage of the variables $V_i$ is that conditioned on $A$ there exist exactly $(1-\delta/6) n \ln n$ single bit flips, which make calculations simpler than with the variables $W_i$. Denote $V = \sum_{i=1}^{\ell_0} V_i$. Clearly, it holds $V_i \leq W_i$, and thus $V \leq W$. Therefore, it is enough to show that $\Pr(\text{whp} V > 0)$ in order to imply $W > 0$ whp, which proves Lemma 3.4. We will show this with the second moment method. We claim that $\text{Var}(V) = O(\mathbb{E}[V]^2/\ln n)$. Then, Chebyshev’s inequality implies that

$$\Pr(V = 0) \leq \Pr \left( |V - \mathbb{E}[V]| \geq \mathbb{E}[V] \right) \leq \frac{\mathbb{E}[V]^2}{\mathbb{E}[V]^2} = O \left( \frac{1}{\ln n} \right).$$

We define some additional notation in order to bound $\mathbb{E}[V]$ and $\text{Var}(V)$. Let $i \neq j$ be two arbitrary relevant bits. Let $B_i$ be the random

1495
variable that denotes the number of flipped relevant bits at time $t$, and let $C_t$ be the random variable that denotes the number of relevant 1-bits at time $t$. Define

$q_{k,\ell} := \Pr(x_{t}^1 = 0 \mid x_{t-1}^\ell = 1, B_t = k, C_{t-1} = \ell), \quad r_{k,\ell} := \Pr(x_{t}^1 = 0 \lor x_{t}^j = 0 \mid x_{t-1}^\ell = x_{t-1}^j = 1, B_t = k, C_{t-1} = \ell).$

Recall that we can assume that the offspring only gets accepted if it has strictly better fitness. Then, the following bounds follow easily, we omit the proof.

Claim 1. It holds that $q_{0,\ell} = 0$, $r_{0,\ell} = 0$, $q_{1,\ell} = 1/n$ and $r_{1,\ell} = 2/n$. For $\ell \leq n/20$ and $k \geq 2$ it holds that $\xi_{k,\ell} \leq 160\ell/n^2$.

Now, the bounds on $\mathbb{E}[V]$ and $\text{Var}(V)$ follow by straightforward calculations. We only give a rough sketch. The last claim says that single bit flips are much more likely to influence the variable $V$.

The claim can be used to bound

\[ \Pr(V_1 = 1) \geq (1 - 1/n)/(1 - \delta/\ln n)(1 - O(1/\ln n)), \quad \text{and} \quad \Pr(V_1 = V_2 = 1) \leq (1 - 1/n)^2(1 - \delta/\ln n + o(n^{-3})). \]

Then, the bounds on $\mathbb{E}[V]$ and $\text{Var}(V)$ follow easily. □

3.2 Proof of Upper Bound

Proof of Theorem 3.3. We give a rough sketch of the proof.

The basic idea is to adapt the upper bound that Witt derived in [13] for the static setup to the scheduled setup. In [13], Witt defines a potential function $X^t = g(x^t)$ by $g(x) = \sum_{i=1}^n g_i x_i$, where $g_1 = 1$, $g_i = \min|y_i, g_{i-1} w_i / w_{i-1}|$, $y_i = (1 + \xi)(1 - p)^{-(n-1)}$, $\xi = \ln n$, and $p = 1/n$. Then, he bounds the multiplicative drift with respect to $g$. More precisely, for any $\xi > 1$ (in Equation (4.1) in [13] this variable is called $\alpha$) and any mutation rate $0 < p < 1$, it holds

\[ \mathbb{E}[(X^t - X^s) \mid X^{t-1} = s] \geq sp(1 - p)^{n-1} (1 - 1/\xi). \] (2)

The key insight is that in the scheduled setup, where the $p_t$ changes every round, one can choose $\xi_t = \xi(1 - p_t)^{n-1}$. Such a trick would not help if we would get a different potential function in each round. However, the potential function $g_t$ defined by $p_t$ and $\xi_t$ coincides with the one defined by $p$ and $\xi$, so all $g_t$ are identical. Thus the definition $X^t = g(x^t)$ still makes sense, and we obtain the following drift with respect to $g$:

\[ \mathbb{E}[(X^t - X^s) \mid X^{t-1} = s] \geq sp_t(1 - p_t)^{n-1} (1 - 1/\xi_t). \] (3)

This bound on the drift together with standard techniques can be used to bound $\mathbb{E}[X^T]$ for some $T = (1 + o(1))\beta n \ln n$. Finally, the theorem follows by Markov’s inequality. □

4 ADAPTIVE SETUP

As mentioned in the introduction, Witt’s proof in [13] can be generalized to obtain the following lower bound for the adaptive setup.

