Population Dynamics and Improved Runtime Guarantees for the $(\mu + 1)$ EA on BinVal

Martin S. Krejca
LIX, CNRS, École Polytechnique,
Institut Polytechnique de Paris
Palaiseau, France
martin.krejca@polytechnique.edu

Frank Neumann
Optimisation and Logistics,
The University of Adelaide
Adelaide, Australia
frank.neumann@adelaide.edu.au

Carsten Witt
Technical University of Denmark
Lyngby, Denmark
cawi@dtu.dk

Abstract

Populations play a key role in the area of evolutionary computation to tackle complex optimization problems. Nevertheless, it is hard to understand the underlying population dynamics from a theoretical perspective, and only a limited number of theoretical results for population-based algorithms are available even for simple benchmark functions. In this paper, we study the classic (μ+1) EA on the benchmark problem BinVal, which allows for exponentially many function values. Previous methods for the analysis, based on fitness levels and multiplicative drift analysis, lead to runtime bounds for this function of size n that include an additive term of $\Theta(n^2)$. We provide new insights into how this standard algorithm optimizes BinVal, and we provide runtime bounds that are polynomial in the population size μ and do not include this additive term. In particular, we prove bounds on the expected runtime that are $O(\mu^5 n \log(n/\mu^4))$ for standard bit mutation, which is $O(n \log n)$ for constant μ. Our analysis considers the population dynamics of the (μ+1) EA more closely, proving that copies created by mutation lead to a low diversity in short blocks of bits across all individuals. We extend this method to mutation operators that cannot create duplicates, and prove bounds similar to standard bit mutation.

ACM Reference Format:
Martin S. Krejca, Frank Neumann, and Carsten Witt. 2025. Population Dynamics and Improved Runtime Guarantees for the $(\mu + 1)$ EA on BinVal. In *Foundations of Genetic Algorithms XVIII (FOGA '25), August 27–29, 2025, Leiden, Netherlands.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3729878.3746614

1 Introduction

For more than 20 years, the rigorous analysis of evolutionary algorithms (EAs) has provided deep insights into their behavior in various settings [9]. This mathematical foundation has led to the design of improved algorithms [e. g., 11, 12, 21] and shown the inherent strong performance of EAs in different areas, such as when employing crossover [29], when utilizing them as algorithm configurators [19], or even as black-box solvers for zero-sum games [3].

Despite all these achievements, certain aspects of EAs remain not well understood. One of the most glaring ones is the impact of the population size on an EA's performance. This is particularly interesting as practical EAs are commonly applied with larger populations. This lack of understanding is partially due to the challenge of keeping track of multiple solutions and their interactions at the same time over the algorithm's run, known as *population dynamics*.

In the theory of EAs, the prototypical population-based EA is the (μ+1) EA, which keeps μ solutions and updates this population iteratively by replacing one worst existing solution with a newly created one, provided that the latter is not worse than the former. Although this framework is very similar to the classic (1+1) EA, which only maintains a single solution, the (μ+1) EA is substantially less studied than the (1+1) EA [9]. A reason for this drastic difference may be the more challenging population dynamics of the (μ+1) EA.

The first theoretical study of the (μ+1) EA dates back to Witt [31], who analyzed the algorithm on the well-known OneMax and LeadingOnes functions and another, lesser studied, function. The results show that the population does not reduce the expected runtime compared to running the algorithm with a single individual. However, the best possible expected runtime is achieved for a certain range of the population size μ, showing that despite a smaller population size being generally better in this setting, the algorithm is robust, to an extent, to a sub-optimal choice of μ.

Further theoretical works study the (μ+1) EA on monotonic functions [23, 26]. Combined, these papers show that for each mutation rate, there exists a monotonic function and a population size (at most linear in the problem size) such that the (μ+1) EA does not optimize the function in expected polynomial time. This is in contrast to the (1+1) EA, which is capable to optimize all monotonic functions efficiently if the mutation rate is less than 1 [23].

This result is indirectly complemented by the work of Lengler and Riedi [25], who proved that the (μ+1) EA efficiently optimizes a dynamic version of the BinVal function if initialized closely to the optimum, whereas the (1+1) EA exhibits an expected exponential runtime in this same setting. This shows the benefit of a population.

The results for the (μ+1) EA on noisy functions are mixed, with Gießen and Kötzing [17] showing that the algorithm performs well for one-bit noise, whereas Friedrich et al. [16] show that it can perform poorly under additive posterior Gaussian noise.

Besides these results, we mention that the variant of the (μ+1) EA with crossover (known as the (μ+1) GA) has been studied in a longer line of research on the Jump function [29, and the references therein], culminating recently in a proven super-exponential speed-up for a variant of this algorithm [29]. This result builds upon a more thorough understanding of the population dynamics of the (μ+1) EA and (μ+1) GA on plateaus of equal fitness [27]. Nonetheless, to this day, the population dynamics of the (μ+1) EA on many more standard theory benchmarks are poorly understood.

ACM ISBN 979-8-4007-1859-5/2025/08
https://doi.org/10.1145/3729878.3746614

A very prominent example benchmark for which we are currently lacking any deep theory for the (μ+1) EA is the BinVal function, which returns the binary value of a bit string. It is a linear function with exponentially many function values, and it is known that the population-less (1+1) EA optimizes this function of problem size n in expected $O(n \log n)$ function evaluations [15]. Interestingly, we note that this function is not that well understood for EAs in general, whereas there are several results for the runtime of estimation-of-distribution algorithms (EDAs) [e. g., 1, 2, 5, 8, 14, 32], which use populations in a different way. The runtime bounds vary heavily with the underlying EDA and range from $\Theta(n)$ over $\Theta(n^2)$ to $O(n^2 \log^3 n)$, for reasonable parameter choices.

Our contribution. We study the (μ+1) EA on the BinVal function and gain a better understanding of the algorithm's population dynamics. Different from traditional methods, which always introduce an additive term of $\Theta(n^2)$ in the runtime (see Section 3), our results only feature multiplicative terms dependent on μ. For standard bit mutation, we prove an expected runtime of $O(\mu^5 n \log(n/\mu^4))$ function evaluations (Theorem 4). For constant values of μ, this simplifies to $O(n \log n)$, matching the runtime of the (1+1) EA. And for $\mu = o((n/\log n)^{1/5})$, this still results in a $o(n^2)$ expected runtime.

As we detail in Section 3, a major part in our analysis is a bound on the maximum Hamming distance in the population across a block of bit positions (Theorem 6). For an appropriately chosen block size, the expected maximum Hamming distance within a block is at most $\frac{1}{4}$. This allows one to apply the multiplicative drift theorem [10] to derive our improved runtime bounds, following the original analysis of Droste et al. [15] for the (1+1) EA.

We derive Theorem 6 via an intricate theorem (Theorem 1) that bounds the expected value of a non-negative random process that has a high likelihood to be 0 but is allowed to perform jumps to large positive values, from where it returns to 0 again under additive drift. Our bound for the additive drift relies on a bound on the moment-generating function of the process. As we explain in Section 2.2, Theorem 1 is inspired by a theorem by Hajek [18] on occupation probabilities. This theorem by Hajek also requires a bound on the moment-generating function of the process, which results in strong tail bounds for the process for larger positive values. However, these bounds are not particularly strong for the state 0, which is the state that we aim for. Theorem 1 addresses this problem.

The proof of Theorem 6 relies on creating copies of the best individual in order to show the drift condition of Theorem 1, which in turn results in the low maximum Hamming distance per block. In order to understand whether the ability of standard bit mutation to likely create copies is important, we furthermore study the (μ+1) EA with 1-bit and plus mutation, the latter of which is standard bit mutation conditional on not creating a copy. To this end, we derive Theorem 8, which also bounds the expected maximum Hamming distance within a block. With this tool, we derive runtime results similar to those for standard bit mutation. In Theorem 7, we prove an expected runtime of $O\big(\mu^6 n \log(n/\mu^5)\big)$ fitness evaluations for computing a $1/\mu$-approximation of BinVal, and we prove an expected runtime of $O\big(\mu^3 n^{3/2} \log(n/\mu^2)\big)$ for optimization.

All our theoretical results hold for any constant mutation rate in the cases of standard bit and plus mutation. However, for the sake of readability, we restrict our analyses to mutation rate 1.

Last, we complement our results with experiments that consider the (μ+1) EA with the three mutation operators above on BinVal as well as on random linear functions and a linear function with increasing weights. These experiments suggest that the plus operator is worse than the standard bit and 1-bit mutation, both of which are hardly distinguishable. Moreover, the results suggest that BinVal is a particularly hard linear function for the (μ+1) EA.

Outline. In Section 2, we define our notation as well as our setting, and we present Theorem 1, which is the backbone of our analysis on low diversity per block. Section 3 contains our runtime analysis for standard bit mutation, and Section 4 for 1-bit and plus mutation. Section 5 explains our experimental setup and discusses our findings. We conclude our paper in Section 6.

2 Preliminaries

For all $m, n \in \mathbb{N}$, we let $[m..n] \coloneqq [m, n] \cap \mathbb{N}$ and $[n] \coloneqq [1..n]$. We work in the search space $\{0, 1\}^n$ and consider the maximization of pseudo-Boolean *(fitness)* functions $f\colon \{0, 1\}^n \to \mathbb{R}_{\geq 0}$. We call each $x \in \{0, 1\}^n$ an *individual* and denote each position $i \in [n]$ by x_i. For all $x, y \in \{0, 1\}^n$, let $d_{\mathrm{H}}(x, y)$ denote the Hamming distance of x and y. For an event A, let $\mathbb{1}\{A\}$ denote the indicator function for A. Last, for all random variables X and Y as well as for all events A with $\Pr[A] > 0$, let $\mathrm{E}[X \mid Y; A] \coloneqq \mathrm{E}[X \cdot \mathbb{1}\{A\} \mid Y]/\Pr[A \mid Y]$.

2.1 Our Setting

Benchmark problem. We analyze the function BinVal, defined for all $x \in \{0, 1\}^n$ as $\textsc{BinVal}(x) = \sum_{i \in [n]} 2^{n-i} \cdot x_i$.

Algorithm. We study the classic (μ+1) EA [31] (Algorithm 1). It works with a population P, which is a multi-set of size $\mu \in \mathbb{N}_{\geq 1}$ of individuals. In each iteration, an individual $x \in P$ is chosen from P uniformly at random and an offspring y is produced from x by mutation, the details of which we specify below. The population P is updated including y and removing an individual z with the worst fitness in P if $f(y) \geq f(z)$ holds. Otherwise, y is discarded.

Mutation. We consider three commonly applied types of mutation. Each of them gets a *parent* individual $x \in \{0, 1\}^n$ and produces a new *offspring* individual $y \in \{0, 1\}^n$ by creating a copy of x and then flipping bits in this copy via various means.

Standard bit mutation decides for each bit in the copy independently to flip it with probability $\frac{1}{n}$. Thus, we have for all $i \in [n]$ that $\Pr[y_i = 1 - x_i] = \frac{1}{n}$ and $\Pr[y_i = x_i] = 1 - \frac{1}{n}$.

1-bit mutation flips exactly one bit in the copy, chosen uniformly at random. That is, for an $i \in [n]$ uniformly at random, we have $y_i = 1 - x_i$, and thus for all $j \in [n] \setminus \{i\}$, we have $y_i = x_i$. We note that 1-bit mutation is the standard mutation for the popular algorithm *randomized local search* (RLS) [7].

Last, *plus mutation* [4] operates the same way as standard bit mutation but enforces to flip at least one bit by repeating standard bit mutation until the offspring differs from the parent.

Runtime. We analyze the (μ+1) EA with respect to the number of fitness evaluations until a globally optimal solution has been produced for the first time. We assume that an individual is evaluated exactly once, namely, at the point of its creation. Since the (μ+1) EA produces exactly one individual per iteration and starts with a population of μ individuals, its runtime is μ plus the number of iterations until an optimum is found for the first time.

