Runtime Analysis of a Compact Genetic Algorithm with High Selection Pressure

Sumit Adak
DTU Compute, Technical University of Denmark
Kongens Lyngby, Denmark
suad@dtu.dk

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

Abstract

The Compact Genetic Algorithm (cGA) is an estimation-of-distribution algorithm that has been receiving much attention especially in the runtime analysis community in recent years. It comes with a single parameter K determining its strength of updates. Contrary to other estimation-of-distribution algorithms like the UMDA, the standard cGA does not have a parameter controlling its selection pressure.

Already the original authors of the cGA (Harik et al. 1999) suggested a generalization exhibiting larger selection pressure by sampling more than the usual number of two individuals per iteration. We analyze such a generalization following closely the original idea, but simplifying it by sampling λ individuals per iteration and updating the probabilistic model according to the difference of the fittest and the least fit sample. The resulting algorithm is analyzed primarily on the well-established OneMax benchmark, and a tight bound of $\Theta(K\sqrt{n/\log\lambda})$ for the number of iterations to find the optimum is obtained, with the upper bound being $O(n\log n/\log\lambda)$ for a suitable K. The lower bound holds for all functions with a unique optimum. To prove our results, we apply strong probabilistic tools, including Slud's inequality, which has not been used in the runtime analysis of evolutionary computation before.

Since the speedup in the number of iterations is limited (in the number of function evaluations, we observe no speedup), we also present an example where the higher selection pressure is essential to find the global optimum efficiently. We study this function mostly empirically, and we also supplement further experimental insights on OneMax.

CCS Concepts

• **Theory of computation → Evolutionary algorithms**.

Keywords

Estimation-of-distribution algorithms, compact genetic algorithm, offspring population, selection pressure, OneMax.

ACM Reference Format:
Sumit Adak and Carsten Witt. 2025. Runtime Analysis of a Compact Genetic Algorithm with High Selection Pressure. In *Foundations of Genetic Algorithms XVIII (FOGA '25), August 27–29, 2025, Leiden, Netherlands.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3729878.3746630

FOGA '25, Leiden, Netherlands

ACM ISBN 979-8-4007-1859-5/2025/08
https://doi.org/10.1145/3729878.3746630

1 Introduction

The theory of estimation-of-distribution algorithms (EDAs) has received increased interest in the last 10 years, in particular from the perspective of rigorous runtime analyses. Following the successful approach from the theory of evolutionary algorithms, researchers have bounded the runtime of the univariate marginal distribution algorithm (UMDA) [11, 12, 20, 21, 35], population-based incremental learning (PBIL) [37, 38, 49, 50], the Compact Genetic Algorithm (cGA) and further EDAs [6, 22] on a variety of benchmark functions and optimization scenarios, see also the overview article [36] from 2020. Especially the cGA, arguably the simplest univariate EDA, was intensively studied in recent years [1–3, 9, 17, 19, 26, 30, 32, 40, 42, 46, 48].

The cGA maintains a probability distribution on $\{0,1\}^n$ that is continuously updated by sampling two individuals (offspring) from the current distribution and updating it in the direction of the better offspring. Although this often gives runtimes that are competitive with evolutionary algorithms on the typical benchmark functions (assuming an optimal setting of the cGA's parameter K, representing the hypothetical population size), it is natural to ask whether more than two samples per iteration may be used to increase its selection pressure. Already the inventors of the cGA [31] suggest a generalization that samples λ offspring and updates in the direction of the fittest of these by computing the vector differences with the remaining $\lambda - 1$ offspring; moreover, they suggest a more involved offspring competition called round-robin. However, the majority of the literature on the cGA tries to limit the number of offspring to 2 and suggests other measures like elitism and injection of additional randomness to improve its empirical runtime [4, 5].

In this paper, we are interested in the impact of larger offspring population sizes in the cGA. We propose a basic generalization of the classical cGA that samples λ offspring from the current distribution and updates the model according to the difference of the best and the worst offspring. This only requires a single update of the probabilistic model per iteration, while the variants with larger offspring populations suggested in [31] are more involved and require at least $\lambda - 1$ model updates per iteration; we will argue in Section 2 that our model is essentially equivalent to Harik et al.'s first model with larger offspring populations. We are interested in whether our generalization of the cGA, analyzed on the well-known OneMax benchmark, can optimize the function in a lower number of iterations than the classical cGA. We note that the number of function evaluations per iteration is λ, so we focus on the "parallel" runtime, assuming that the λ offspring can be created and evaluated in parallel. This is similar to the viewpoint taken for $(1+\lambda)$ EAs and $(1,\lambda)$ EAs, as discussed later.

Our contribution: We give a rigorous, asymptotically tight runtime analysis of the above-mentioned generalization of the cGA with an offspring population of size λ, called cGA$_\lambda$. The main runtime bound, given for OneMax, essentially reflects what Chernoff bounds intuitively suggest: when individuals are sampled with the highest possible variance in the model, i. e., uniformly from $\{0,1\}^n$, then with probability $\Theta(1/\lambda)$ the maximum deviation from the expected value is in the order $\Theta(\sqrt{n\log\lambda})$, so we expect a constant fraction of the offspring to give this progress. This leads to a lower bound of $\Omega(K\sqrt{n/\log\lambda})$ on the expected runtime with not too much effort, and this bound even holds for all functions with a unique optimum. However, it is significantly more involved to prove the matching upper runtime bound $O(K\sqrt{n/\log\lambda})$ on OneMax for a suitable choice of the parameter K, which is the main contribution of this paper. We explain the main challenges and relate to existing literature on offspring populations in the following.

The role of offspring populations has been intensively studied in evolutionary computation before, especially on the OneMax function (see, e. g., [7, 34, 43]). For certain (1+λ) EAs/$(1,\lambda)$ EAs and certain ranges of λ, the speedup in the number of generations can be $\Theta(\lambda)$. In contrast to this, the cGA$_\lambda$ provably only reduces the number of generations by $\Theta(\log\lambda)$. This hinges on fundamentally different working principles of these types of search heuristics.

Typically, EAs choose the best of λ offspring for their next generation. This naturally leads to the analysis of order statistics, e. g., the maximum number of bits flipped in an application of the mutation operator. Such analyses of order statistics in the context of mutation are challenging, but some good bounds are known. Unfortunately, they do not carry well over to the analysis of EDAs. Roughly speaking, the sampling variance of mutation operators is usually much lower than that of the sampling procedure in EDAs, which start with a uniform distribution of variance $\Theta(n)$, and only gradually lower this variance over a longer period of time, resulting in Poisson-binomial distributions of varying success probability per trial (i. e., per bit value sampled).