Theorem 4.1 (Lower Bound). For any adaptive choice of mutation rates, the runtime of the $(1 + 1)$ EA on any linear function with $n$ relevant bits is whp at least $(1 - o(1))\ln n$.

Note that this lower bound on the optimization time coincides with the optimization time of the standard $(1 + 1)$ EA if the number of relevant bits $n$ is known. Since it is known that mutation rate $p = 1/n$ is the optimal choice for $n$ relevant bits, it is not surprising that adaptive mutation rates cannot achieve smaller runtime.

Interestingly, we propose an unbiased, comparison-based $(1 + 1)$ EA (see Algorithm 4) with adaptive mutation rate policy that optimizes any linear function with unknown number $n$ of relevant bits in time $(1 + o(1))\ln n$. The idea of Algorithm 4 is the following. Assume we would know a value $m = O(n)$, say for concreteness $n/2 \leq m \leq 2n$. To estimate the exact value of $n$, we start from a random search point $x$, and create an offspring $y$ with a mutation rate of $p = m^{-1-\epsilon}$. Note that $p = o(1/n)$, so it is very unlikely to flip more than one relevant bit. Hence, we may assume that no multi bit flip occurs. Then, the probability to flip a relevant bit is $np$, and with probability $1/2$ it is a 0-bit. Hence, if we repeat the test $m$ times, always starting with a new random $x$, then we expect to see $S = m \cdot np/2$ cases with $f(y) > f(x)$. This number is concentrated, so we may estimate $n = m' := 2S/(mp)$. Afterwards, we optimize with the standard $(1+1)$ EA with mutation rate $p' = 1/m'$. This approach works if we start with an $m$ such that $m/2 \leq m \leq 2n$. However, if $m$ is too small, then the same test will tell us so, since we will get an estimate $m' > 2m$ in this case. Therefore, Algorithm 4 proceeds as follows. In every iteration $m$ is doubled, then an estimate $m'$ of $n$ is computed as described above, and only if $m/2 \leq m' \leq 2m$, the $(1 + 1)$ EA with $p = 1/m'$ is run for $10m'$ steps. Formally, we show the following theorem.

Theorem 4.2 (Upper Bound). The optimization time of Algorithm 4 on any linear function $f$ with $n$ relevant bits is whp at most $(1 + o(1))\ln n$.

Proof Sketch. Note that we do exponential search (the $m$ is doubled in every round) and that for each $m$ the estimation part of the algorithm needs $2m$ function evaluations and the optimization part needs $O(m \ln m)$ function evaluations if it is executed and otherwise 0. The theorem can be proven using standard concentration inequalities and by dividing the execution of the algorithm into three phases according to the current size of $m$. While $m \leq \sqrt{n}$, the number of function evaluations is at most $O(\sqrt{n} \ln^2 n)$. While $\sqrt{n} \leq m \leq n/100$, the optimization part of the algorithm will not be executed, and therefore the number of function evaluations is at most $O(n)$. For $n/100 \leq m \leq 2n$ the estimate $m'$ of $n$ is so precise such that $f$ is optimized within $(1 + o(1))\ln n$ rounds the first time the optimization part of the algorithm is executed. Before that only $O(n)$ function evaluations are required. □

5 INITIAL SEGMENT UNCERTAINTY MODEL

In this section we analyze the runtime of the $(1+1)$ EA with position dependent mutation rates $\bar{p}$ on the OneMax function with support on the initial segment $[n]$. For every summable and monotone decreasing sequences $\bar{p}$ the expected runtime is upper bounded by $O((\ln n)/\bar{p}_n)$, cf. Theorem 14 in [3]. Note that it is advantageous for this upper bound to take a summable sequence that decays as slowly as possible. However, it is known that there exist no smallest decaying summable sequence (cf. Section 2.6 in [4]).
The linear hidden subset problem for the (1+1) EA with scheduled and adaptive mutation rates
GECCO ’18, July 15–19, 2018, Kyoto, Japan