Algorithm 1: The $(\mu$+1) EA with population size $\mu \in \mathbb{N}_{\geq 1}$ and mutation operator mut: $\{0,1\}^n \to \{0,1\}^n$, maximizing a given pseudo-Boolean function $f\colon \{0,1\}^n \to \mathbb{R}$.

```
1 $t \leftarrow 0$;
2 $P_t \leftarrow \mu$ individuals from $\{0,1\}^n$ drawn uniformly at random
    (u.a.r.), with repetition;
3 while termination criterion not met do
4     $x \leftarrow$ select an individual in $P_t$ u.a.r;
5     $y \leftarrow \mathrm{mut}(x)$;
6     $z \leftarrow \arg\min_{x' \in P_t} f(x')$;
7     if $f(y) \geq f(z)$ then $P_{t+1} \leftarrow (P_t \setminus \{z\}) \cup \{y\}$;
8     else $P_{t+1} \leftarrow P_t$;
9     $t \leftarrow t+1$;
```

2.2 Mathematical Tools

Our proofs make use of drift analysis [24] to show that the diversity of the population of the $(\mu$+1) EA, measured in the maximum Hamming distance within a block of consecutive bits, tends to small values and is expected to stay small. Besides hitting times, we are therefore also concerned with bounds on occupation probabilities and expected states for processes exhibiting additive drift to a small target state like state 0. We note that already the seminal paper by Hajek [18] on drift analysis states powerful theorems on occupation probabilities that have been applied in several previous runtime analyses, e. g., [13, 28]. However, surprisingly, Hajek's theorems seem too weak for our purposes. We are confronted with a boundary state that has a very low probability of being left, but Hajek's framework seems more suited for processes on the real line that satisfy a strong uniform drift bound within a certain interval. In particular, it does not use additional information on single states with an exceptionally low transition probability.

To address this gap, we give a self-contained occupation probability theorem that is suited for our scenario. We present it in a general fashion and believe it to be useful in other contexts. Except for well-established tail bounds for additive drift, the proof of our new theorem uses only elementary arguments.

Theorem 1. *Let $(X_t)_{t\in\mathbb{N}}$ be a random process, adapted to a filtration $(\mathcal{F}_t)_{t\in\mathbb{N}}$, over a state space $S \subseteq \mathbb{R}_{\geq 0}$ with $0 \in S$. Suppose there are $\lambda \in \mathbb{R}_{>0}$, $\rho \in [0,1)$, $p_{\text{jump}} \in (0,1)$, $p_{\text{safe}} \in [0,1)$, and $s_{\text{jump}}, s_{\text{safe}} \in \mathbb{R}_{>0}$ such that the following holds for all $t \in \mathbb{N}$:*

(1) $\mathrm{E}[e^{\lambda(X_{t+1}-X_t)} \mid \mathcal{F}_t] \cdot \mathbb{1}\{X_t > 0\} \leq \rho$.
(2) $\Pr[X_{t+1} > 0 \mid X_t = 0] \leq p_{\text{jump}}$.
(3) $\Pr[X_{t+1} > s_{\text{safe}} \mid X_t = 0] \leq p_{\text{safe}}$.
(4) $\mathrm{E}[X_{t+1} \mid \mathcal{F}_t; X_t = 0 \wedge X_{t+1} > 0] \leq s_{\text{jump}}$.
(5) $X_0 = 0$.

Then for all $t \in \mathbb{N}$, we have

$$\mathrm{E}[X_t] \leq \left(2p_{\text{jump}}\left\lceil \frac{\ln(p_{\text{jump}}) - \lambda s_{\text{safe}}}{\ln \rho}\right\rceil + p_{\text{safe}}\right)s_{\text{jump}}. \qquad (1)$$

If only assumptions (1)–(4) are satisfied, then (1) holds for all $t \geq T$, where $T \coloneqq \inf\{t \geq 0 \mid X_t = 0\}$, and we have $\mathrm{E}[T] \leq \frac{\lambda X_0}{1-\rho}$.

For the proof of Theorem 1, we make use of a simple tail bound for processes with additive drift, very similar to those for multiplicative drift. We use it in order to show that the process quickly reaches 0 again once it leaves this value. We do not claim the tail bound to be new. It follows well-established ideas from papers on drift analysis [18] and appears as a special case of Theorem 3.2(iii) in [22]. For the sake of completeness and clarity, we give the simple result and its short proof here.

Theorem 2. *Let $(X_t)_{t\in\mathbb{N}}$ be a stochastic process, adapted to a filtration $(\mathcal{F}_t)_{t\in\mathbb{N}}$, over some state space $S \subseteq \mathbb{R}$, where $0 \in S$. Assume there exist $\lambda \in \mathbb{R}_{>0}$ and $\rho \in [0,1)$ such that for all $t \in \mathbb{N}$, we have*

$$\mathrm{E}[e^{\lambda(X_{t+1}-X_t)} \mid \mathcal{F}_t] \cdot \mathbb{1}\{X_t > 0\} \leq \rho.$$

Let $T \coloneqq \inf\{t \in \mathbb{N} \mid X_t \leq 0\}$. Then, for all $t \in \mathbb{N}$, we have

$$\Pr[T > t \mid \mathcal{F}_0] \leq e^{\lambda X_0}\rho^t.$$

Proof. We artificially extend the process to all reals to satisfy the drift condition also when $X_t \leq 0$, i. e., we also assume for all $t \in \mathbb{N}$ that $\mathrm{E}[e^{\lambda(X-t+1-X_t)} \cdot \mathbb{1}\{X_t \leq 0\} \mid \mathcal{F}_t] \leq \rho$, resulting in $\mathrm{E}[e^{\lambda(X_{t+1}-X_t)} \mid \mathcal{F}_t] \leq \rho$. This does not change the distribution of T.

We prove via induction on $t' \in \mathbb{N}$ that $\mathrm{E}[e^{\lambda X_{t'}} \mid \mathcal{F}_0] \leq e^{\lambda X_0}\rho^{t'}$, noting that we omit from now on the indicator function in the assumption, since it now holds for all signs of X.

The base case holds trivially, as X_0 is $\mathcal{F}_0$-measurable. For the inductive step for $t' \in \mathbb{N}$, using the law of total expectation, that X is adapted to $\mathcal{F}$, the assumption of the theorem, as well as the induction hypothesis, we obtain

$$\begin{aligned}
\mathrm{E}[e^{\lambda X_{t'+1}} \mid \mathcal{F}_0] &= \mathrm{E}[e^{\lambda(X_{t'+1}-X_{t'}+X_{t'})} \mid \mathcal{F}_0] \\
&= \mathrm{E}[\mathrm{E}[e^{\lambda(X_{t'+1}-X_{t'})}e^{\lambda X_{t'}} \mid \mathcal{F}_{t'}] \mid \mathcal{F}_0] \\
&= \mathrm{E}[\mathrm{E}[e^{\lambda(X_{t'+1}-X_{t'})} \mid \mathcal{F}_{t'}]e^{\lambda X_{t'}} \mid \mathcal{F}_0] \\
&\leq \rho \mathrm{E}[e^{\lambda X_{t'}} \mid \mathcal{F}_0] \leq e^{\lambda X_0}\rho^{t'+1}.
\end{aligned}$$

By Markov's inequality, we see that

$$\Pr[X_t > 0 \mid \mathcal{F}_0] = \Pr[e^{\lambda X_t} > e^{\lambda 0} \mid \mathcal{F}_0] \leq \mathrm{E}[e^{\lambda X_t} \mid \mathcal{F}_0]/e^{\lambda 0} \leq e^{\lambda X_0}\rho^t.$$

Since $\Pr[T > t \mid \mathcal{F}_0] \leq \Pr[X_t > 0 \mid \mathcal{F}_0]$, the theorem follows. □

We now prove Theorem 1 via a case distinction on how quickly the process X returns to 0 once it leaves this value.

Proof of Theorem 1. We first prove the statement on $\mathrm{E}[X_t]$. Let $t \in \mathbb{N}_{>0}$, as the statement is trivial for $t = 0$, due to assumption 5. Since X only takes on non-negative values, if $\Pr[X_t > 0] = 0$, then $\mathrm{E}[X_t] = 0$, and the statement holds trivially. Hence, we assume in the following that $\Pr[X_t > 0] > 0$.

By the law of total expectation and $\mathrm{E}[X_t \mid X_t = 0] = 0$, it suffices to consider $\mathrm{E}[X_t] = \mathrm{E}[X_t \mid X_t > 0] \cdot \Pr[X_t > 0]$. In the following, we bound $\mathrm{E}[X_t \mid X_t > 0]$ and $\Pr[X_t > 0]$ separately from above.

For bounding $\mathrm{E}[X_t \mid X_t > 0]$, recalling assumption 4, we claim for all $t' \in \mathbb{N}_{>0}$ that $\mathrm{E}[X_{t'} \mid X_{t'} > 0] \leq s_{\text{jump}}$. We prove this claim via induction on $t' \in \mathbb{N}_{>0}$.

The base case holds due to assumptions 4 and 5. For the inductive step for $t' \in \mathbb{N}$, we make a case distinction with respect to the event $\{X_{t'} = 0\}$.

In the case of $X_{t'} = 0$, the claim follows from assumption 4.

If $X_{t'} > 0$, we obtain via Jensen's inequality, the concavity of ln, and assumption 1 that conditional on $X_{t'>0}$, defining for all $x \in \mathbb{R}_{>0}$ the notation $\ln^+(x) \coloneqq \ln(x)$ as well as $\ln^+(0) \coloneqq 0$, that

$$\begin{aligned} &\mathrm{E}[\lambda(X_{t'+1} - X_{t'}) \mid \mathcal{F}_{t'}] \cdot \mathbb{1}\{X_{t'} > 0\} \\ &\quad = \mathrm{E}\Big[\ln e^{\lambda(X_{t'+1}-X_{t'})} \Big| \mathcal{F}_{t'}\Big] \cdot \mathbb{1}\{X_{t'} > 0\} \\ &\quad \le \ln^+\big(\mathrm{E}[e^{\lambda(X_{t+1}-X_t)} \mid \mathcal{F}_{t'}] \cdot \mathbb{1}\{X_{t'} > 0\}\big) \quad \le \ln^+(\rho), \end{aligned}$$

which shows that $\mathrm{E}[X_{t'+1} \mid \mathcal{F}_{t'}] \cdot \mathbb{1}\{X_{t'} > 0\} \le X_{t'} \cdot \mathbb{1}\{X_{t'} > 0\}$ since $\rho < 1$. Taking the expectation on both sides and dividing by $\Pr[X_{t'} > 0 \mid \mathcal{F}_{t'}]$ shows that $\mathrm{E}[X_{t'+1} \mid X_{t'} > 0] \le \mathrm{E}[X_{t'} \mid X_{t'} > 0]$. Owing to the induction hypothesis, we have that $\mathrm{E}[X_{t'} \mid X_{t'} > 0] \le s_{\mathrm{jump}}$, concluding this case and thus proving the claim.

For bounding $\Pr[X_t > 0]$**,** we consider for some parameter T to be specified later, the phase of the last $\min\{t, T\}$ steps before time t. For $X_t > 0$ to occur during this phase, it is either necessary that X left state 0 at some time and never returned to 0 within the last T steps (event A_t), or returned within the last T steps and left at least once (event $\overline{A_t} \cap L_t$). By a union bound over T steps and using assumption 2, we have $\Pr[\overline{A_t} \cap L_t] \le p_{\mathrm{jump}} T$. Let $u(T)$ be an upper bound on $\Pr[A_t \cap S_t]$, the event of *not* returning to 0 within T steps *and* starting at state at most s_{safe}. Note that there is the error event $A_t \cap \overline{S_t}$ of probability at most p_{safe} that occurs if the process left 0 towards a state higher than s_{safe} at the last exit from 0.