For the case of a truly uniform distribution, [13] have precisely studied the order statistics of the best OneMax-value of λ samples. Their approach is to a larger extent based on approximations of the binomial distribution with success probability $1/2$ by the normal distribution and the Berry-Esseen inequality. However, we do not find it obvious how to use the Berry-Esseen inequality in our study. When considering the best of λ offspring drawn from the same distribution with a certain high variance, we are confronted with individual success probabilities of $\Theta(1/\lambda)$, where λ may be a growing function. This corresponds to values of the cumulative density of the normal distribution that are $1-o(1)$. The Berry-Esseen inequality is not strong enough here since it only bounds the absolute, but not the relative error of the normal approximation by a constant. An alternative approach from [10], based on tools from [8], to bound the order statistic from the best of λ uniform individuals via the normal approximation is not directly applicable to the analysis of the cGA either since it comes with several restrictions, most crucially the assumption of a super-constant variance. Our approach for the analysis of the maximum order statistic of λ offspring generated by a Poisson-binomial distribution still proceeds by iterative approximations via binomial and then normal distributions, using strong, but easy-to-apply tools for estimations of the Poisson-binomial distribution [47]. To obtain non-asymptotic bounds on the drift, we apply Slud's inequality [44], which we have not seen used earlier in a runtime analysis of a randomized search heuristic.

We note that the UMDA also creates a larger number λ of offspring than the classical cGA. However, typically, not the maximum order statistic is relevant in these analyses (at least in the context of OneMax; the frequently studied LeadingOnes function leads to a different optimization scenario) but a constant fraction of the λ best, e. g., by choosing $\mu=\lambda/2$ individuals to update the probabilistic model.

Our main result recovers the time $O(K\sqrt{n})$ (holding with high probability) for the classical cGA with $\lambda=2$. It is arguably a simpler proof than the probabilistic approach from the earlier works showing tight bounds for the cGA on OneMax, most notably [46], which approach was also used in several later works (e. g., [1, 26, 40, 48]). In this paper, we exclusively rely on a drift analysis [39] of a potential function that combines all elements of the frequency vector of the cGA, assuming low genetic drift. This avoids the involved probabilistic arguments related to the drifts of single frequencies from the earlier works and functions without the distinction between random-walk steps and biased steps of frequencies in the above-mentioned works. In this sense, our analysis of the progress made in an iteration of the cGA$_\lambda$ is much closer to the approach from [24] for the classical cGA on OneMax, which was used and refined in [17] to analyze the cGA on jump functions. However, in this paper, we have to analyze the progress of the best of λ offspring.

Furthermore, this paper presents a new additive drift theorem with tail bounds for processes where not only the expected one-step change, but also a lower probability bound for a minimum change is known. Plugging in the results of our revised analysis of the drift of the potential function, we arrive at a high-probability bound without tediously analyzing moment-generating functions of drifts like in [46]. In essence, these arguments also appear in the proof of [17, Lemma 15] but we feel that a self-contained lemma analyzing the progress and success probabilities is useful.

This paper is structured as follows. In Section 2, we formally introduce the cGA$_\lambda$ with an offspring population size of λ. Moreover, we present several technical tools that are important for its analysis. Section 3 proves a general lower bound on the runtime for the cGA$_\lambda$, which includes OneMax as a special case. Section 4 states and proves a tight upper bound on the runtime of the cGA$_\lambda$ on OneMax, which is our main result. Since increasing λ only gives a small, logarithmic speed-up on OneMax, we present in Section 5 an example, where the runtime decreases drastically, possibly by an exponential term, when λ is increased from its standard setting 2 to a slightly larger value. This is empirically demonstrated in Section 6, along with further experiments detailing the influence of λ on the runtime on OneMax.

2 Preliminaries

2.1 Algorithm

The cGA$_\lambda$, displayed in Algorithm 1, is a randomized search heuristic for the optimization of pseudo-Boolean functions $f\colon\{0,1\}^n\to\mathbb{R}$. It evolves a probability vector $p_t=(p_{t,1},\dots,p_{t,n})$, also called frequency vector, which is initialized at $(1/2, \dots, 1/2)$, over time. It generalizes the well-known cGA ([31]) by creating $\lambda \geq 2$ samples $x_1, \dots, x_\lambda$, where $x_i = (x_{i,1}, \dots, x_{i,n})$ for $i \in \{1, \dots, \lambda\}$, according to p_t, instead of a fixed number of 2 samples. The best and the worst of these samples with respect to f are taken to adjust the probability vector such that each entry is increased or decreased by $1/K$ in the direction of the bit value of the best in case of a difference. Here $K \geq 2$ is the main parameter of the algorithm (sometimes called *hypothetical population size*). As usual in runtime analyses, we restrict all entries of the frequency vector to a proper subinterval of $[0, 1]$ to avoid premature, irreversible convergence at the extremal values 0 and 1. Here we choose the interval as $[1/((\lambda-1)n), 1-1/((\lambda-1)n)]$, generalizing the $[1/n, 1-1/n]$ from the classical cGA. Note that we choose the borders of the interval depending on λ. This maintains in each iteration a probability of $\Theta(1/n)$ of observing an offspring that samples a bit differently from the value that the border emphasizes.

Algorithm 1: Compact Genetic Algorithm with λ Offspring (cGA$_\lambda$)

```
t ← 0;
$p_{t,1} \leftarrow p_{t,2} \leftarrow \cdots \leftarrow p_{t,n} \leftarrow 1/2$;
while termination criterion not met do
    for $i \in \{1, \dots, \lambda\}$ do
        for $j \in \{1, \dots, n\}$ do
            $x_{i,j} \leftarrow 1$ with prob. $p_{t,i}$;
            $x_{i,j} \leftarrow 0$ with prob. $1 - p_{t,i}$;
    $x^{\max} := \arg\max\{f(x) \mid x \in \{x_1, \dots, x_\lambda\}\}$ (breaking ties arbitrarily);
    $x^{\min} := \arg\min\{f(x) \mid x \in \{x_1, \dots, x_\lambda\}\}$ (breaking ties arbitrarily);
    for $i \in \{1, \dots, n\}$ do
        $\overline{p}_{t+1,i} \leftarrow p_{t,i} + \frac{1}{K}(x_i^{\max} - x_i^{\min})$;
        $p_{t+1,i} \leftarrow \max\{\min\{\overline{p}_{t+1,i}, 1 - \frac{1}{(\lambda-1)n}\}, \frac{1}{(\lambda-1)n}\}$;
    $t \leftarrow t + 1$;
```

We define the *runtime* as the smallest value of $t\lambda$ such that at iteration t a solution of maximum f-value is sampled. Note that the runtime is by λ larger than the number of iterations to account for the number of fitness function evaluations during an iteration; this reflects the common cost measure in runtime analyses. We will often bound the expected number of iterations to sample an optimum, which can be considered a kind of parallel runtime in which the λ offspring are generated and evaluated in parallel.