Theorem 5.1. Let \( \bar{p} \) be a monotone decreasing sequence with \( p_1 \leq 1/2 \), and let \( \bar{q} \) be an arbitrary non-summable sequence. Then there is a constant \( c > 0 \) such that for infinitely many \( n \in \mathbb{N} \) the expected optimization time of the (1+1) EA with position dependent mutation probabilities \( \bar{p} \) on the OneMax function with support on the initial segment \( [n] \) is at least \( c \ln(n)/q_n \).

We denote the \( k \)-fold iterative logarithm \( \ln(\ldots \ln(x)) \) by \( \ln^{(k)}(x) \). Since the sequence \( q_n = 1/(n \prod_{i=1}^{n} \ln^{(i)}(n)) \) is non-summable [4, Lemma 2.4], we obtain the following upper bound.

Corollary 5.2. Let \( \bar{p} \) be a monotone decreasing sequence with \( p_1 \leq 1/2 \). Then, there is \( c > 0 \) such that the expected optimization time of the (1+1) EA with position dependent mutation probabilities \( \bar{p} \) on the OneMax function with support on the initial segment \( [n] \) is at least \( cn \ln^2(n) \prod_{j=2}^{\infty} \ln^{(j)}(n) \) for infinitely many \( n \in \mathbb{N} \).

Note that this upper bound is tight in the sense that for any \( k \geq 0 \) the summable sequence \( p_n := 1/(n \prod_{j=1}^{k} \ln^{(j)}(n)) \) achieves a runtime of \( O(n \ln^2(n) \prod_{j=2}^{k} \ln^{(j)}(n)) \).

5.1 Proof of Lower Bound

In the following, we assume that the position dependent mutation rates \( \bar{p} \) are monotonically decreasing and smaller than 1/2. Further, we define \( S_n = \sum_{i=1}^{\lfloor n/2 \rfloor} p_i \) and \( S = \lim_{n \to \infty} S_n \).

Before we come to the technical details proof, we first give an overview over the proof. The crucial step will be to show that the expected runtime is at least \( \Omega(\ln(n) M_n) \), where \( M_n := \min \left\{ e^{S_n/(S_n p_{\lfloor n/2 \rfloor})} n^{1+\epsilon} / \ln(n) \right\} \). This will be done in Lemma 5.3 for the case that \( \bar{p} \) is summable, where the formula can be simplified. The hard part of the proof is to show this bound for non-summable \( \bar{p} \), which is done in Lemma 5.4. Afterwards, we show by a rather short argument in Lemma 5.5 that the inverse of the sequence \( M_n \) is summable for every monotone sequence \( \bar{q} \), and that for any non-summable sequence \( \bar{q} \) we have \( M_n \geq 1/q_n \) for infinitely many values of \( n \).

We start with the lower bound on the optimization time. The first lemma assumes that \( \bar{p} \) is summable, and it follows rather easily from the fact that every bit in \( [(n/2), \ldots, n] \) needs to flip at least once. We omit the proof.

Lemma 5.3. Let \( \bar{p} \) be such that \( S < \infty \). Then the expected optimization time of the (1+1) EA with position dependent mutation rates \( \bar{p} \) on the OneMax function with support on the initial segment \( [n] \) is at least \( \Omega(\ln(n)/p_{\lfloor n/2 \rfloor}) \).

A similar bound holds for non summable sequences, but is much harder to prove. Note that the first term \( \ln(n)^{e^{S_n}/S_n p_{\lfloor n/2 \rfloor}} \) in the bound in Lemma 5.4 generalizes the bound in Lemma 5.3, since there we have \( S_n = \Theta(1) \).