Combining these cases, we have

$$\Pr[X_t > 0] \le p_{\mathrm{jump}} T + u(T) + p_{\mathrm{safe}}.$$

We now propose settings of T and bounds $u(T)$ that prove the bound claimed in the theorem. To this end, we apply Theorem 2. Assuming that we start from a value of at most s_{safe} and setting $T \coloneqq \lceil (\ln(p_{\mathrm{jump}}) - \lambda s_{\mathrm{safe}}) / \ln(\rho) \rceil$, we have

$$u(T) \le e^{\lambda s_{\mathrm{safe}}} \rho^{(\ln(p_{\mathrm{jump}}) - \lambda s_{\mathrm{safe}})/\ln(\rho)} \le p_{\mathrm{jump}}.$$

Consequently, we obtain overall

$$\begin{aligned} \Pr[X_t > 0] - p_{\mathrm{safe}} &\le p_{\mathrm{jump}} T + p_{\mathrm{jump}} \\ &\le 2 p_{\mathrm{jump}} T = 2 p_{\mathrm{jump}} \left\lceil \frac{\ln(p_{\mathrm{jump}}) - \lambda s_{\mathrm{safe}}}{\ln(\rho)} \right\rceil. \end{aligned}$$

Combining both bounds above concludes the proof of the first statement, i. e., the bound (1).

For the second statement of the theorem, to bound $\mathrm{E}[T]$, let $t \in \mathbb{N}$, and assume that $t < T$. We reconsider the bound $\mathrm{E}[\lambda(X_{t+1} - X_t) \mid \mathcal{F}_t] \le \ln(\rho)$ proven above, dropping the indicator for $X_t > 0$, which always evaluates to 1 with our current assumption on t. Since $\ln(\rho) = \ln(1 - (1 - \rho)) \le 1 - \rho$, we have $\mathrm{E}[\lambda(X_{t+1} - X_t) \mid \mathcal{F}_t] \le 1 - \rho$, implying $\mathrm{E}[X_t - X_{t+1} \mid \mathcal{F}_t] \ge \frac{1-\rho}{\lambda} \eqqcolon \delta$. Using the additive drift theorem, the bound $\mathrm{E}[T] \le \frac{X_0}{\delta} = \frac{\lambda X_0}{1-\rho}$ follows. □

Moreover, we make use of the classic multiplicative drift theorem by Doerr et al. [10] to bound certain phases in our runtime analysis.

Theorem 3 (Multiplicative drift [10], [24, Theorem 2.3.11]). *Let $n \in \mathbb{N}$, let $(X_t)_{t \in \mathbb{N}}$ be a random process over $[0..n]$, and let $T = \inf\{t \in \mathbb{N} \mid X_t = 0\}$. Moreover, assume that there is a $\delta \in \mathbb{R}_{>0}$ such that for all $t \in \mathbb{N}$, we have $\mathrm{E}[(X_t - X_{t+1}) \cdot \mathbb{1}\{t < T\} \mid X_t] \ge \delta X_t \cdot \mathbb{1}\{t < T\}$. Then, $\mathrm{E}[T] \le \frac{1}{\delta}(1 + \ln n)$.*

3 Subquadratic Runtime Bounds for Standard Bit Mutation

Our main result in this section is Theorem 4, which shows that the (μ+1) EA with population size $\mu = o((n/\log n)^{1/5})$ optimizes BinVal in $o(n^2)$ expected fitness evaluations. More generally, we prove an expected runtime bound that is polynomial in μ and only features a factor of $n \log n$. In particular, for constant values of μ, we get a bound of $O(n \log n)$, which matches the expected runtime of the (1+1) EA on BinVal [15].

To the best of our knowledge, these are the first runtime bounds of the (μ+1) EA on BinVal that are $o(n^2)$ for $\mu \ge 2$, with the previously best known results being $O(n^2)$.

Theorem 4. *Let $\mu \in \mathbb{N}_{\ge 2}$ with $\mu \le \big(n/(2048e)\big)^{1/4}$, and let $n \in \mathbb{N}_{\ge 2}$. The (μ+1) EA optimizes BinVal of problem size n in $O\big(\mu^5 n \log(n/\mu^4)\big)$ expected fitness evaluations.*

In particular, for $\mu = o((n/\log n)^{1/5})$, it optimizes BinVal in $o(n^2)$ expected fitness evaluations.

Before we prove Theorem 4, we discuss how existing analysis methods can be applied to derive rather easy upper bounds in this setting. Afterward, we overview our proof method.

Bounds from existing methods. One approach is to use the method of artificial fitness levels for the general class of linear functions (upper bound for the (1+1) EA given in Theorem 6 in [30]) and obtain an upper bound of $O(n\mu \log \mu + n^2)$ on the expected optimization time. This is done by considering the $n + 1$ fitness levels, where each fitness level considers the longest prefix of 1s in the population. Each such level is taken over in time $O(\mu \log \mu)$ by the best individual (pessimistically assuming that no improvement is found before this takeover). If this is the case, then each individual at the current fitness level has a probability of at least $1/(en)$ of reaching a higher fitness level. Taking into account that the (μ+1) EA needs to reach a higher fitness level at most n times leads to the stated upper bound. This method has been used for the (μ+1) EA on the LeadingOnes benchmark function [31].

Another approach is to use multiplicative drift analysis [10] with the objective function as a drift function and to consider the selection for reproduction of the best individual in the current population, which happens with probability at least $1/\mu$. This leads to an upper bound of $O(\mu n \log f_{\max})$, where $f_{\max}$ is the maximum function value. Since $f_{\max} = 2^n - 1$ for BinVal, the bound is $O(\mu n^2)$. This method was discussed by Doerr et al. [10, Section 3] in the context of the (1+1) EA on BinVal.

A third method, which only works immediately for the (1+1) EA, is to use additive drift analysis [20] on the number of 1s of the (only) individual maintained by the algorithm. In the most pessimistic case, there is only a single 0 left, requiring a probability of $\frac{1}{n}$ for making progress. Moreover, in this case, the expected number of 1s that is lost is at most $\frac{n-1}{n} = 1 - \frac{1}{n}$. Together, this results in a drift of at least $1/n^2$ and, in turn, in an expected runtime of $O(n^3)$. We note that Colin et al. [6] use an improved version of this argument for an algorithm called PO-EA, proving an expected runtime bound of $O(n^{3/2})$ on all monotone functions. However, it is not clear how this improved version translates directly to the (μ+1) EA. Moreover, this bound is still larger than $O(n \log n)$.

Overall, all methods only lead to a bound being no better than $O(n^2)$, even for constant μ.

Our proof strategy. Our analysis follows the ideas of Droste et al. [15], who study the (1+1) EA on BinVal via multiplicative drift analysis[1] on the number of 1s of the (only) individual in the population, leading to an $O(n \log n)$ bound. When calculating the drift directly based on the entire bit string, this only results in a drift bound of $\Omega(1/n^2)$, as discussed above. Droste et al. [15] circumvent this problem by considering the two halves (called *blocks*) of the bit string separately. By a careful analysis of how many bits are flipped per block, they show that this leads to a positive multiplicative drift per block, leading to the $O(n \log n)$ bound.

We adapt this method to populations. We consider the number of 1s in a block of a best individual[2] in the population, called x. If x is selected as parent for mutation and improved, we conduct the same analysis as Droste et al. [15]. However, if x is replaced by an individual created from another individual in the population, it is harder to bound how many 1s we may lose in the potential in this block. To counteract this, we keep track of the maximum discrepancy of 1s among individuals per block. Since our computations rely on drift, it is sufficient to have a bound on the *expected* discrepancy per block. We formalize this result in Theorem 6. It shows for a suitable block length, after a certain number of iterations, that the discrepancy in the population is in fact only a constant.

With a bound on the discrepancy in the population, we analyze the progress per block in the same fashion as Droste et al. [15], formalized in Lemma 5. We note though that we need to choose far more blocks than two. In fact, we chose $\Theta(\mu^4)$ blocks, which in turn results immediately in the constraint $\mu = O(n^{1/4})$. Summing over the waiting time per block, we derive Theorem 4.

Definitions for our proof strategy. We consider $k \in \mathbb{N}_{\geq 1}$ phases, where phase $i \in [k]$ only considers the time until all bits at positions $[\lfloor (i-1)\frac{n}{k} \rfloor + 1..\lfloor i\frac{n}{k} \rfloor]$ of the best individual in the population are 1s. To this end, we call for all $i \in [k]$ the interval $[\lfloor (i-1)\frac{n}{k} \rfloor + 1..\lfloor i\frac{n}{k} \rfloor]$ *block i*. Note that by the definition of BinVal, for all $i \in [n]$, we have that $\text{BinVal}(0^{i-1}10^{n-i}) > \text{BinVal}(0^i 1^{n-i})$. Thus, any single bit in a block at position $i \in [k]$ has a higher BinVal weight than all other bits combined in blocks at larger indices, motivating to solve the blocks in increasing order.

Formally, we define the runtime of each of the k phases inductively. Let $S^{(0)} \coloneqq 0$, and for all $i \in [k]$, let $S^{(i)} \coloneqq \inf\{t \in \mathbb{N} \mid t \geq S^{(i-1)} \wedge \exists x \in P_t \forall j \in [\lfloor i\frac{n}{k} \rfloor] \colon x_j = 1\}$. In addition, let $T^{(i)} \coloneqq S^{(i)} - S^{(i-1)}$. Lemma 5 bounds the runtime of each phase.

Lemma 5. *Let $n \in \mathbb{N}_{\geq 2}$, $\mu \in \mathbb{N}_{\geq 2}$ with $\mu \leq (n/(2048e))^{1/4}$, $k \in \mathbb{N}$ with $k \geq 2048e\mu^4$, and $i \in [k]$. Consider the (μ+1) EA optimizing BinVal with problem size n. Furthermore, let S' be the first iteration in that all blocks up to $i-1$ are optimized, and let $\widetilde{S}$ denote the first iteration after S' in which the maximum Hamming distance in block i is 0. Then for the expected runtime of phase i defined above, accounting only for iterations starting from iteration $\widetilde{S}$, we have $\mathrm{E}[T^{(i)}] \leq 4e\mu n(1 + \ln\lceil \frac{n}{k} \rceil)$.*

Bounding the population discrepancy. To prove Lemma 5, we develop the above-mentioned analysis of population dynamics within individual blocks. Theorem 6 bounds the expected Hamming distance of the population within a block. Its proof relies on Theorem 1 and on showing that it is likely to create an offspring that is identical to the best individual in the block that is considered. For standard bit mutation, it is likely to create a copy of the entire parent. For mutation operators that do not allow to create copies, we prove a similar statement in Theorem 8, which we discuss in more detail in Section 4.

Theorem 6. *For $\mu \in \mathbb{N}_{\geq 2}$, consider the ($\mu$+1) EA on the BinVal function of size $n \in \mathbb{N}_{\geq 2}$. Let $\varepsilon \leq 1/(2048e\mu^4)$ and consider any block of bits $[a..a + \lfloor \varepsilon n \rfloor]$ for $a \in [n - \lfloor \varepsilon n \rfloor]$. For the random population P_t at time $t \in \mathbb{N}$, let $H_t \coloneqq \max_{x,y \in P_t} \sum_{i=a}^{a+\lfloor \varepsilon n \rfloor} |x_i - y_i|$ be the maximum Hamming distance in the block.*

If all individuals of P_0 are identical in the block, then $\mathrm{E}[H_t] \leq 1/4$ for all $t \in \mathbb{N}$.

Proof. At the core of this proof, we use Theorem 1. To bound the maximum Hamming distance in the block b that we consider, we introduce a potential function based on the following quantities. For all $t \in \mathbb{N}$, let P_t be the population at time t. We define:

- $S_t^{(b)}$, the number of individuals that differ from the best individual in P_t in block b,
- $H_t^{(b)}$, the maximum Hamming distance in block b at time t.

Then the potential is defined as $\Phi_t^{(b)} \coloneqq S_t^{(b)} + H_t^{(b)}$. We omit the index b if there is no source of confusion.

In the following, we set $X_t = \Phi_t^{(b)}$ and verify the conditions of Theorem 1. That is, we we have to bound $p_{\text{jump}}, \rho, s_{\text{jump}}$ and to choose parameters for s_{safe} and corresponding p_{safe}.

The first and most comprehensive task is to verify Assumption 1, which informally says that X_t drifts towards 0, in the strong sense of the moment-generating function.