To ease the theoretical analyses, we follow throughout this paper the common assumption that frequencies are well behaved [17], defined as that $1 - \frac{2}{(\lambda-1)n}$, i. e., the length of the allowed frequency interval, is an even multiple of $1/K$; in other words, the set of possible frequency settings is $\{\frac{1}{(\lambda-1)n}, \frac{1}{(\lambda-1)n} + \frac{1}{K}, \dots, \frac{1}{2}, \dots, 1 - \frac{1}{(\lambda-1)n} - \frac{1}{K}, 1 - \frac{1}{(\lambda-1)n}\}$.

The cGA$_\lambda$ is similar, but not identical to the cGA with high selection pressure from [31] using an offspring population of size s. Their algorithm samples s offspring in the same way as our algorithm, but then carries out $s-1$ frequency updates by comparing all other samples to the best sample and updating all entries by $\pm 1/K$ in the direction of the best. Hence, frequencies will only be increased if the best has a 1-entry at the bit position and only be decreased if it has a 0-entry; however, the update in the corresponding positions may take place up to $s-1$ times, depending on how many other samples differ in the bit. This is not very different from our cGA$_\lambda$ with an update parameter adjusted by a factor of $1/(s-1)$, so we use our simplified version using the best and worst sample. We believe that Harik et al.'s version could be analyzed with similar methods as our cGA$_\lambda$. However, no theoretical analysis of that algorithm exists.

2.2 Technical Tools

In the remainder of this section, we will state several technical tools required for the analysis of the cGA$_\lambda$. The first one is related to genetic drift. Even though the statement that we will use has been originally developed for the classical cGA, it immediately carries over to the cGA$_\lambda$ since the underlying property of a weak preference analyzed in [23, Lemma 3] works in the same way for all $\lambda \geq 2$. Formally, we say that a pseudo-Boolean function f has a weak preference for a value $j \in \{0, 1\}$ at a position $i \in \{1, \dots, n\}$, if and only if, for all $x_1, \dots, x_n \in \{0, 1\}$, it holds that

$$f(x_1, \dots, x_{i-1}, x_i, x_{i+1}, \dots, x_n) \leq f(x_1, \dots, x_{i-1}, j, x_{i+1}, \dots, x_n).$$

This gives us the following lemma limiting genetic drift. It is directly extracted from Theorem 4 in [23].

Lemma 1 ([23]). *Let f be a pseudo-Boolean function with a weak preference for 1 at position $i \in \{1, \dots, n\}$. Consider the cGA$_\lambda$ optimizing f with parameter K. Let $T \in \mathbb{N}$. Then we have for all $T \geq 0$ and $\gamma > 0$ that*

$$\Pr\left[\min_{t \in \{0, \dots, T\}} p_{t,i} \leq p_{0,i} - \gamma\right] \leq 2\exp\left(-\frac{\gamma^2 K^2}{T}\right).$$

Another lemma that we use and that was originally developed for the classical cGA deals with the effects of capping frequencies at one of the borders $\{\frac{1}{(\lambda-1)n}, 1 - \frac{1}{(\lambda-1)n}\}$. It generalizes a statement from [17, Lemma 8]. The notation $\mathrm{Bin}(n, p)$ denotes an independently drawn random variable following a binomial distribution with parameters n and p, and the inequalities in the statement of the lemma mean stochastic domination.

Lemma 2. *Consider the cGA$_\lambda$ and let $D_t := |x^{\max}|_1 - |x^{\min}|_1$ be the difference in the number of one-bits of best and worst offspring at an arbitrary time t. Then for the difference of the frequency vectors, it holds that*

$$\frac{D_t - \mathrm{Bin}(n, 2/n)}{K} \leq \sum_{i=1}^{n} (p_{t+1,i} - p_{t,i}) \leq \frac{D_t + \mathrm{Bin}(n, 2/n)}{K}.$$

Proof. We follow the ideas of the proof of Lemma 8 in [17]. We call a frequency update at a bit position *blocked* if the best and the worst individual differ in the position and the update by $1/K$ would break a frequency border $\frac{1}{(\lambda-1)n}$ or $1 - \frac{1}{(\lambda-1)n}$. Because of the well-behaved frequency assumption, this can only happen for frequencies at the border. If an update is not blocked, then the frequency changes by $\frac{1}{K}$ times the difference of the corresponding bit in $x^{\max}$ and $x^{\min}$.

For each frequency that is at a border there occur λ samples that with probability $\frac{1}{(\lambda-1)n}$ each sample a value that may lead to a blocked update. Hence, the probability of a blocked update per bit is bounded from above by $\frac{\lambda}{(\lambda-1)n} \le \frac{2}{n}$. This happens independently at each bit with a border frequency, of which there are at most n. □

The number of one-bits in an individual sampled by the cGA$_\lambda$ with frequency vector $(p_1, \dots, p_n)$ follows a Poisson-binomial distribution with corresponding probability vector. We will need several results on the tail of the Poisson-binomial distribution. The following classical result by Hoeffding [33] states that the binomial distribution is the most spread out over all Poisson-binomial distributions with a given expectation.

Lemma 3 ([33]). *Let X follow a Poisson-binomial distribution with success probabilities $p_1, \dots, p_n$. Let $\bar{p} = \frac{1}{n}\sum_{i=1}^{n} p_i$, and let $\tilde{X} \sim \mathrm{Bin}(n, \bar{p})$. Then*

(1) $\mathrm{P}[X \le k] \le \mathrm{P}[\tilde{X} \le k]$ *for* $0 \le k \le n\bar{p} - 1$
(2) $\mathrm{P}[X \le k] \ge \mathrm{P}[\tilde{X} \le k]$ *for* $n\bar{p} \le k \le n$.

We will also apply a kind of opposite of the previous lemma. The following result by Gleser [28], which generalizes similar statements from [33], shows that among the Poisson-binomial distributions with a fixed expectation, the ones with extremal success probabilities are most concentrated around the expectation. In the lemma, the term that a vector $(p_1, \dots, p_n)$ *majorizes* a vector $(p'_1, \dots, p'_n)$ means that $\sum_{i=1}^{k} p_i \le \sum_{i=1}^{k} p'_i$ for $1 \le k \le n-1$ and $\sum_{i=1}^{n} p_i = \sum_{i=1}^{n} p'_i$.

Lemma 4 ([28]). *Let X follow a Poisson-binomial distribution with parameters $p_1, \dots, p_n$ and $\tilde{X}$ a Poisson-binomial distribution with parameters $p'_1, \dots, p'_n$, where $\sum_{i=1}^{n} p_i = \sum_{i=1}^{n} p'_i$. Let $\bar{p} = \frac{1}{n}\sum_{i=1}^{n} p_i$. If $(p_1, \dots, p_n)$ majorizes $(p'_1, \dots, p'_n)$ then*

(1) $\mathrm{P}[X \le k] \le \mathrm{P}[\tilde{X} \le k]$ *for* $0 \le k \le n\bar{p} - 2$
(2) $\mathrm{P}[X \le k] \ge \mathrm{P}[\tilde{X} \le k]$ *for* $n\bar{p} + 2 \le k \le n$.
(3) $\mathrm{Var}(X) \le \mathrm{Var}(X')$.