Lemma 5.4. Let \( \bar{p} \) be such that \( S = \infty \). The optimization time of the (1+1) EA with position dependent mutation rates \( \bar{p} \) on the OneMax function with support on the initial segment \( [n] \) is at least \( \Omega(\min \{ \ln(n)^{e^{S_n}/S_n p_{\lfloor n/2 \rfloor}}, n^{1+1/\delta} \}) \).

Proof sketch. We first observe that the bound is easy in some cases: if \( p_n \leq n^{-1-\delta} \) for \( \delta := 0.1 \), then it takes a long time to flip the \( n \)-th bit, and if \( S_n \geq n^\delta \), then it takes a long time to make the very last step towards the optimum, because we typically flip many bits at once. So we assume that none of these cases happen.

We argue by pigeonhole principle that there is a medium sized set \( B \) of \( (|B| = n^{1-2\delta}) \) such that all \( p_i \in B \), differ by at most a factor of 2, and such that \( p_1 = O(p_{\lfloor n/2 \rfloor}) \). In particular, it can be shown that \( \sum_{i \in B} p_i = o(1) \). We consider the case that \( B \) is close to optimal, i.e., that the number \( B_1(t) \) of \( 1 \)-bits in \( B \) is at most \( t |B| \) for some \( \epsilon < n^{-3/4} \). Then, we study the drift \( \Delta_t := B_1(t) - B_1(t+1) \). The main part of the proof is to show that \( E[\Delta_t] \leq C \epsilon |B| p_{\lfloor n/2 \rfloor} n^{1-\epsilon} \) for a constant \( C \), from which the theorem follows by a lower bound multiplicative drift theorem [13].

Note that the term \( C \epsilon |B| p_{\lfloor n/2 \rfloor} n^{1-\epsilon} \) roughly resembles the probability that exactly one \( 1 \)-bit is flipped in \( B \) (probability \( \approx \epsilon |B| p_{\lfloor n/2 \rfloor} \)) and at most one bit is flipped in \( A := [n] \setminus B \) (probability \( \approx S_n e^{-S_n} \)). However, it would be incorrect to say that this is the leading term. It is not even necessarily the leading term among those terms that contribute positively to the drift. For example, consider the case that \( A \) is not well-optimized, for illustration we may imagine that all of these bits are \( 1 \)-bits. Then, a much more likely scenario for an improvement in \( B \) is that many bits in \( A \) are flipped (which improves the fitness, and has probability \( \Theta(1) \) instead of \( O(S_n e^{-S_n}) \)). In addition, one \( 1 \)-bit in \( B \) is flipped. However, in this case there is an even more likely scenario: a similar combination of bits in \( A \) is flipped, a 0-bit in \( B \) is flipped, and thus \( B_1(t) \) moves away from the optimum. (In general, the situation is more complex than for the case that \( A \) consists only of 1-bits.) So what we really show is that all terms that contribute positively to the drift are at most most
\( C\frac{\beta}{2\sqrt{2S}} e^{-2S}, \) or they are counterbalanced by even larger terms that contribute negatively to the drift. Nevertheless, this gives a drift bound of \( \mathbb{E}[\Delta_t] \leq C\frac{\beta}{2\sqrt{2S}} e^{-2S}, \) as required.

It remains to show the bound \( \mathbb{E}[\Delta_t] \leq C\frac{\beta}{2\sqrt{2S}} e^{-2S} \) when \( |B_t| = e|B| \) for \( c < n^{-3/4}. \) Denote by \( F_k \) the event that \( k \) bits flip in \( B. \) First, we show that no matter which bit flips in \( A \) we condition on, for \( k \geq 2 \) it holds that \( \mathbb{E}[\Delta_t | F_k] = O(e^t). \) Then, we denote by \( A_0 \) the number of 0-bits in \( A \) that flip to 1 and by \( A_10 \) the number of 1-bits in \( A \) that flip to 0. In order to bound \( \mathbb{E}[\Delta_t] \), we write it as a sum of 6 terms \( D_1, \ldots, D_6 \) defined as follows:

\[
\begin{align*}
D_1 & = \Pr(A_01 < A_10)\mathbb{E}[\Delta_t | A_01 < A_10] \\
D_2 & = \Pr(A_01 = A_10 = 0)\mathbb{E}[\Delta_t | A_01 = A_10 = 0] \\
D_3 & = \Pr(A_01 = A_10 > 0)\mathbb{E}[\Delta_t | A_01 = A_10 > 0] \\
D_4 & = \Pr(A_01 = 1 + A_10 = 1)\mathbb{E}[\Delta_t | A_01 = A_10 = 1] \\
D_5 & = \Pr(A_01 = 1 + A_10 > 1)\mathbb{E}[\Delta_t | A_01 = 1 + A_10 > 1] \\
D_6 & = \Pr(A_01 \geq 2 + A_10)\mathbb{E}[\Delta_t | A_01 \geq 2 + A_10].
\end{align*}
\]

The idea is the following. If \( A_01 = A_10 = 0 \) or \( A_01 = A_10 + 1 = 1, \) which happens with probability at most \( e^{-2n} \) and \( S_\infty e^{-2S}, \) respectively, then the leading term of \( \mathbb{E}[\Delta_t] \) is the probability that we flip one 1-bit in \( B \) which is at most \( 2e|B|_p \), where \( p_B \) is such that \( p_B \leq p_i \leq 2p_B \) for \( i \in B. \) Therefore, \( D_2 + D_3 \leq 3e|B|_p e^{-2S}. \) If \( A_01 \geq 2 + A_10, \) then \( \mathbb{E}[\Delta_t] = O(e^t) \) since the offspring is only accepted if at least two 1-bits flip in \( B. \) Therefore, \( D_6 = O(e^t). \) In the other cases \( A_01 \geq 1, \) thus it is likely that the number of 1-bits decreases in \( A, \) which will cause \( \mathbb{E}[\Delta_t] \) to be negative. To see this, note that it is much more likely that a 0-bit flips to 1 than that a 1-bit flips to 0 in \( B. \) If more than one bit flips in \( B, \) then \( \mathbb{E}[\Delta_t] \leq O(e^t). \) Those ideas allow us to show \( D_1 + D_2 + D_3 \leq 0 \). These bounds imply the bound on the drift.

The next lemma links the bound on the runtime with non-summable sequences.

**Lemma 5.5.** Let \( (p_n)_{n \geq 1} \) and \( (q_n)_{n \geq 1} \) be sequences of positive reals with \( p_n \leq 1 \) for all \( n \in \mathbb{N}. \) Let \( I \subseteq \mathbb{N} \) be the set of all indices \( n \) such that

\[
\min \left\{ \frac{n!^{-1}}{q_n}, \frac{\exp\left(\sum_{i=1}^{n} p_i\right)}{n! \cdot \prod_{i=2}^{n} p_i} \right\} \leq 1.
\]

Then \( \sum_{n \in I} q_n < \infty. \) In particular, if \( \sum_{n=0}^{\infty} q_n = \infty \) then \( \sum_{n \in \mathbb{N} \setminus I} q_n = \infty. \)

**Proof Sketch.** We take the inverse of (4), replace the resulting maximum of two terms by their sum, and sum over all \( n \in I. \) Then \( \sum_{n \in I} q_n \) is bounded by the sum of two sums, and we must show that each of them converges. The most difficult one is

\[
S_1 := \sum_{n=1}^{\infty} \frac{p_{[n/2]} f(k-1)}{\prod_{i=1}^{n} p_i},
\]

where \( f(x) := x e^{-x}, \) in the case that \( \sum_{n=1}^{\infty} p_n = \infty. \) For all \( k \in \mathbb{N} \) we define \( n_k := \min\{n \geq 1 \mid \sum_{i=1}^{n_k} p_i \geq k\}. \) Then we compute, omitting details,

\[
S_1 \approx \sum_{k=1}^{\infty} \sum_{n=2n_k-2}^{2n_k-2} \frac{p_{[n/2]} f(k-1)}{\prod_{i=1}^{n} p_i} = \sum_{k=1}^{\infty} \Theta(k) \cdot f(k-1) < \infty.
\]

**Proof of Theorem 5.1.** Let \( \widetilde{p} \) be any monotone decreasing sequence with \( p_1 \leq 1/2, \) and let \( \widetilde{q} \) be any non-summable sequence. If \( \widetilde{p} \) is summable, then Lemma 5.3, otherwise Lemma 5.4, implies that the expected optimization time is \( \Omega(\ln(n)e^{en}/S\beta n^{1/2}, n^{1/1}). \) Then, Lemma 5.5 implies the theorem.