We define $p_{i,j} = \Pr[X_{t+1} = j \mid X_t = i]$ (formally, these transition probabilities are only defined for a given population P_t, which we ignore in the following by providing worst-case bounds holding for all P_t such that $X_t = i$). Moreover, we define $p_{i,\leq j} \coloneqq \sum_{k \leq j} p_{i,k}$.

We have

- For $i \geq 1$, $p_{i,\leq i-1} \geq (1 - 1/n)^n/\mu \geq e^*/\mu$, where $e^* = e^{-1}(1 - 1/n)$, since it is sufficient to choose the best individual and to produce a clone to decrease the component S_t in the potential function (removing another individual).
- For $k > \mu$, $p_{i,i+k} \leq \binom{\varepsilon n}{k-\mu} \left(\frac{1}{n}\right)^{k-\mu} \leq \frac{\varepsilon^{k-\mu}}{(k-\mu)!}$ since the S_t-component can increase by at most $\mu - 1$ in an iteration and at least $k - \mu + 1$ bits in the block have to flip to increase the H_t-component by $k - \mu + 1$. To keep the bounds simple, we work with a pessimistic increase of S_t of μ instead of $\mu - 1$, so only at least $k - \mu$ bits have to flip.
- For $1 \leq k \leq \mu$, $p_{i,i+k} \leq \binom{\varepsilon n}{1}\frac{1}{n} = \varepsilon$ since at least one bit in the block has to flip to change either component of $X_t = \Phi_t^{(b)}$.
- We pessimistically bound $p_{i,j} \geq 0$ for $j < i - 1$.

We now proceed by first bounding *the (transformed) drift away from the target* $\sum_{k=1}^{\mu+n-i} e^{\lambda k} p_{i,i+k}$ uniformly for i and then including *the drift towards the target* $e^{\lambda 0} p_{i,i} + \sum_{k=1}^{i} e^{-\lambda k} p_{i,i-k}$ to finally bound ρ. This split will make it easier to adjust the analysis to slightly modified settings like in Section 4.

[1]This term was not coined back then, but the arguments fit this framework.
[2]We break ties arbitrarily. Since BinVal is bijective, ties for the best individual only result in duplicates, not changing the bit string.

Transformed drift away from the target. Using the above bounds for $p_{i,i+k}$, we have for any i that

$$\begin{aligned}\sum_{k=1}^{\mu+n-i} e^{\lambda k} p_{i,i+k} &\le \sum_{k=1}^{\min\{\mu,\mu+n-i\}} e^{\lambda k}\varepsilon + \sum_{k=\mu+1}^{\mu+n-i} e^{\lambda k}\frac{\varepsilon^{k-\mu}}{(k-\mu)!}\\ &\le \varepsilon\frac{e^{\lambda(\mu+1)}-e^{\lambda}}{e^{\lambda}-1} + e^{\lambda\mu}\sum_{k=1}^{n-i} e^{\lambda k}\frac{\varepsilon^k}{k!},\end{aligned}$$

where the last equality stems from a geometric sum and an index transformation. Letting the final sum go to infinity and identifying the exponential function, we obtain the bound for any i that

$$\sum_{k=1}^{\mu+n-i} e^{\lambda k} p_{i,i+k} \le \varepsilon\frac{e^{\lambda(\mu+1)}-e^{\lambda}}{e^{\lambda}-1} + e^{\lambda\mu}(e^{e^{\lambda}\varepsilon}-1) =: D(\lambda,\varepsilon).$$

Transformed drift towards the target. We have

$$\begin{aligned}e^{\lambda 0}p_{i,i} + \sum_{k=1}^{i} e^{-\lambda k}p_{i,i-k} &\le \left(1-\sum_{k=1}^{i}p_{i,i-k}\right) + e^{-\lambda}\sum_{k=1}^{i}p_{i,i-k}\\ &\le 1-(1-e^{-\lambda})\frac{e^*}{\mu} =: G(\lambda),\end{aligned}$$

where the last inequality used our bound on $p_{i,\le i-1}$ from above. Also this bound holds for any current potential i.

For Theorem 1 to be applied, we have to bound the expression $\rho(\lambda,\varepsilon) := D(\lambda,\varepsilon)+G(\lambda)$ by some value less than 1. Choosing $\lambda = \frac{1}{\mu}$, assuming $\varepsilon \le \frac{1}{2}$ and noting that $\mu \ge 2$, we have

$$\begin{aligned}\rho(\lambda,\varepsilon) &= D(\lambda,\varepsilon)+G(\lambda)\\ &= \varepsilon\frac{e^{1+1/\mu}-e^{1/\mu}}{e^{1/\mu}-1} + e^1(e^{e^{1/\mu}\varepsilon}-1) + 1 - (1-e^{-1/\mu})\frac{e^*}{\mu}\\ &\le \varepsilon\frac{e^{3/2}-e^{1/2}}{1/\mu} + e\left(e^{e^{1/2}\varepsilon}-1\right) + 1 - \left(1-\left(1-\frac{1}{2\mu}\right)\right)\frac{e^*}{\mu},\end{aligned}$$

where we used $e^x \ge 1+x$ and $e^{-x} \le 1-x/2$ for $x \in [0,1]$. Further noting that $e^{e^{1/2}\varepsilon} \le 1+3\varepsilon$ for $\varepsilon \le 1/2$, $e^* \ge e^{-1}/2$ for $n \ge 2$ and again using $\mu \ge 2$, the bound is no larger than

$$(e^{3/2}-e^{1/2})\varepsilon\mu + 3e\varepsilon + 1 - \frac{e^*}{2\mu^2} \le \underbrace{7\mu\varepsilon}_{\ge D(\lambda,\varepsilon)} + \underbrace{1-\frac{e^{-1}}{4\mu^2}}_{\ge G(\lambda)}.$$

By $\varepsilon \le \frac{1}{56e\mu^4}$, we have $D(\lambda,\varepsilon) \le \frac{1}{8e\mu^3}$ and obtain $\rho(\lambda,\varepsilon) = D(\lambda,\varepsilon)+G(\lambda) \le \frac{1}{8e\mu^3} + 1 - \frac{1}{4e\mu^2} \le 1 - \frac{1}{8e\mu^2}$. Hence, we verified Assumption 1 with $\rho = 1-\frac{1}{8e\mu^2}$ for the given choices of ε and λ.

To verify Assumption 2, we analyze p_{jump}, the probability of increasing the potential from state 0. Since it is necessary to flip at least one bit in block b to leave state 0, we have $p_{\text{jump}} \le \frac{\varepsilon n}{n} = \varepsilon$.

To verify Assumption 3, in particular to set s_{safe}, we recall the above estimates of $p_{i,i+k}$. In particular, the probability $p_{0,\ge\mu+k}$ of leaving 0 towards state at least $\mu+k$, where $k \ge 1$, is at most $\sum_{j=k}^{\infty} p_{0,\mu+j} \le \sum_{j=k}^{\infty}\frac{\varepsilon^j}{j!} \le 2\frac{\varepsilon^k}{k!}$. Since $\varepsilon \le 1/2$, we have for $s_{\text{safe}} := 4\mu$ that $p_{i,\ge s_{\text{safe}}} \le 2^{1-3\mu}$, so $p_{\text{safe}} = 2^{1-3\mu}$, which will be sufficient for our purposes.

To verify Assumption 4, i. e., to bound s_{jump}, we will not use the above bounds on transition probabilities out of 0 since they provide upper bounds only. However, s_{jump} lives in a conditional state, so we implicitly divide by the probability of leaving state 0, which only works directly when lower bounds on transition probabilities are available. The following reasoning is a more direct approach.

We claim $s_{\text{jump}} \le 2\mu$. To show this, we prove that when the process leaves 0, then an expected number of at most 2 bits flip in the considered block. Pessimistically assuming that the component $S_t^{(b)}$ of X_t corresponding to the number of different individuals increases by $\mu-1$ in such a step, this gives us $s_{\text{jump}} \le \mu-1+2 = \mu+1 \le 2\mu$.

We consider the block b of length εn and potential 0, i. e., the population is identical in the block. We condition on that the potential increases, i. e., at least one bit of block b flips. We distinguish between two cases (heavily relying on that we optimize BinVal):

Case 1 A zero-bit flips left of block b. Then an improving individual is created and any change to block b is accepted. The expected number of flipping bits in block b, conditional on changing b, is $\varepsilon/\bigl(1-(1-1/n)^{\varepsilon n}\bigr) \le 2$, where the numerator is the unconditional number of flipping bits in b and the denominator the probability of flipping at least one bit of b.

Case 2 No zero-bit flips left of the current block b. In order for block b to be changed and the offspring to be accepted into the next population, at least one-zero bit in b must flip (if there are no zero-bits, the potential cannot change). Let Z be the indices of the zero-bits in block b. We partition the random mutation, which changes the block, into $|Z|$ many groups $M_1,\dots,M_{|Z|}$ according to the left-most flipping zero-bit. We claim that the expected number of flipping bits in each M_j, where $j \in Z$, is $O(1)$. This holds since if zero-bit j is the left-most flipping zero-bit, then the mutation will be accepted no matter how the bits right of j flip. There are at most $\varepsilon n - 1$ such bits, so the expected number of flipping bits in addition to j in block b is less than ε, so again the expected total number of flipping bits in b is at most 2.

Altogether, $s_{\text{jump}} \le \mu-1+2 \le 2\mu$ as suggested. We have now dealt with the first four assumptions of Theorem 1. By the prerequisites of the present theorem, Assumption 5 is also satisfied. Hence, we are ready to prove the statement on $\mathrm{E}[X_t]$.

Application of Theorem 1.

We abbreviate $B := 2p_{\text{jump}}\lceil(\ln(p_{\text{jump}})-\lambda s_{\text{safe}})/\ln(\rho)\rceil$ and obtain, plugging in the parameters ρ, p_{jump} and s_{safe},

$$\begin{aligned}B &\le 2\varepsilon\left\lceil\frac{\ln(\varepsilon)-\lambda(4\mu)}{\ln(1-1/(8e\mu^2))}\right\rceil \le 2\varepsilon\lceil -\ln(\varepsilon)+\lambda(4\mu)\rceil 8e\mu^2\\ &\le 16e\varepsilon\mu^2(5+\ln(1/\varepsilon)),\end{aligned}$$

where we recall $\lambda = 1/\mu$. Choosing $\varepsilon = 1/(2048e\mu^4)$, which satisfies $\varepsilon \le 1/(56e\mu^4)$ as required above, we have

$$B \le \frac{1}{128\mu^2}(5+4\ln\mu+\ln(2048e)) \le \frac{0.07}{\mu},$$

where we noticed that the right-hand side was monotonically decreasing for $\mu \ge 2$. Hence, by the final statement of Theorem 1,

$$\begin{aligned}\mathrm{E}[X_t] &\le (p_{\text{safe}}+B)s_{\text{jump}} \le (2^{1-3\mu}+0.07/\mu)\cdot(2\mu)\\ &\le 2^{2-3\mu}\mu + 0.14 \le 2^{-4}+0.14 \le 1/4.\end{aligned}$$

Clearly, $X_t = \Phi_t^{(b)}$ is a bound on the maximum Hamming distance in the block, so the theorem follows. □

We remark that the above analysis can be simplified in the case $\mu = 2$, where it is sufficient to analyze the drift on the Hamming distance of the two individuals of the population. However, since we will prove a bound of $O(n \log n)$ for all constant μ anyway, this simplified drift analysis will not give any improved runtime bounds in O-notation.

Bounding the runtime of each phase. Using Theorem 6, we prove Lemma 5. We do so by considering the number of 0s in the current block in the best individual x in the population, following the ideas of Droste et al. [15]. We then condition on the event to make progress, that is, to replace x. Furthermore, we make a case distinction with respect to whether x is chosen as a parent for the mutation or not. If x is chosen, then we know that we lose at least one 0, and the number of remaining bits that are flipped follows a binomial distribution with success probability $\frac{1}{n}$ and the block size as number of trials. Since a block is small enough, this leads to positive drift in this case.