To prove our upper bound on the runtime, the following lemma is central. It gives a non-asymptotic bound on the tail of the binomial distribution via the normal distribution.

Lemma 5 (Slud's inequality [44]). *Let $X \sim \mathrm{Bin}(n, p)$ and k be an integer. If $p \le 1/4$ and $k \ge np$ or if $p \le 1/2$ and $np \le k \le n(1-p)$, we have*

$$\mathrm{P}[X \ge k] \ge 1 - \Phi\left(\frac{k - np}{\sqrt{np(1-p)}}\right),$$

where Φ denotes the cumulative distribution function of the standard normal distribution.

We will use the following lemma to bound the upper tail of the standard normal distribution. The lower bound of the lemma is only useful for $x > 1$.

Lemma 6 ([25]). *The cumulative density function of the standard normal distribution satisfies for $x > 0$ that*

$$\left(\frac{1}{x} - \frac{1}{x^3}\right)\frac{1}{\sqrt{2\pi}}e^{-x^2/2} \le 1 - \Phi(x) \le \frac{1}{x}\frac{1}{\sqrt{2\pi}}e^{-x^2/2}$$

We will also need to bound the probability that a binomially distributed random variable is at least its expectation. To that end, the following lemma from [29] (see also [15] for an elementary analysis) comes in handy.

Lemma 7 (Theorem 1 in [29]). *Let $X \sim \mathrm{Bin}(n, p)$ and $p > 1/n$. Then $\mathrm{P}[X \ge \mathrm{E}[X]] \ge 1/4$.*

Finally, we will apply a new drift theorem. It assumes a lower bound on the probability of improving steps of a certain size and uses straightforward arguments related to Chernoff bounds for geometric random variables. We note that similar arguments have been used in the literature several times before; most notably in the proof of the upper bound, holding with high probability, for the runtime of the classical cGA on jump functions [17]. However, we did not find the following formulation as a self-contained drift theorem. In this paper, we only use its 2nd statement, but for the sake of completeness, we also include a bound on the expected hitting time.

Theorem 8. *Let X_t, $t \ge 0$, be a monotonically decreasing stochastic process on $\mathbb{R}$, adapted to a filtration $\mathcal{F}_t$. Assume that there are $\varepsilon, p > 0$ such that for all t with $X_t > 0$ it independently holds that $\mathrm{P}[X_t - X_{t+1} \ge \varepsilon \mid \mathcal{F}_t] \ge p$. Then the following holds for the first hitting time $T := \inf\{t \ge 0 \mid X_t \le 0\}$:*

(1) $\mathrm{E}[T \mid X_0] \le \frac{X_0}{p\varepsilon} + \frac{1}{p}$.
(2) *For all* $\delta > 0$, $\mathrm{P}[T \ge (1+\delta)\frac{X_0}{p\varepsilon}] \le \exp\left(-\frac{\delta^2}{2(1+\delta)}(\frac{X_0}{\varepsilon} - 1)\right)$.

Proof. The first statement is derived via the additive drift theorem with overshooting [18, Theorem 4].[1] We note that the monotonicity of the process implies $\mathrm{E}[X_t - X_{t+1} \mid \mathcal{F}_t] \ge p\varepsilon =: \delta'$. Moreover, thanks to the monotonicity we satisfy the assumption of an upper bound on the state space from the drift theorem. We now pessimistically limit the progress by ε; more formally, we assume $X_{t+1} \ge X_t - \epsilon$ for all $t \ge 0$. This pessimistic assumption leads to a process that is stochastically only larger than the real process and whose hitting time of state ≤ 0 is stochastically at least as large as in the real process. Hence, by our assumption we have $\mathrm{E}[X_T \mid X_0] \ge -\varepsilon$. Now, the drift theorem gives $\mathrm{E}[T \mid X_0] \le \frac{X_0 - \mathrm{E}[X_T \mid X_0]}{\delta'} \le \frac{X_0 + \varepsilon}{p\varepsilon} = \frac{X_0}{p\varepsilon} + \frac{1}{p}$.

For the second statement, we call a step t a success when $X_t - X_{t+1} \ge \varepsilon$. By assumption, the waiting time for a success dominates a geometric distribution with parameter p. Using the monotonicity of the process, $\frac{X_0}{\varepsilon}$ successes are sufficient to decrease the state of the process from X_0 to at most 0. Applying Chernoff bounds for sums of geometrically distributed random variables (Theorem 1.10.32 in [16]), setting $n = X_0/\varepsilon$), the claim follows. □

Potential Function. As common in analyses of the cGA on OneMax-like functions (e. g., [17, 46]), we often consider the stochastic process described by the so-called *potential* $\varphi_t := \sum_{i=1}^{n}(1 - p_{t,i})$, which is the sum of the probabilities of sampling 0. In the case of OneMax, it indicates the distance of the current vector to the optimal vector choosing $1 - 1/((\lambda-1)n)$ in every component. Moreover, $n - \varphi_t$ is the expected number of one-bits sampled in the offspring at time t. We also define $\tilde{\varphi}_{t+1}$ to be the pseudo-potential summing

[1] We note that there seems to be a minor typo in the original source, which states $\mathrm{E}[T \mid X_0] \le \frac{\mathrm{E}[X_T \mid X_0] - X_0}{\delta'}$ despite the process drifting to smaller values. We have flipped the sign of the right-hand side to correct this.

up all $1-\overline{p}_{t+1,i}$ after line 11 of the cGA$_\lambda$, i. e., before potentially capping them at the borders $\frac{1}{(\lambda-1)n}$ and $1-\frac{1}{(\lambda-1)n}$. In particular, if K is small, terms in this sum may be outside $[0,1]$.

Notation. By $\log(x)$ we denote the binary logarithm of x.

3 Lower Bound

In this section, we show our general lower bound on the runtime of the cGA$_\lambda$ that holds with overwhelming probability. It applies to all functions with a unique optimum and is based on relatively standard applications of Chernoff bounds to bound the progress of the probabilistic model.

Theorem 9. *Let $\lambda = poly(n)$ and consider the cGA$_\lambda$ on any function with a unique global optimum. Then with probability at least $1-2^{-\Omega(n^{1/6})}$, the number of iterations to sample the global optimum is $\Omega(K\sqrt{n/\log\lambda})$.*

Proof. By symmetry, w. l. o. g. we assume that the all-ones string is the global optimum. The main idea is to show that with overwhelming probability, it takes $\Omega(K\sqrt{n/\log\lambda})$ iterations until the potential for the first time becomes less than $n/4$. If the potential is at least $n/4$, then, by Lemma 3, the probability of sampling the optimum, i. e., the all-ones string, is at most $((n-\varphi_t)/n) \le (3/4)^n$. Even after multiplying with the number of offspring $\lambda = poly(n)$ and any polynomial number of iterations, this exponentially small failure term will decrease our final time bound only by $2^{-\Omega(n)}$.