**6 CONCLUSIONS**

We have precisely analyzed the optimal strategies for the hidden subset problem for the scheduled and the adaptive setup. Both are asymptotically faster than the best strategy for the static setup. For the adaptive setup, the unknown \( n \) does not increase the runtime. For the non-adaptive setup, there is a price to pay, namely we lose a factor of \( \beta/e \approx 1.307 \) in the runtime. The best algorithm in this case follows a rather natural schedule \( p_i = \alpha i/\ln i, \) except for the surprising factor \( \alpha. \) The best schedule is surprisingly rigorously determined, and even slight deviations from the optimal schedule lead to a loss in performance. On the other hand, the algorithm that achieves runtime \( (1 \pm o(1))\ln n \) in the adaptive case is arguably rather artificial and ad hoc. Most common strategies like the 1/5-rule adapt the mutation rate in small steps, see [7, 8] for reviews. It is an interesting question whether the same runtime can be achieved with such strategies.

Another intriguing question is on the connection between the hidden subset problem and the initial segment uncertainty model. On all studied fitness functions, the optimal runtimes of these algorithms are asymptotically equal – for LEADINGONES the connection is even more intimate. It remains an open question whether a general connection can be found between the two models.

**REFERENCES**

[1] Stephan Catthubard, Per Kristian Lehre, and Xin Yao. 2011. Non-uniform mutation rates for problems with unknown solution lengths. In Foundations of Genetic Algorithms (FOGA). ACM, 173–180.
[2] Thomas H. Cormen, Charles E. Leiserson, Robert L. Rivest, and Clifford Stein. 2001. Introduction to Algorithms (2. ed.). MIT Press.
[3] Benjamin Doerr, Carola Doerr, and Timo Kötzing. 2015. Solving problems with unknown solution length at (almost) no extra cost. In Genetic and Evolutionary Computation Conference (GECCO). ACM, 831–838.
[4] Benjamin Doerr, Carola Doerr, and Timo Kötzing. 2017. Unknown solution length problems with no asymptotically optimal run time. In Genetic and Evolutionary Computation Conference (GECCO). ACM, 1367–1374.
[5] Benjamin Doerr, Carola Doerr, and Jing Yang. 2016. Optimal parameter choices via precise black-box analysis. In Genetic and Evolutionary Computation Conference (GECCO). ACM, 1123–1130.
[6] Carola Doerr and Johannes Lengler. 2017. OneMax in black-box models with several restrictions. Algorithmica 78, 2 (2017), 610–640.
[7] Ágoston E Eiben, Robert Hinterding, and Zbigniew Michalewicz. 1999. Parameter control in evolutionary algorithms. IEEE Transactions on Evolutionary Computation 3, 2 (1999), 124–141.
[8] Giorgos Karafotias, Mark Hoogendoorn, and Ágoston E Eiben. 2015. Parameter control in evolutionary algorithms: Trends and challenges. IEEE Transactions on Evolutionary Computation 19, 2 (2015), 167–187.
[9] Elias Koutsoupias and Christos Papadimitriou. 1999. Worst-case equilibria. In Symposium on Theoretical Aspects of Computer Science (STACS), Vol. 99. Springer, 404–413.
[10] Per Kristian Lehre and Xin Yao. 2014. Runtime analysis of the (1 + 1) EA on computing unique input output sequences. Information Sciences 259 (2014), 510–531.
[11] Tim Roughgarden and Eva Tardos. 2002. How bad is selfish routing? Journal of the ACM (JACM) 49, 2 (2002), 236–259.
[12] Carsten Witt. 2012. Optimizing linear functions with randomized search heuristics—the robustness of mutation. In Symposium on Theoretical Aspects of Computer Science (STACS), Vol. 14. LIPIcs. 420–431.
[13] Carsten Witt. 2013. Tight bounds on the optimization time of a randomized search heuristic on linear functions. Combinatorics, Probability and Computing 22, 2 (2013), 294–318.