For the other case, if x is not improved, the analysis is similar but needs to account for the expected Hamming distance of the parent and x. To this end, we apply Theorem 6 and also get positive drift, due to the small block size.

Together with the probability of making progress at all, that is, flipping at least one 0 in the block, we get multiplicative drift with a factor of at least $1/(4e\mu n)$, concluding the proof.

Proof of Lemma 5. We aim at showing that the expected number of 0s of the best individual in the population decreases and exhibits multiplicative drift. To this end, for all $x \in \{0,1\}^n$, let $|x|_{0,i} := |\{j \in [\lfloor (i-1)\frac{n}{k} \rfloor + 1..\lfloor i\frac{n}{k} \rfloor] \mid x_j = 0\}|$ denote the number of 0s of x in block i. Moreover, for all $t \in \mathbb{N}$, let $X_t := \min\{|y|_{0,i} \mid y \in \arg\max_{x \in P_{t+S^{(i-1)}}} \textsc{BinVal}(x)\}$.

Let $t \in \mathbb{N}$ such that $X_t > 0$. We aim at bounding $\mathrm{E}[X_t - X_{t+1} \mid X_t]$ from below by a positive number. Let A denote the event that $X_t \neq X_{t+1}$ (and we show below that it has positive probability), and let B denote the event that we choose a best individual $x \in P_t$ as parent for the mutation in iteration t. Thus,

$$\begin{aligned}\mathrm{E}[X_t - X_{t+1} \mid X_t] = \big(&\mathrm{E}[(X_t - X_{t+1}) \cdot \mathbb{1}\{B\} \mid X_t; A] \\ &+ \mathrm{E}[(X_t - X_{t+1}) \cdot \mathbb{1}\{\overline{B}\} \mid X_t; A]\big) \cdot \Pr[A \mid X_t].\end{aligned}$$

We bound $\Pr[A \mid X_t]$ and the two terms in the sum from below.

Bounding $\Pr[A \mid X_t]$. In order for A to occur, the best individual x needs to be replaced by a strictly better individual, as BinVal is bijective. One way of doing so is to pick x as parent for the mutation and create an offspring that flips exactly one 0 in x in block i and does not flip any other bit. This results in

$$\Pr[A \mid X_t] \geq \frac{1}{\mu}\left(1 - \frac{1}{n}\right)^{n-1} \frac{X_t}{n} \geq \frac{X_t}{e\mu n}.$$

Bounding the expected difference intersected with B. Let $y \in \{0,1\}^n$ denote the offspring generated by mutating x, recalling that due to the condition A, it is the new best individual. Since we consider BinVal, the leftmost position in that x and y differ is in block i, and x has a 0 in that position, and y has a 1. Each of the following positions in block i flips with probability $\frac{1}{n}$, resulting in a random number of 0s turning into 1s, and vice versa. We call the resulting difference in 0s of these random changes D, such that we have $\mathrm{E}[|y|_{0,i} \cdot \mathbb{1}\{B\} \mid X_t; A] = \mathrm{E}[(|x|_{0,i} - 1 + D) \cdot \mathbb{1}\{B\} \mid X_t; A]$, recalling that $|x|_{0,i} = X_t$. We note that D is stochastically dominated by $D' \sim \mathrm{Bin}(\lceil \frac{n}{k} \rceil - 1, \frac{1}{n})$, as the length of each block is at most $\lceil \frac{n}{k} \rceil$ and we have a difference between each position of x and y only with independent probability $\frac{1}{n}$. Thus, and since $k \geq 2$, we get

$$\begin{aligned}\mathrm{E}[(X_t - X_{t+1}) \cdot \mathbb{1}\{B\} \mid X_t; A] &\geq \mathrm{E}\left[\left(1 - \frac{1}{k}\right) \cdot \mathbb{1}\{B\} \,\middle|\, X_t; A\right] \\ &\geq \frac{1}{2} \cdot \mathrm{E}[\mathbb{1}\{B\} \mid X_t; A].\end{aligned}$$

Bounding the expected difference intersected with $\overline{B}$. Let $x' \in P_t$ denote the individual chosen for mutation, which is, by $\overline{B}$, not a best individual. Furthermore, let $y \in \{0,1\}^n$ denote the offspring of this mutation, which, due to the condition A, is a new best individual. Given this notation, we decompose

$$\begin{aligned}\mathrm{E}[(X_t - X_{t+1}) \cdot \mathbb{1}\{\overline{B}\} \mid X_t; A] &= \mathrm{E}[(|x|_{0,i} - |y|_{0,i}) \cdot \mathbb{1}\{\overline{B}\} \mid X_t; A] \\ &= \mathrm{E}[(|x|_{0,i} - |x'|_{0,i}) \cdot \mathbb{1}\{\overline{B}\} \mid X_t; A] \\ &\quad + \mathrm{E}[(|x'|_{0,i} - |y|_{0,i}) \cdot \mathbb{1}\{\overline{B}\} \mid X_t; A],\end{aligned}$$

where we bound the two terms separately in the following.

For $\mathrm{E}[(|x|_{0,i} - |x'|_{0,i}) \cdot \mathbb{1}\{\overline{B}\} \mid X_t; A]$, since $x, x' \in P_t$, we bound $|x|_{0,i} - |x'|_{0,i} \geq -d_{\mathrm{H}}(x,x') \geq -\max_{z,z' \in P_t} d_{\mathrm{H}}(z,z')$. Furthermore, note that A is independent of the maximum Hamming distance in P_t. Hence, bounding $\mathbb{1}\{\overline{B}\} \leq 1$, we get

$$\mathrm{E}[(|x|_{0,i} - |x'|_{0,i}) \cdot \mathbb{1}\{\overline{B}\} \mid X_t; A] \geq \mathrm{E}[-\max_{z,z' \in P_t} d_{\mathrm{H}}(z,z') \mid X_t].$$

By Theorem 6 with $\varepsilon := \frac{1}{k}$, it follows that

$$\mathrm{E}[(|x|_{0,i} - |x'|_{0,i}) \cdot \mathbb{1}\{\overline{B}\} \mid X_t; A] \geq -\frac{1}{4}.$$

For $\mathrm{E}[(|x'|_{0,i} - |y|_{0,i}) \cdot \mathbb{1}\{\overline{B}\} \mid X_t; A]$, since x is a best individual and since $\textsc{BinVal}(x') < \textsc{BinVal}(x)$, it follows by condition A that x' and y differ in at least two positions, both of which are 0s in x' and 1s in y. By an analogous argument to the previous case, now only considering all but two positions in block i, we get

$$\begin{aligned}\mathrm{E}[(|x'|_{0,i} - |y|_{0,i}) \cdot \mathbb{1}\{\overline{B}\} \mid X_t; A] &\geq \mathrm{E}\left[\left(2 - \frac{1}{k}\right) \cdot \mathbb{1}\{\overline{B}\} \,\middle|\, X_t; A\right] \\ &\geq \mathrm{E}[\mathbb{1}\{\overline{B}\} \mid X_t; A].\end{aligned}$$

Taken together, we bound

$$\mathrm{E}[(X_t - X_{t+1}) \cdot \mathbb{1}\{\overline{B}\} \mid X_t; A] \geq \mathrm{E}[\mathbb{1}\{\overline{B}\} \mid X_t; A] - \frac{1}{4}.$$

Combining all bounds. Using the bounds from above, we have

$$\begin{aligned}&\mathrm{E}[(X_t - X_{t+1}) \cdot \mathbb{1}\{B\} \mid X_t; A] + \mathrm{E}[(X_t - X_{t+1}) \cdot \mathbb{1}\{\overline{B}\} \mid X_t; A] \\ &\geq \frac{1}{2} \cdot \mathrm{E}[\mathbb{1}\{B\} \mid X_t; A] + \mathrm{E}[\mathbb{1}\{\overline{B}\} \mid X_t; A] - \frac{1}{4} \\ &\geq \frac{1}{2} - \frac{1}{4} = \frac{1}{4}.\end{aligned}$$

Multiplying this with our bound for $\Pr[A \mid X_t]$ yields

$$\mathrm{E}[X_t - X_{t+1} \mid X_t] \geq \frac{X_t}{4e\mu n}.$$

The proof concludes by applying the multiplicative drift theorem to X, noting that $T^{(i)} = \inf\{t \in \mathbb{N} \mid X_t < 1\}$ and that $X_0 \leq \lceil \frac{n}{k} \rceil$. □

Proving our main result.

Proof of Theorem 4. Let $k = \lceil 2048e\mu^4 \rceil$ be the number of phases we consider. We aim to apply Lemma 5 to all k phases, also accounting for the respective waiting times until the maximum Hamming distance in each block is 0 for the first time. In particular, this implies that Lemma 5 is applicable to all phases.

Bounding the times until each block as a maximum Hamming distance of 0. Let $i \in [k]$, consider block i, and recall the notation from Lemma 5. Let $S^\star = \widetilde{S} - S'$. According to the proof of Theorem 6 and the values defined therein, that is, $\lambda = \mu^{-1}$, $\varepsilon \le (2048e\mu^4)^{-1}$, and $\rho = 1 - (8e\mu^2)^{-1}$, Theorem 1 is applicable, whose second statement bounds $\mathrm{E}[S^\star] \le \mu^{-1}\varepsilon n \cdot 8e\mu^2 \le n/\mu^3$. Since we have this waiting time for each of the k blocks, the total waiting time across all blocks adds up to at most $kn/\mu^3 = O(\mu n)$.

Concluding. By construction of the phases and the previous discussion, we have for the runtime $T \coloneqq \inf\{t \in \mathbb{N} \mid \exists x \in P_t : |x|_1 = n\}$ that $T \le O(\mu n) + \sum_{i\in[k]} T^{(i)}$. By Lemma 5, it follows for all $i \in [k]$ that $\mathrm{E}[T^{(i)}] \le 4e\mu n(1 + \ln\lceil \frac{n}{k} \rceil)$. Substituting this bound into the estimate of T, simplifying, and adding μ fitness evaluations for the initialization of the algorithm concludes the proof. □

4 Subquadratic Runtime Bounds for 1-Bit and Plus Mutation

Our main result in this section is Theorem 7, which proves expected runtime bounds for the (μ+1) EA with 1-bit and plus mutation similar to those in Theorem 4. For certain sublinear regimes of μ, we get $o(n^2)$ runtime bounds. The main difference in our analysis is that for the proof of Theorem 7, we can no longer rely on Theorem 6, which heavily relied on the standard bit mutation's ability to create copies. Now, as we detail below, we are able to prove a similar result (Theorem 8) if we ignore a certain amount of bits. As a result, Theorem 7 mentions an approximation result, which results for constant μ again in an expected runtime of $O(n \log n)$.

In addition, we show how our approximation result can be turned into an optimization runtime result. However, this adds, besides others, a factor of $\sqrt{n}$ to the runtime. Hence, for constant μ, this bound simplifies only to $O(n^{3/2} \log n)$.

Theorem 7. *Let $n \in \mathbb{N}_{\ge 1}$, let $\mu \in \mathbb{N}_{\ge 1}$ and consider the (μ+1) EA with 1-bit or plus mutation operator on BinVal.*

(1) *Assume $\mu \le \big(n/(8192e)\big)^{1/5}$. Then the ($\mu$+1) EA optimizes the first $n - \lceil n/\mu \rceil$ bits of BinVal in $O\big(\mu^6 n \log(n/\mu^5)\big)$ expected fitness evaluations. For $\mu = o((n/\log n)^{1/6})$, this corresponds to $o(n^2)$ expected fitness evaluations.*

(2) *Assume $\mu \le \big(n/(8192e)^2\big)^{1/4}$. Then the ($\mu$+1) EA finds the global optimum of BinVal in $O\big(\mu^3 n^{3/2} \log(n/\mu^2)\big)$ expected fitness evaluations. For $\mu = o((n/\log n)^{1/6})$, this corresponds to $o(n^2)$ expected fitness evaluations.*

Our proof strategy. Our strategy follows mostly the one we applied in Section 3. However, the drift analysis in Theorem 6 of the maximum Hamming distance in the population relied crucially on the ability of standard bit mutation to create clones of the parent. For the 1-bit and plus mutation operator, such clones are impossible. However, it is actually very likely that the offspring and parent coincide in the currently considered block of εn bits. Considering a mutation that chooses as parent an individual with best BinVal value when only considering the bits of the considered block as well as all bits left to it (of higher value), a mutation that only changes bits to the right of the block (i. e., of lower weight) is accepted by the (μ+1) EA unless the whole population has the same configuration in the block already. Hence, our upcoming analyses consider situations where it is still relatively likely to flip bits to the right of the current block (and no other bits). We stop our analysis at the point where only few bits at the tail of the bit string are left and the probability of such mutations has become too small. This corresponds to solutions that have a long prefix of leading ones and provide a good approximation of the optimum solution in genospace, i. e., in Hamming distance. With respect to the optimum BinVal-value, such a solution provides an approximation of exponentially small error.