Given a current potential φ_t, we now analyze its drift. Let $x_1,\dots,x_\lambda$ be the λ samples generated in the cGA$_\lambda$ before sorting according to fitness and $a_{\max}$ and $a_{\min}$ be their maximum and minimum number of ones (noting that the fitness function may be different from OneMax). Let X be the number of ones in an arbitrary sample, i. e., $\mathrm{E}[X] = n-\varphi_t$.

For $a_{\max} - a_{\min} \ge d$ to occur it is necessary that $a_{\max} \ge \mathrm{E}[X] + d/2$ or $a_{\min} \le \mathrm{E}[X] - d/2$. Set $d(i) := i\sqrt{n\ln\lambda}$ for $i \ge 1$. We use the following additive Chernoff bound (e. g., [16]): for all $d \ge 0$, $\mathrm{P}[X \ge \mathrm{E}[X] + d] \le e^{-d^2/n}$. Then, $\mathrm{P}[X \ge \mathrm{E}[X] + d(i)/2] \le \lambda^{-i^2/4}$. Hence, $\mathrm{P}[a_{\max} \ge \mathrm{E}[X] + d(i)/2] \le \lambda^{-i^2/4+1}$ by a union bound over the λ offspring, and by a symmetrical argument, the same holds for $\mathrm{P}[a_{\min} \le \mathrm{E}[X] - d(i)/2]$. By another union bound,

$$\mathrm{P}[a_{\max} - a_{\min} \ge d(i)] \le 2\lambda^{-i^2/4+1}. \tag{1}$$

We are now ready to compute a bound on the expected difference between $a_{\max}$ and $a_{\min}$, which will be the basis for the drift estimate of the potential. Let $D = a_{\max} - a_{\min}$. Since D only takes non-negative integer values, we have

$$\mathrm{E}[D] = \sum_{t=1}^{\infty} \mathrm{P}[D \ge t].$$

To bound the terms of the last sum, we use (1) and note that $i^2/4 - 1 \ge i^2/8$ for $i \ge 3$. Hence, for $i \ge 3$ we have $\mathrm{P}[D \ge d(i)] \le 2\lambda^{-i^2/8}$. Moreover, we use the trivial estimate $\mathrm{P}[D \ge t] \le 1$ for $t < d(3)$. The remaining terms will be bounded in blocks of $d(1) = \sqrt{n\ln\lambda}$ terms, more precisely, we bound $\mathrm{P}[D \ge t] \le \mathrm{P}[D \ge d(1)\lfloor t/d(1)\rfloor]$ for $t \ge d(3)$ and obtain $d(1)$ terms of the kind $\mathrm{P}[D \ge d(i)]$ for any $i \ge 3$. Hence, we have for any current frequency vector p_t that

$$\begin{aligned}
\mathrm{E}[D \mid p_t] &\le d(3) + \sum_{t=d(3)}^{\infty} \mathrm{P}[D \ge d(1)\lfloor t/d(1)\rfloor] \\
&= d(3) + \sum_{j=3}^{\infty} d(1)\cdot \mathrm{P}[D \ge d(j)] \\
&\le 3\sqrt{n\ln\lambda} + d(1)\sum_{j=3}^{\infty} 2\lambda^{-j^2/8} \\
&\le 3\sqrt{n\ln\lambda} + 2\sqrt{n\ln\lambda}\sum_{j=3}^{\infty} 2^{-j^2/8} = O\Big(\sqrt{n\log\lambda}\Big).
\end{aligned}$$

Finally, we apply Lemma 2. Here we pessimistically assume that $|x^{\max}|_1 = a_{\max}$ and $|x^{\min}|_1 = a_{\min}$ since this is the case where the change of potential is largest. Now the lemma yields

$$\mathrm{E}[\varphi_t - \varphi_{t+1} \mid p_t] \le \frac{\mathrm{E}[D \mid p_t]}{K} + \frac{\mathrm{E}[\mathrm{Bin}(n, 2/n)]}{K} \le \frac{O(\sqrt{n\log\lambda})}{K} + \frac{2}{K}.$$

Hence, using $\lambda = poly(n)$, we finally have

$$\mathrm{E}[\varphi_t - \varphi_{t+1} \mid p_t] = O\Big(\sqrt{n\log\lambda}/K\Big).$$

Since we would like to obtain a high-probability statement, we will not use a drift theorem directly on the previous expectation but will apply concentration inequalities. Let c be the implicit constant from the previous bound, i. e., in expectation the potential decreases by at most $\delta^* := c\sqrt{n\log\lambda}/K$ per iteration. We consider the phase consisting of the first $T^* := n/(8\delta^*) = \frac{K}{8c}\sqrt{n/\log\lambda}$ iterations after initialization and let $\delta_t = \varphi_t - \varphi_{t+1}$ be the random decrease in potential in iteration $t \in \{1,\dots,T^*\}$. For $\Delta := \sum_{t=1}^{T^*}\delta_t$ it holds that $\mathrm{E}[\Delta] \le T^*\delta^* \le n/8$. Now, we use (1) on the difference $D/K = (a_{\max} - a_{\min})/K$ and a Chernoff bound on the error $F \sim \mathrm{Bin}(n, 2/n)/K$ introduced by capping, see Lemma 2. For both quantities, we estimate the probability that they exceed $n^{2/3}/K$. Along with a union bound, we obtain for any $t \in \{1,\dots,T^*\}$ that

$$\begin{aligned}
\mathrm{P}[\delta_t \ge 2n^{2/3}/K] &\le \mathrm{P}[D \ge d(n^{1/6}/\sqrt{\ln\lambda})] + \mathrm{P}[F \ge n^{2/3}/K] \\
&= e^{-\Omega(n^{1/3})} + e^{-\Omega(n^{2/3})} = e^{-\Omega(n^{1/3})}.
\end{aligned}$$

By a union bound, we have $\delta_t \le n^{2/3}/K$ for all $t \in \{1,\dots,T^*\}$ with probability $1-T^*e^{-\Omega(n^{1/3})} = 1-e^{-\Omega(n^{1/3})}$, which we assume to happen. Now, applying an additive Hoeffding bound (Theorem 1.10.9 in [16]) on Δ with a range of size $2n^{2/3}/K$ for the δ_t, we have

$$\begin{aligned}
\mathrm{P}[\Delta \ge n/4] \le \mathrm{P}[\Delta \ge \mathrm{E}[\Delta] + n/8] &\le e^{-2(n/8)^8/(T^*(4n^{4/3}/K^2))} \\
&= e^{-\Omega(Kn^2/(\sqrt{n}n^{4/3}))} = e^{-\Omega(Kn^{1/6})}.
\end{aligned}$$

Hence, up to failure events of total probability $e^{-\Omega(n^{1/6})}$, the potential after the first T^* iterations will be at least $n/2 - n/4 = n/4$. Together with a union bound on the probability of sampling the optimum in these iterations (as analyzed in the first paragraph of this proof), with probability at least $1-e^{-\Omega(n^{1/6})}$ the optimum will not be sampled in the first $T^* = \Omega(K\sqrt{n/\log\lambda})$ iterations. □

4 Upper Bound

We show that the lower bound from Theorem 9 is asymptotically tight by proving a matching upper bound. As usual in upper bounds for the cGA (and similarly for other EDAs), it also involves a minimum value for its parameter K to control genetic drift. Using the minimum viable K, the time bound in iterations will be by an asymptotic factor of $\sqrt{\log \lambda}$ smaller than with the classical cGA (i. e., $\lambda = 2$). As mentioned above, the number of fitness function evaluations, which is the classical cost measure for "runtime", is by a factor of λ bigger.

Theorem 10. *Let $K \geq c\sqrt{n/\log \lambda}\log n$ for sufficiently large constant $c > 0$ and assume $\lambda = poly(n)$. Then the number of iterations of the cGA$_\lambda$ to optimize OneMax is $O(K\sqrt{n/\log \lambda})$ with high probability. This is $O(n\log n/\log \lambda)$ for the smallest choice of K covered by this theorem.*

The following lemma is at the heart of the proof. Recall that $\tilde{\varphi}_{t+1}$ is the pseudo-potential summing up all $1-\overline{p}_{t+1,i}$ before potentially capping them at their borders. The potential after capping terms of φ_{t+1} may be slightly larger as analyzed later in the proof of Theorem 10 – due to an assumption of all frequencies being at least $1/4$, capping will never decrease the potential.

Lemma 11. *Consider the cGA$_\lambda$ on OneMax. Let $\varphi_t \geq 40$ and assume $p_i^{(t)} \geq 1/4$ for all $i \in \{1,\dots,n\}$. Then with probability at least $1/953$ it holds that $\varphi_t - \tilde{\varphi}_{t+1} \geq \sqrt{(1/10)\varphi_t(\ln\lambda - \ln\ln\lambda)}/K$.*

Delaying the proof of the lemma, we give the proof of the upper bound theorem.

Proof of Theorem 10: We consider a period of $T = \kappa K\sqrt{n/\log\lambda}$ steps, where $K \geq c\sqrt{n/\log\lambda}\log n$ for a sufficiently large constant $c > 0$ and $\kappa > 0$ is another sufficiently large constant that does not depend on c. The first step is to limit genetic drift. Choosing $\gamma = 1/4$ in Lemma 1, the probability of any fixed frequency ever dropping below $1/4$ in T steps is at most

$$2e^{-\frac{K}{16\kappa\sqrt{n/\log\lambda}}} \leq 2e^{-\frac{c}{16\kappa}\log n}.$$

If we choose c large enough, the probability is at most $n^{-c/(16\kappa)}$, and by a union bound, the probability of at least one frequency dropping below $1/4$ is still at most $n^{-c/(16\kappa)+1}$. So all frequencies will stay at $1/4$ or higher with high probability, which we assume in the following. We next show that the optimum is sampled in T steps with high probability.

We will apply Lemma 11 over phases exhibiting similar drift to estimate the accumulated decrease of potential. We do so by summing up the differences between the current potential φ_t and the pseudo-potential $\tilde{\varphi}_{t+1}$ over the steps of the phase, assuming for the moment that $\varphi_{t+1} = \tilde{\varphi}_{t+1}$. More formally, we consider a process X_t, $t \geq 0$, where X_0 is the φ-value at the beginning of the phase and $X_{t+1} = X_t - (\varphi_t - \tilde{\varphi}_{t+1})$, bounding φ_t in Lemma 11 by the smallest φ-value in the phase. Since the objective function is OneMax, we have $|x^{\max}|_1 \geq |x^{\min}|_1$ in Algorithm 1 and hence this X_t-process is monotonically decreasing. After analyzing the accumulated progress over all phases, we will apply Lemma 2 separately to estimate the accumulated loss due to capping frequencies at their borders. Such a separation into a monotonic process and an error estimate of the boundary effects was also done in [17], albeit at the level of individual phases.

We now describe the phases used in our drift analysis in detail. Phase 1 starts with initialization, where $X_0 = \varphi_0 = n/2$. Phase i, where $i \geq 1$, ends before the first time when $X_t \leq n/2^{i+1}$ and Phase $i+1$ starts in this iteration. Recall that the X_t-process is monotonically decreasing the whole time. We stop the consideration at the point where the X_t-value drops below a value of φ_{stop}, where φ_{stop} is bounded from above by a constant analyzed below and $\varphi_{\text{stop}} \geq 40$ as required by Lemma 11. Hence, we consider at most $\log n$ phases. Such divisions of processes with variable drift into phases of roughly identical drift but exponentially decreasing length like $n/2^i$ were frequently used before the introduction of a variable drift theorem, see, e. g., the analysis of a simple MMAS$_{\text{ib}}$ ant colony optimizer on OneMax [41]. Similarly, such a division into exponentially decreasing length appears in the aforementioned upper bound for the usual cGA on jump functions [17].

According to the lemma, each step of Phase i with probability at least $1/953$ decreases the potential by at least

$$\varepsilon := \frac{\sqrt{(1/10)(n/2^{i+1})(\ln\lambda - \ln\ln\lambda)}}{K}$$

The total amount by which the potential has to decrease to start the next phase is at most $n/2^i$ – note that a monotonic process is considered. We apply the 2nd statement of Theorem 8 with $X_0 = n/2^i$, ε as above and $p = 1/953$. Setting $\delta := 1$ for $1 \leq i \leq (\log n)/2$ and otherwise $\delta := c'\log n$ for an arbitrary constant c', the probability of not finishing the phase in at most

$$\frac{2\cdot 953Kn}{\sqrt{(1/10)(n/2^{i+1})(\ln\lambda - \ln\ln\lambda)}\cdot 2^i} = O\left(K\sqrt{n/(2^i\log\lambda)}\right)$$

steps is at most

$$\exp\left(-\Omega\left(K\sqrt{n/(2^i\log\lambda)}\right)\right) = \exp(-\Omega(K))$$

for $i \leq (\log n)/2$. For $i > (\log n)/2$ and $i \leq \log n$, the upper bound on the phase length is $O(Kn^{1/4}\delta/\sqrt{\log\lambda}) = O(Kn^{1/4}\log n/\sqrt{\log\lambda})$ and the failure probability is at most $\exp(-\Omega(K\delta/\sqrt{\log\lambda}))$. Hence, assuming the constant c' in δ large enough, the bound on the failure probability is still at most $\exp(-\Omega(K))$. Altogether, by a simple union bound, all phases are bounded as desired with high probability. The accumulated length of all phases in this case is

$$\sum_{i=1}^{(\log n)/2} O\left(K\sqrt{n/(2^i\log\lambda)}\right) + O(\log n)\cdot O(Kn^{1/4}/\sqrt{\log\lambda}),$$

which is $O(K\sqrt{n/\log\lambda}) \leq T/2$ for an appropriate choice of κ.