To turn our approximation result into an optimization result, we apply Theorem 8 for bounding the population discrepancy for all but $\Theta(\mu^2\sqrt{n})$ bits (over different blocks). We then bound the runtime for optimizing the remaining $\Theta(\mu^2\sqrt{n})$ bits via the method of artificial fitness levels, as explained at the beginning of Section 3.

Bounding the population discrepancy.

Besides requiring some slack s of bit positions not considered, the statement of Theorem 8 itself and its proof are similar to Theorem 6.

Theorem 8. *Let $n \in \mathbb{N}_2$ and $s \in \mathbb{N}_{\ge 1}$ with $s < n$. For $\mu \in \mathbb{N}_{\ge 2}$, consider the ($\mu$+1) EA with either 1-bit or plus mutation on the BinVal function of size n. Let $\varepsilon \le \frac{(s/n)}{4096e\mu^4}$ and consider any block of bits $[a..a + \lfloor \varepsilon n \rfloor]$ for $a \in [n - s - \lfloor \varepsilon n \rfloor]$. For the random population P_t at time $t \in \mathbb{N}$, let $H_t \coloneqq \max_{x,y\in P_t} \sum_{i=a}^{a+\lfloor \varepsilon n \rfloor} |x_i - y_i|$ be the maximum Hamming distance in the block.*

If all individuals of P_0 are identical in the block, then $\mathrm{E}[H_t] \le 1/4$ for all $t \in \mathbb{N}$.

Proof. Let $g^* \coloneqq 1 - (1 - 1/n)^n \approx 1 - e^{-1}$ be the probability that standard bit mutation flips at least one bit. Revisiting the proof of Theorem 6 and using the same potential function, we have the following changes for the transition probabilities. For the sake of simplicity, all following bounds hold for both operators, although there actually are small differences between the two operators:

- For $i \ge 1$, $p_{i,\le i-1} \ge \frac{1}{g^*\mu}\left(1 - \frac{1}{n}\right)^{n-s} \frac{s}{n} \ge \frac{s}{g^* e\mu n} \ge \frac{s}{e\mu n}$, since it is sufficient to choose the best individual and to flip at positions $[s + 1..n]$ to produce an accepted offspring that is identical in the block. Note that this consideration relies on the structure of the function BinVal.
- For $k > \mu$, $p_{i,i+k} \le \frac{1}{g^*}\binom{\varepsilon n}{k-\mu}\left(\frac{1}{n}\right)^{k-\mu} \le \frac{\varepsilon^{k-\mu}}{g^*(k-\mu)!}$ in the same way as for the standard bit mutation, conditioned on the event that at least one bit flips. For the 1-bit flip operator, this even ignores that only one bit flips.
- For $1 \le k \le \mu$, $p_{i,i+k} \le \frac{1}{g^*}\binom{\varepsilon n}{1}\frac{1}{n} = \frac{\varepsilon}{g^*}$ in the same way as for standard bit mutation, conditioned on that at least one bit flips.
- We pessimistically bound $p_{i,j} \ge 0$ for $j < i - 1$.

In the following, we work with the bounds $1 \le 1/g^* \le 2$. This means that all bounds $p_{i,i+k}$ for $k \ge 1$ are at most by a factor of 2 larger than in the proof of Theorem 6. Crucial changes have only happened for $p_{i,\le i-1}$.

In the rest of the proof, we work with

$$G(\lambda) = 1 - (1 - e^{-\lambda})\frac{s}{e\mu n},$$

(where the fraction replaces $\frac{e^*}{\mu}$ from the proof of Theorem 6) and

$$D(\lambda, \varepsilon) \coloneqq 2\left(\varepsilon \frac{e^{\lambda(\mu+1)} - e^{\lambda}}{e^{\lambda} - 1} + e^{\lambda\mu}(e^{e^{\lambda}\varepsilon} - 1)\right),$$

i. e., the previous value multiplied by 2. We choose $\lambda = 1/\mu$ to obtain

$$\rho(\lambda, \varepsilon) \le 14\mu\varepsilon + 1 - \frac{s}{2e\mu^2 n}.$$

Choosing $\varepsilon \le \frac{s}{56e\mu^4 n}$ (i. e., by an asymptotic factor of s/n smaller than before), we have $D(\lambda, \varepsilon) \le \frac{s}{4e\mu^3 n}$ and obtain $\rho(\lambda, \varepsilon) \le \frac{s}{4e\mu^3 n} + 1 - \frac{s}{2e\mu^2 n} \le 1 - \frac{s}{4e\mu^2 n}$, since $\mu \ge 2$.

For the other parameters of Theorem 1, we incorporate the above-mentioned factor of 2, giving $p_{\text{jump}} = 2\varepsilon$, $s_{\text{jump}} \le 4\mu$, $p_{\text{safe}} \le 2^{2-3\mu}$, and $s_{\text{safe}} = 5\mu$. Analogous calculations to the proof of Theorem 6 yield for $B \coloneqq 2p_{\text{jump}}\lceil(\ln(p_{\text{jump}}) - \lambda s_{\text{safe}})/\ln\rho\rceil$, recalling $\lambda = 1/\mu$, that

$$\begin{aligned} B &\le 4\varepsilon\left\lceil\frac{\ln(2\varepsilon) - \lambda(5\mu)}{\ln(1 - s/(4e\mu^2 n))}\right\rceil \le 4\varepsilon\lceil -\ln(2\varepsilon) + \lambda(4\mu)\rceil 5e\mu^2 n/s \\ &\le 16e\varepsilon\mu^2(n/s)\big(6 + \ln(1/(2\varepsilon))\big). \end{aligned}$$

Choosing $\varepsilon = (s/n)/(4096e\mu^4)$, satisfying the requirement $\varepsilon \le s/(56e\mu^4 n)$ above, we have

$$\begin{aligned} B &\le \frac{1}{256\ln(n/s)\mu^2}(6 + 4\ln(\mu) + \ln(8192e) + \ln(n/s)) \\ &\le \frac{(6 + 4 + \ln(8192e) + 1)\ln\mu}{256\mu^2} \le \frac{0.015}{\mu}, \end{aligned}$$

where we notice that the right-hand side was monotonically decreasing for $\mu \ge 2$. Hence, by the final statement of Theorem 1,

$$\mathrm{E}[X_t] \le (p_{\text{safe}} + B)s_{\text{jump}} \le (2^{2-4\mu} + 0.015/\mu) \cdot (4\mu) \le 1/4. \quad \square$$

Proof of our main result. Given Theorem 8, we prove Theorem 7. For the approximation result, we apply Theorem 8 with $s \coloneqq \lfloor\frac{n}{2\mu}\rfloor$ and $k \coloneqq \lceil 80192e\mu^5\rceil$ and proceed otherwise as in the proof of Lemma 5. For the optimization result, we apply Theorem 8 with $s \coloneqq \lfloor\frac{\mu^2}{2}\sqrt{n}\rfloor$ and $k \coloneqq \lceil 8192e\mu^2\sqrt{n}\rceil$ and conduct a similar analysis to before. Afterward, we argue via artificial fitness levels, showing that it takes at most $O(\mu n\log\mu)$ iterations to increase one level. The result follows by noting that we need to increase order s times.

Proof of Theorem 7. For both statements, we follow a similar analysis as in the proof of Lemma 5, relying now Theorem 8. In particular, we make use of the same notation from Section 3.

We handle both cases of Theorem 7 separately.

• **Case 1—Approximation.** Let $s \coloneqq \lfloor\frac{n}{2\mu}\rfloor$ and $k \coloneqq \lceil 8192e\mu^5\rceil$. We consider k blocks, stopping our analysis once the first $k - 1$ blocks are optimized. Note that this implies that at least $\lfloor(k-1)\frac{n}{k}\rfloor \ge n - \lfloor\frac{n}{k}\rfloor \ge n - \lceil\frac{n}{\mu}\rceil$ bits are optimal in a best individual of the $(\mu{+}1)$ EA.

As our proof follows the one of Lemma 5 very closely, we use the following notation from that proof. For all $i \in [k]$ and all $x \in \{0,1\}^n$, let $|x|_{0,i} \coloneqq |\{j \in [\lfloor(i-1)\frac{n}{k}\rfloor + 1..\lfloor i\frac{n}{k}\rfloor] \mid x_j = 0\}|$, that is, the number of 0s in block i of x. Moreover, for all $t \in \mathbb{N}$, let $X_t \coloneqq \min\{|y|_{0,i} \mid y \in \arg\max_{x \in P_{t+S(i-1)}} \textsc{BinVal}(x)\}$.

Expected runtime of optimizing a single block. Consider block $i \in [k-1]$, and consider iteration $t \in \mathbb{N}$ such that $X_t > 0$, such that all blocks in $[i-1]$ are optimized, and such that the maximum Hamming distance in block i was 0 in an iteration at most t after all blocks in $[i-1]$ are optimized.

We aim at bounding $\mathrm{E}[X_t - X_{t+1} \mid X_t]$ from below in the same fashion as in the proof of Lemma 5. To this end, let A denote the event that $X_t \ne X_{t+1}$, and let B denote the event that we choose a best individual $x \in P_t$ as parent for the mutation in iteration t. Analogously to the proof of Lemma 5, we get

$$\begin{aligned} \mathrm{E}[X_t - X_{t+1} \mid X_t] = \big(&\mathrm{E}[(X_t - X_{t+1}) \cdot \mathbb{1}\{B\} \mid X_t; A] \\ &+ \mathrm{E}[(X_t - X_{t+1}) \cdot \mathbb{1}\{\overline{B}\} \mid X_t; A]\big) \cdot \Pr[A \mid X_t]. \end{aligned}$$

We bound the three terms on the right-hand side separately from below. When we say that arguments are as or similar to *before*, we refer to the proof of Lemma 5.