We are left with an analysis of the error introduced by capping frequencies at their borders. We consider a total period of $T = O(K\sqrt{n/\log\lambda})$ steps, which subsumes the $T/2$ steps consisting of at most $\log n$ phases analyzed above, and a final period of at most $T/2 = O(K\sqrt{n/\log\lambda})$ steps explained below. According to Lemma 2, the total error in a period of T steps is dominated by the sum of T independent random variables following $\frac{\text{Bin}(n,2/n)}{K}$. This sum is $\text{Bin}(Tn, 2/n)/K$. By Chernoff bounds, the error is bounded by $4T/K = O(\sqrt{n/\log\lambda})$ with probability $1-e^{-\Omega(T)} = 1-e^{-\Omega(\sqrt{n/\ln\lambda})}$. We assume this error bound to hold. Moreover, we pessimistically assume the accumulated (unwanted) increase of potential by capping to happen when it is smallest, i. e., when progress is most difficult. This is the case then $\varphi_t = \varphi_{\text{stop}}$, where Lemma 11 gives a progress (decrease) of potential (before capping) that with probability at least $1/953$ is at least

$$\frac{\sqrt{(\varphi_{\text{stop}}/10)(\ln\lambda - \ln\ln\lambda)}}{K} \ge \frac{\sqrt{(\varphi_{\text{stop}}/20)\ln\lambda}}{K} \ge \frac{\sqrt{\varphi_{\text{stop}}/29}}{K},$$

using $\ln\lambda - \ln\ln\lambda \ge (\ln\lambda)/2$ and $\ln\lambda \ge 0.69$ for $\lambda \ge 2$. Hence, using Theorem 8 as above and again ignoring the capping errors, another period of length $2 \cdot 953\sqrt{29/\varphi_{\text{stop}}} \cdot (4\kappa)K\sqrt{n/\log\lambda}$ is sufficient with probability $1 - 2^{-\Omega(K)}$ to decrease the potential by at least $4\kappa\sqrt{n/\log\lambda}$, where κ is the constant from our definition of T. Note that this final period will be at most $T/2 = (\kappa/2)K\sqrt{n/\log\lambda}$ long if φ_{stop} is large enough. Hence, it is included in the total number of T steps, so the possible error occurring in it is already accounted for in the above error bound $4T/K$. By definition, we have $4T/K \le 4\kappa\sqrt{n/\log\lambda}$. Since we considered a total decrease of potential (before capping) of at least $n + 4\kappa\sqrt{n/\log\lambda}$, the actual decrease will be at least n. Note that these bounds n and $n + 4\kappa\sqrt{n/\log\lambda}$ are pessimistic since we stop the analysis prematurely when $\varphi_t \le \varphi_{\text{stop}}$.

Once $\varphi_t = O(1)$, we claim that there is a constant probability of the cGA$_\lambda$ sampling the optimum. This holds by Lemma 4, noting that the expected number of ones per sampled offspring is $n - \varphi_t$, so the probability of sampling all ones is minimized if as many frequencies as possible take their extreme values $1 - \frac{1}{(\lambda-1)n}$ and $1/4$. Even if $\lceil (4/3)\varphi_t \rceil$ frequencies are at $1/4$, the probability of sampling only ones is at least $\left(1 - \frac{1}{(\lambda-1)n}\right)^n (1/4)^{(4/3)\varphi_t + 1} = \Omega(1)$. See the proof of Lemma 11 for a similar use of Lemma 4.

After $\varphi_t \le \varphi_{\text{stop}}$, we consider a final phase of $\max\{1, (c\log n)/\lambda\}$ iterations and analyze the event of sampling the optimum in it. If each iteration of this phase satisfies $\varphi_t = O(1)$, implying a constant probability of sampling the optimum, then the probability of not sampling it in the phase is $(1 - \Omega(1))^{c\log n} = n^{-\Omega(c)}$ since each iteration takes λ samples. We now show that the potential and therefore the probability of sampling the optimum stays constant in the phase with high probability. The aim is to apply the additive Chernoff bound depending on the variance $\mathrm{P}[X \ge \mathrm{E}[X] + d] \le e^{-d^2/(2\operatorname{Var}(X) + 2d/3)}$, which, in particular, applies to the sum of n independent Bernoulli trials, see, e. g., [16, Theorem 1.10.12]. We consider the n bit values sampled in the creation of an offspring of the cGA$_\lambda$ as the trials and let X_k be the number of bits that are sampled as 0 in the k-th offspring at iteration t. The potential increases by at most $\max\{X_k \mid k = 1, \dots, \lambda\}/K$ in the iteration since each position must be sampled with both possible bit values in $x^{\max}$ and $x^{\min}$ to increase the corresponding frequency. Clearly, $\mathrm{E}[X_k] = \varphi_t$ and $\operatorname{Var}(X_k) = \sum_{i=1}^{n} p_{t,i}(1 - p_{t,i}) \le \varphi_t$. Applying the Chernoff bound with $d = \log n$ and $\operatorname{Var}(X) = O(1)$ (implicitly using the largest variance observed throughout the phase, which we show to be $O(1)$ with high probability), we have $\varphi_{t+1} - \varphi_t \le (\log n)/K$ with probability $1 - \lambda 2^{-\Omega(\log^2 n)} = 1 - n^{-\omega(1)}$, using a union bound over the λ offspring. Recalling that $K = \Omega(\sqrt{n/\log\lambda}\log n)$, the total increase is bounded by

$$\max\left\{1, \frac{c\log n}{\lambda}\right\} O((\log n)/K) = O\Big((\log^2 n)/\sqrt{n/\log\lambda}\Big) = O(1)$$

with high probability. □

We now give the delayed proof of the main lemma.

Proof of Lemma 11: Denote $f(x) = \textsc{OneMax}(x)$ and let $f_{\max}$ and $f_{\min}$ be the maximum and minimum order statistics, respectively, of the OneMax-value of the λ offspring generated at time t, i. e., after sorting the offspring by decreasing OneMax-value. We show that with probability at least $1/953$, it holds that $f_{\max} - f_{\min} \ge \sqrt{(1/10)\varphi_t(\ln\lambda - \ln\ln\lambda)}$. To that end, we split the λ created offspring into 2 sets $\Lambda_1 = \{x_1, \dots, x_{\lfloor\lambda/2\rfloor}\}$, $\Lambda_2 = \{x_{\lfloor\lambda/2\rfloor+1}, \dots, x_\lambda\}$ containing the first $\lfloor\lambda/2\rfloor$ created and $\lceil\lambda/2\rceil$ last created offspring, respectively, before they are sorted. Let $s = \varphi_t \ge 40$. We claim:

(1) With probability at least $1/4$, it holds that $\min_{x\in\Lambda_1} f(x) \le n - s$.
(2) With probability at least 0.0042, it holds that $\max_{x\in\Lambda_2} f(x) \ge n - s + \sqrt{(1/10)s(\ln\lambda - \ln\ln\lambda)}$.