Bounding $\Pr[A \mid X_t]$**.** As before, we need to create an individual that is better than the currently best individual x. To this end, it is sufficient to flip exactly one 0 in x in block i. For both 1-bit and plus mutation, this probability only differs by a constant factor, yielding

$$\Pr[A \mid X_t] \ge \frac{1}{\mu}\left(1 - \frac{1}{n}\right)^{n-1}\frac{X_t}{n}\left(1 - \left(1 - \frac{1}{n}\right)^n\right)^{-1} \ge \frac{X_t}{(e-1)\mu n}.$$

Bounding the expected difference intersected with B**.** Let $y \in \{0,1\}^n$ denote the offspring of the best individual x (before mutation). In the case of 1-bit mutation, $|y|_{0,i} = |x|_{0,i} - 1$ and thus $\mathrm{E}[(X_t - X_{t+1}) \cdot \mathbb{1}\{B\} \mid X_t; A] = \mathrm{E}[\mathbb{1}\{B\} \mid X_t; A]$, as exactly one bit is flipped during mutation. For plus mutation, more bits can flip. However, as we condition on A (implying that already at least one bit flipped), all other bit flips occur independently with probability $\frac{1}{n}$, leading to an analogous situation as before. Thus, for either mutation, we get

$$\mathrm{E}[(X_t - X_{t+1}) \cdot \mathbb{1}\{B\} \mid X_t; A] \ge \frac{1}{2} \cdot \mathrm{E}[\mathbb{1}\{B\} \mid X_t; A].$$

Bounding the expected difference intersected with $\overline{B}$**.** As before, let $x' \in P_t$ denote the non-best individual chosen for mutation, and let $y \in \{0,1\}^n$ denote its offspring. We consider again the decomposition

$$\begin{aligned} \mathrm{E}[(X_t - X_{t+1}) \cdot \mathbb{1}\{\overline{B}\} \mid X_t; A] = &\mathrm{E}[(|x|_{0,i} - |x'|_{0,i}) \cdot \mathbb{1}\{\overline{B}\} \mid X_t; A] \\ &+ \mathrm{E}[(|x'|_{0,i} - |y|_{0,i}) \cdot \mathbb{1}\{\overline{B}\} \mid X_t; A], \end{aligned}$$

We bound the second expectation analogously to the previous case, noting that x' and y differ in at least two positions in block i, as x' is not the best individual but y is, after mutation. We get

$$\mathrm{E}[(|x'|_{0,i} - |y|_{0,i}) \cdot \mathbb{1}\{\overline{B}\} \mid X_t; A] \ge \mathrm{E}[\mathbb{1}\{\overline{B}\} \mid X_t; A].$$

For the other expectation, we apply Theorem 8. Recall that we assume that the maximum Hamming distance in block i was 0 in an iteration at most t, satisfying one of the assumptions of the theorem. We choose $\varepsilon \coloneqq \frac{1}{k}$, noting that $\varepsilon \le s/(4096e\mu^4 n)$ is satisfied. Hence,

$$\mathrm{E}[(|x|_{0,i} - |x'|_{0,i}) \cdot \mathbb{1}\{\overline{B}\} \mid X_t; A] \ge -\frac{1}{4}.$$

Together with the previous bound, we get

$$\mathrm{E}[(X_t - X_{t+1}) \cdot \mathbb{1}\{\overline{B}\} \mid X_t; A] \ge \mathrm{E}[\mathbb{1}\{\overline{B}\} \mid X_t; A] - \frac{1}{4}.$$

Concluding a single phase. Combining the bounds for the expectations above, we get as before

$$\mathrm{E}[(X_t - X_{t+1}) \cdot \mathbb{1}\{B\} \mid X_t; A] + \mathrm{E}[(X_t - X_{t+1}) \cdot \mathbb{1}\{\overline{B}\} \mid X_t; A] \ge \frac{1}{4}.$$

Multiplying this with the bound for $\Pr[A \mid X_t]$ yields

$$\mathrm{E}[X_t - X_{t+1} \mid X_t] \geq \frac{X_t}{4(e-1)\mu n}.$$

Applying the multiplicative drift theorem to X and $T^{(i)}$, as defined in Section 3, shows that the expected number of iterations to optimize a single block $i \in [k-1]$ is at most $4(e-1)\mu n(1 + \ln\lceil\frac{n}{k}\rceil)$.

Concluding case 1. We conclude this case by summing over all possible phases $i \in [k-1]$, substituting the value of k as defined at the beginning of this part of the proof. Note that the sum of the waiting times of $O(kn/\mu^3) = O(\mu^2 n)$ iterations for reaching a maximum Hamming distance of 0 in each block for the first time, which follows as before from the second statement of Theorem 1 and the appropriate values chosen in the proof of Theorem 8, is of lower order and thus vanishes in the result.

• **Case 2—Full runtime.** The proof of this case is similar to case 1, but we choose $s \coloneqq \lfloor\frac{\mu^2}{2}\sqrt{n}\rfloor$ and $k \coloneqq \lceil 8192e\mu^2\sqrt{n}\rceil$, noting that $\frac{n}{k} \geq 1$ and $s \leq \frac{n}{2}$ by the bound on μ. We increase s (and thus adjusting k) because we show that the remaining s bits are optimized efficiently by increasing the shortest prefix of 1s in the population.

Optimizing all but $2s$ bits. The calculations until all bits but the remaining $2s$ are optimized are mostly analogous to case 1. We only check that the inequalities that involve s and k still hold, as these values are different now.

First, we observe that optimizing the first $k-1$ blocks implies that at most $2s$ bits are not optimized, as we have $(k-1)\lfloor\frac{n}{k}\rfloor \geq n - 2\frac{n}{k} \geq n - 2s$.

Next, for bounding the expected values, we see that $k \geq 2$ trivially holds by the definition of k. Thus, the only inequality left to check is the one we use when applying Theorem 8. As in case 1, we choose $\varepsilon \coloneqq \frac{1}{k}$, and we see that $\varepsilon \leq s/(4096e\mu^4 n)$ is satisfied.

Overall, we see that each block $i \in [k-1]$ is optimized in $4(e-1)\mu n(1+\ln\lceil\frac{n}{k}\rceil)$ iterations in expectation. Moreover, the expected time until the diversity in the block is 0 for the first time after all preceding blocks are optimized is, as in case 1, $O(n/\mu^3)$ iterations. By summing over those $k-1$ blocks, we have that the total expected number of iterations until all blocks are optimized is $O(k\mu n\log(\lceil\frac{n}{k}\rceil) + kn/\mu^3)$, which is in $O(\mu^3 n^{3/2}\log(n/\mu^2))$.

Optimizing the remaining $2s$ bits. We consider the shortest prefix of 1s in the population. That is, for all $t \in \mathbb{N}$, let $Y_t \coloneqq \min\{i \in \mathbb{N} \mid \exists x \in P_{t+S^{(k-1)}}, z \in \{0,1\}^{i-1} : x = 1^{n-i}0z\}$. We bound the expected number of iterations it takes to decrease the value of Y by 1 from above by an $a \in \mathbb{R}_{>0}$. Since we are done once Y is decreased $2s$ times, this results in a bound of $2sa$ for this part.

For decreasing Y, we count the number of individuals with the shortest 1-prefix. That is, for all $t \in \mathbb{N}$, let $Z_t \coloneqq |\{x \in P_{t+S^{(k-1)}} \mid \exists z \in \{0,1\}^{Y_t - 1} : x = 1^{n-Y_t}0z\}|$. Note that once Z reaches 0, then Y decreases by 1. Moreover, as long as Y does not decrease, Z does not increase. In the following, we bound the expected time until Z reaches 0 or Y decreases by discarding any updates to the population that decrease Y while Z is positive. This results in an upper bound on the expected time to decrease Y (by Z reaching 0).

Consider an iteration $t \in \mathbb{N}$ such that $t \geq S^{(k-1)}$ and $Z_t > 0$. In order to decrease Z_t, it is sufficient to choose one of the Z_t individuals with the shortest 1-prefix and flip their leftmost 0, denoted by event C. For both 1-bit and plus mutation, we get

$$\Pr[C \mid Z_t] \geq \frac{Z_t}{\mu}\frac{1}{n}\left(1 - \frac{1}{n}\right)^{n-1}\left(1 - \left(1 - \frac{1}{n}\right)^{n}\right)^{-1} \geq \frac{Z_t}{(e-1)\mu n}.$$

Since $Z \leq \mu$, we have by the multiplicative drift theorem that the expected number of iterations to decrease Y by 1 is at most $(e-1)\mu n(1+\ln\mu)$. As we need to decrease Y at most $2s \leq \mu^2\sqrt{n}$ times, as mentioned above, this results in a total number of at most $(e-1)\mu^3 n^{3/2}(1+\ln\mu)$.

Conclusion. The runtime bound for optimizing the remaining $2s$ bits is in the order of the runtime bound for optimizing the other, prior bits. This concludes this case and thus the proof. □

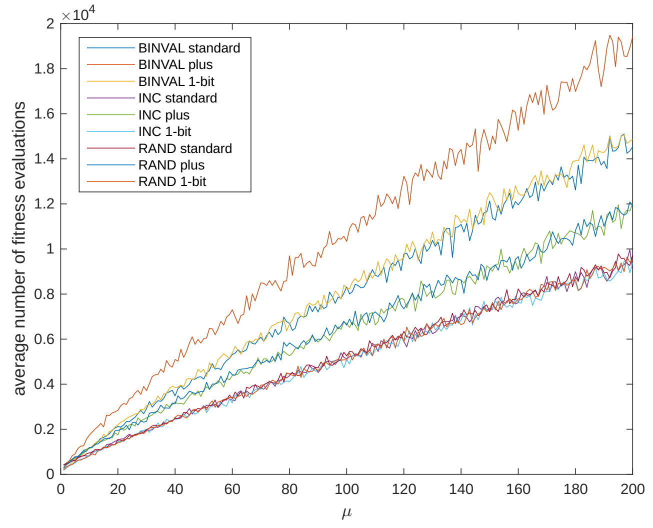

Figure 1: Average number of fitness evaluations of the (μ+1) EA to optimize BinVal, INC, and random linear functions for standard bit, 1-bit, and plus mutation. The problem size n is 50 here. Moreover, for each value of $\mu \in [200]$, we run 30 independent experiments and report on their average. We provide more details in Table 1.

5 Experimental Investigations

We complement our theoretical findings from Sections 3 and 4 through an experimental study. We have two goals in mind. (1) We are interested in seeing how tight our theoretical upper bounds are in terms of scaling in μ, as these factors appear to be an artifact of our proof method. (2) We are interested in seeing whether the performance for BinVal also translates to different types of pseudo-Boolean linear functions. Both of these goals aim at evaluating the importance of creating duplicates, as used in the proofs.

Setup. We consider the (μ+1) EA using standard bit mutation, plus mutation, and 1-bit mutation.

In addition to the function BinVal, we investigate the function $\mathrm{INC}(x) = \sum_{i\in[n]} i \cdot x_i$, which has increasing weights with the position of the bit string. Furthermore, we consider random linear functions where each weight is chosen uniformly at random from the set $[1..10000]$ to capture different types of linear functions. Our aim is to investigate the runtime in dependence of the population

Table 1: Mean number of fitness evaluations *(Mean)* and standard deviation *(Std)* of the setting from Figure 1. Moreover, we report on the p-values of a Mann–Whitney U test of comparing the average performances of the three different mutation operators pairwise. That is, p_1 denotes the test of standard bit vs. plus mutation, p_2 denotes the test of standard bit vs. 1-bit, and p_3 denotes the test of plus vs. 1-bit mutation. Bold numbers indicate statistical significance, that is, a p-value of at most 0.05.

Fitness function	μ	**Standard**		**Plus**			**1-Bit**			
		Mean	Std	Mean	Std	p_1-val	Mean	Std	p_2-val	p_3-val
	1	412	164.701	292	93.189	**0.000**	192	60.492	**0.000**	**0.000**
	2	503	135.635	495	181.921	0.501	327	92.611	**0.000**	**0.000**
	5	746	186.430	908	196.534	**0.002**	676	148.099	0.179	**0.000**
	10	1144	257.940	1740	419.301	**0.000**	1135	264.708	0.767	**0.000**
BinVal	20	2123	325.910	2841	528.984	**0.000**	2197	497.897	0.352	**0.000**
	50	4226	733.952	6022	1234.003	**0.000**	4655	829.073	0.081	**0.000**
	100	8317	1172.915	10337	2238.968	**0.000**	8243	1178.151	0.848	**0.000**
	150	12127	1795.684	14385	2547.257	**0.000**	12498	1879.594	0.329	**0.001**
	200	14531	2455.809	19411	3104.897	**0.000**	14864	1827.707	0.352	**0.000**
	1	439	186.610	281	81.177	**0.000**	178	48.377	**0.000**	**0.000**
	2	455	165.603	420	164.583	0.420	278	61.568	**0.000**	**0.000**
	5	611	183.260	718	240.815	**0.036**	502	116.813	**0.018**	**0.000**
	10	823	148.560	1171	273.864	**0.000**	834	174.766	0.953	**0.000**
INC	20	1506	265.880	1750	360.279	**0.006**	1449	291.664	0.429	**0.002**
	50	2948	461.604	3726	813.798	**0.000**	3059	523.466	0.399	**0.001**
	100	5122	673.706	6694	1323.273	**0.000**	5090	743.822	0.929	**0.000**
	150	7358	1039.708	9158	1602.601	**0.000**	7275	1047.375	0.589	**0.000**
	200	9106	954.140	12011	2201.030	**0.000**	9371	1431.626	0.589	**0.000**
	1	408	132.788	265	80.045	**0.000**	197	62.659	**0.000**	**0.001**
	2	473	160.341	428	161.759	0.188	327	104.608	**0.000**	**0.009**
	5	595	165.180	782	258.535	**0.001**	557	171.587	0.225	**0.000**
	10	955	267.248	1190	264.307	**0.001**	826	159.760	**0.039**	**0.000**
Random	20	1550	232.036	1952	314.544	**0.000**	1402	302.220	**0.012**	**0.000**
	50	3018	480.281	3679	615.092	**0.000**	3015	451.416	0.767	**0.000**
	100	5257	856.777	6622	1275.860	**0.000**	5075	701.017	0.520	**0.000**
	150	7216	799.634	9072	1364.549	**0.000**	7327	865.041	0.515	**0.000**
	200	9725	1211.971	11909	1923.706	**0.000**	9485	1214.229	0.464	**0.000**

size. We consider problems of size $n = 50$ and carry out 30 runs for each value of $\mu \in [200]$. In the case of random linear functions, 30 random linear functions are constructed and one run per algorithm is carried out for each of them. The average number of fitness evaluations to obtain the optimal solution is given in Figure 1, and Table 1 displays average number of fitness evaluations, standard deviations, and p-values using Mann–Whitney U tests for comparing pairs of algorithms for $\mu \in \{1, 2, 5, 10, 20, 50, 100, 150, 200\}$. We call a result significant if the p-value is at most 0.05.

Observations. Examining Figure 1, we clearly see a separation in terms of the problems and algorithms considered. The top 3 graphs are for the problem BinVal and show a significantly higher runtime for plus mutation compared to standard bit mutation and 1-bit mutation. The next 2 lower graphs, shown in blue and green, belong to plus mutation applied to INC and random linear functions. The remaining 4 graphs, with the lowest average runtime per value of μ, belong to standard bit mutation and 1-bit mutation when used to optimize INC and random linear functions. Altogether, it shows that standard bit mutation and 1-bit mutation perform very similarly and outperform plus mutation for $\mu \geq 5$. Furthermore, BinVal is harder to be optimized by the considered algorithms than INC and random linear functions, and there is no clear difference in terms of difficulty when tackling INC or random linear functions.

These insights can also be gained from the results displayed in Table 1, where p-values for the comparison of standard bit mutation and 1-bit mutation shown in the second last column are higher than 0.05 for $\mu \geq 10$. For the special case of $\mu = 1$ which results in the classic (1+1) EA, we see that standard bit mutation is clearly outperformed by plus mutation which is in turn outperformed by 1-bit mutation. This further confirms that the different mutation operators have a different behavior when used in a (μ+1) EA with sufficiently large μ compared to the (1+1) EA.

Comparing the results displayed in Figure 1 to the obtained theoretical runtime bounds, we observe that the runtime increases quite moderately with μ (potentially linear), which suggests that our runtime bounds are not tight in terms of their dependence on μ.

6 Conclusion

We contributed to the theoretical analysis of population-based evolutionary algorithms and have provided new insights and improved upper bounds for the optimization of BinVal. Our analysis relies on obtaining new tools for showing that the population is not particularly diverse during the optimization run when considering only parts of the individuals (Theorems 6 and 8). These results rely on creating offspring that are identical to the parent within the considered part. For 1-bit and plus mutation, which cannot create a copy of the parent, our bounds are a bit weaker.

As improved results for standard bit mutation, we obtain bounds of $O(n \log n)$ for constant μ and $o(n^2)$ as long as $\mu = o(n^{1/5}/\log n)$ (Theorem 4). For 1-bit and plus mutation, we achieve similar results. In particular, for constant μ, we have an approximation result of $O(n \log n)$ and an optimization result of $O(n^{3/2} \log n)$.

Our experimental results show a much better scaling behavior in terms of μ than our theoretical bounds. Moreover, they indicate that random linear functions and one with linearly increasing weights seem to be easier than BinVal.

We note that both our theoretical and empirical results for plus mutation are orthogonal to those of Carvalho Pinto and Doerr [4]: In our work, plus mutation seems to be harmful, whereas it is useful in the setting of Carvalho Pinto and Doerr [4], who consider the use of crossover. This suggests that the plus operator may be more suited for algorithms that employ crossover.

Although we believe that our work marks an important step to understanding population-based evolutionary algorithms better, there remain several interesting problems. As our experiments also suggest, the dependence of our theoretical bounds on μ is possibly not optimal. As we discussed in Section 3, traditional methods result in bounds with a (near-)linear dependence on μ, which we believe to be true. Furthermore, our current proof method requires populations of sublinear size in n. It would be interesting to see if this method can be extended to cover larger population sizes.

Another interesting future extension is to consider general linear functions. Our experiments suggest that BinVal is a particularly hard example. We thus expect that the $(\mu+1)$ EA also optimizes other linear functions in a similar order of expected fitness evaluations.

Moreover, we deem the analysis of other mutation operators than standard bit mutation important. Standard bit mutation allows for easy copies of best individuals, thus leading to a lower diversity in the population, which in turn allows to use similar methods as for population-less algorithms. As we show, similar arguments can be translated to mutation operators that cannot create copies, but it remains open just how largely this impacts the runtime.

Acknowledgments

This research benefited from the support of the FMJH Program Gaspard Monge for optimization and operations research and their interactions with data science. This work has also been supported by the Australian Research Council through grant FT200100536 and by the Independent Research Fund Denmark (grant ID 10.46540/2032-00101B). Martin S. Krejca was supported by the EuroTech Visiting Researcher Programme. The authors thank the participants and organizers of Dagstuhl seminar 24271 for facilitating discussions that supported further developing the ideas discussed in this work.

References

[1] Sumit Adak and Carsten Witt. A runtime analysis of the multi-valued compact genetic algorithm on generalized LeadingOnes. In *Proc. of EvoCOP '25*, volume 15610, pages 1–17, 2025.
[2] Adetunji David Ajimakin and V. Susheela Devi. The competing genes evolutionary algorithm: Avoiding genetic drift through competition, local search, and majority voting. *IEEE Transactions on Evolutionary Computation*, 27(6):1678–1689, 2023.
[3] Alistair Benford and Per Kristian Lehre. Runtime analysis of coevolutionary algorithms on a class of symmetric zero-sum games. In *Proc. of GECCO '24*, pages 1542–1550, 2024.
[4] Eduardo Carvalho Pinto and Carola Doerr. A simple proof for the usefulness of crossover in black-box optimization. In *Proc. of PPSN '18*, pages 29–41, 2018.
[5] Marcel Chwiałkowski, Benjamin Doerr, and Martin S. Krejca. Runtime analysis of the compact genetic algorithm on the LeadingOnes benchmark. *IEEE Transactions on Evolutionary Computation*, pages 1–10, 2025. Early access.
[6] Sylvain Colin, Benjamin Doerr, and Gaspard Férey. Monotonic functions in EC: anything but monotone! In *Proc. of GECCO '14*, pages 753–760, 2014.
[7] Benjamin Doerr and Carola Doerr. The impact of random initialization on the runtime of randomized search heuristics. *Algorithmica*, 75(3):529–553, 2016.
[8] Benjamin Doerr and Martin S. Krejca. Significance-based estimation-of-distribution algorithms. *IEEE Transactions on Evolutionary Computation*, 24 (6):1025–1034, 2020.
[9] Benjamin Doerr and Frank Neumann, editors. *Theory of Evolutionary Computation—Recent Developments in Discrete Optimization*. Springer, 2020. Also available at http://www.lix.polytechnique.fr/Labo/Benjamin.Doerr/doerr_neumann_book.html.
[10] Benjamin Doerr, Daniel Johannsen, and Carola Winzen. Multiplicative drift analysis. *Algorithmica*, 4(64):673–697, 2012.
[11] Benjamin Doerr, Carola Doerr, and Franziska Ebel. From black-box complexity to designing new genetic algorithms. *Theoretical Computer Science*, 567:87–104, 2015.
[12] Benjamin Doerr, Huu Phuoc Le, Régis Makhmara, and Ta Duy Nguyen. Fast genetic algorithms. In *Proc. of GECCO '17*, pages 777–784, 2017.
[13] Benjamin Doerr, Carsten Witt, and Jing Yang. Runtime analysis for self-adaptive mutation rates. *Algorithmica*, 83(4):1012–1053, 2021.
[14] Stefan Droste. Not all linear functions are equally difficult for the compact genetic algorithm. In *Proc. of GECCO '05*, pages 679–686, 2005.
[15] Stefan Droste, Thomas Jansen, and Ingo Wegener. On the analysis of the (1+1) evolutionary algorithm. *Theoretical Computer Science*, 276:51–81, 2002.
[16] Tobias Friedrich, Timo Kötzing, Martin S. Krejca, and Andrew M. Sutton. The compact genetic algorithm is efficient under extreme Gaussian noise. *IEEE Transactions on Evolutionary Computation*, 21(3):477–490, 2017.
[17] Christian Gießen and Timo Kötzing. Robustness of populations in stochastic environments. *Algorithmica*, 75(3):462–489, 2016.
[18] Bruce Hajek. Hitting-time and occupation-time bounds implied by drift analysis with applications. *Advances in Applied Probability*, 13:502–525, 1982.
[19] George T. Hall, Pietro S. Oliveto, and Dirk Sudholt. On the impact of the performance metric on efficient algorithm configuration. *Artificial Intelligence*, 303: 103629:1–103629:27, 2022.
[20] Jun He and Xin Yao. A study of drift analysis for estimating computation time of evolutionary algorithms. *Natural Computing*, 3(1):21–35, 2004.
[21] Martin S. Krejca and Carsten Witt. A flexible evolutionary algorithm with dynamic mutation rate archive. In *Proc. of GECCO '24*, pages 1578–1586, 2024.
[22] Per Kristian Lehre and Carsten Witt. Tail bounds on hitting times of randomized search heuristics using variable drift analysis. *Combinatorics, Probability & Computing*, 30(4):550–569, 2021.
[23] Johannes Lengler. A general dichotomy of evolutionary algorithms on monotone functions. *IEEE Transactions on Evolutionary Computation*, 24(6):995–1009, 2020.
[24] Johannes Lengler. Drift analysis. In Benjamin Doerr and Frank Neumann, editors, *Theory of Evolutionary Computation: Recent Developments in Discrete Optimization*, pages 89–131. Springer, 2020. Also available at https://arxiv.org/abs/1712.00964.
[25] Johannes Lengler and Simone Riedi. Runtime analysis of the $(\mu + 1)$-EA on the dynamic BinVal function. *Springer Natural Computer Science*, 3(4):324, 2022.
[26] Johannes Lengler and Xun Zou. Exponential slowdown for larger populations: The $(\mu+1)$-EA on monotone functions. *Theoretical Computer Science*, 875:28–51, 2021.
[27] Johannes Lengler, Andre Opris, and Dirk Sudholt. Analysing equilibrium states for population diversity. *Algorithmica*, 86(7):1–35, 2024.
[28] Andrei Lissovoi and Carsten Witt. MMAS versus population-based EA on a family of dynamic fitness functions. *Algorithmica*, 75(3):554–576, 2016.
[29] Andre Opris, Johannes Lengler, and Dirk Sudholt. A tight $O(4^k/p_c)$ runtime bound for a $(\mu+1)$ GA on Jump$_k$ for realistic crossover probabilities. In *Proc. of GECCO '24*, pages 1605–1613, 2024.
[30] Ingo Wegener. *Methods for the Analysis of Evolutionary Algorithms on Pseudo-Boolean Functions*, pages 349–369. Springer US, 2002.
[31] Carsten Witt. Runtime analysis of the $(\mu+1)$ EA on simple pseudo-Boolean functions. *Evolutionary Computation*, 14(1):65–86, 2006.
[32] Carsten Witt. Domino convergence: why one should hill-climb on linear functions. In *Proc. of GECCO '18*, pages 1539–1546, 2018.