Clearly, since the two subpopulations are created independently, the two claims together give an event of joint probability $0.0042/4 \ge 1/953$. This implies the lemma since $\varphi_t - \tilde{\varphi}_{t+1} = (f_{\max} - f_{\min})/K$, recalling that $\tilde{\varphi}_{t+1}$ sums up the complementary frequencies before capping. We now give the proofs of the two claims.

For the first claim, we only consider x_1 and use Hoeffding's result given in Lemma 3, part 2, to relate the Poisson-binomial distribution to a binomial distribution. Let $X \sim \operatorname{Bin}(n, (n-s)/n)$, i. e., $\mathrm{E}[X] = \mathrm{E}[n - \varphi_t] = n - s$. The lemma shows the binomial distribution is the "most spread out" Poisson-binomial distribution, i. e., $\mathrm{P}[f(x_1) \le n - s] \ge \mathrm{P}[X \le n - s]$. If we can show that $\mathrm{P}[X \le \mathrm{E}[X]] \ge 1/4$, the claim follows. To this end, note that $Y := n - X$ follows a binomial distribution with parameters n and s/n. Since $s > 1$, Lemma 7 shows $\mathrm{P}[Y \ge \mathrm{E}[Y]] \ge 1/4$, which is equivalent to $\mathrm{P}[X \le \mathrm{E}[X]] \ge 1/4$.

For the second claim, we let $v(\lambda) := \sqrt{(1/10)s(\ln\lambda - \ln\ln\lambda)}$ and prove for any $x \in \Lambda_2$ the property $\mathrm{P}[f(x) \ge n - s + v(\lambda)] \ge 0.0086/\lambda$. Hence, by the independent creation of the offspring, there exists at least one such $x \in \Lambda_2$ with probability at least $1 - (1 - 0.0086/\lambda)^{\lambda/2} \ge 1 - e^{-0.0086/2} \ge 0.0042$.

To show the property for an arbitrary $x \in \Lambda_2$, we first note that $v(\lambda) \ge 2$ since $s \ge 40$ and $\lambda \ge 2$. Hence, applying the Schur-convexity arguments from [28] (see 2nd statement of Lemma 4), the probability of the event $f(x) \ge n - s + v(\lambda)$ is minimized if as many frequencies as possible take the extreme values $1 - 1/((\lambda-1)n)$ and $1/4$ while preserving their sum $n - s/n$. This gives at least $\lfloor (4/3)(s-1) \rfloor =: s'$ frequencies of value $1/4$, at most $n - 1$ frequencies of value $1 - \frac{1}{(\lambda-1)n}$ and at most one frequency taking a third value in $(1/4, 1 - \frac{1}{(\lambda-1)n})$. The probability of setting all bits belonging to the second and the third groups to 1 is at least $(1 - 1/n)^{n-1}/4 \ge e^{-1}/4$. For the first group, we are confronted with a binomially distributed random variable X with s' trials and success probability $p = 1/4$; its variance equals

$$\operatorname{Var}(X) = s'p(1-p) = s' \cdot \frac{1}{4} \cdot \frac{3}{4} \ge \left(\frac{4}{3}(s-1) - 1\right)\frac{3}{16} \ge \frac{s}{5}$$

since $s \ge 40$. We now estimate the probability of having at least $k := \lceil s'/4 + v(\lambda) \rceil$ successes in the given binomial distribution. Since

$v(\lambda) \geq 2$, we have $k \leq s'/4 + 2v(\lambda)$. Hence, by Slud's inequality (Lemma 5), the probability $\mathrm{P}[X \geq k]$ of at least k successes in the first group is at least

$$1 - \Phi\left(\frac{k - s'/4}{\sqrt{\mathrm{Var}(X)}}\right) \geq 1 - \Phi\left(\frac{2v(\lambda)}{\sqrt{s/5}}\right)$$
$$= 1 - \Phi(\sqrt{2(\ln \lambda - \ln\ln \lambda)}),$$

where $\Phi(x)$ is the cumulative distribution function of the standard normal distribution. Now, using Lemma 12 below, this probability is at least $0.094/\lambda$. Combining with the bound $e^{-1}/4$ for the probability of setting all bits of other frequencies to 1, we obtain a joint probability of at least $0.0086/\lambda$ as demanded by the property. □

We give the remaining proof of a bound of a particular Φ-value of the standard normal distribution as used in the preceding lemma.

Lemma 12. *Let $\lambda \geq 2$ and $x(\lambda) = \sqrt{2(\ln \lambda - \ln\ln \lambda)}$. Then for the cumulative distribution function of the standard normal distribution it holds that $1 - \Phi(x(\lambda)) \geq 0.094/\lambda$.*

Proof. By standard arguments from calculus, we obtain that $\ln \lambda - \ln\ln \lambda$ takes its minimum value 1 for $\lambda = e$ in the domain $[2, \infty)$. In particular, $x(\lambda) \geq \sqrt{2}$, so we can apply the lower bound

$$1 - \Phi(x) \geq \left(\frac{1}{x} - \frac{1}{x^3}\right) \frac{1}{\sqrt{2\pi}} e^{-x^2/2}.$$

for $x \geq 1$ from Lemma 6. Plugging in $x(\lambda)$, we obtain

$$1 - \Phi(x) \geq \left(\frac{1}{(2\ln \lambda - 2\ln\ln \lambda)^{1/2}} - \frac{1}{(2\ln \lambda - 2\ln\ln \lambda)^{3/2}}\right) \frac{\ln \lambda}{\sqrt{2\pi}\lambda}.$$

We bound the last expression by analyzing

$$f(\lambda) = (\ln \lambda)\left(\frac{1}{(2\ln \lambda - 2\ln\ln \lambda)^{1/2}} - \frac{1}{(2\ln \lambda - 2\ln\ln \lambda)^{3/2}}\right).$$

First of all, by the above lower bound on $x(\lambda)$, we have $(x(\lambda))^3 \geq 2x(\lambda)$ and therefore

$$f(\lambda) \geq \frac{1}{2} \frac{\ln \lambda}{(2\ln \lambda - 2\ln\ln \lambda)^{1/2}}.$$

Using $z - \ln z \leq (1 - \ln(\ln(2))/\ln 2)z \leq 1.53z$ for $z \geq \ln 2$, we have

$$f(\lambda) \geq \frac{1}{2\sqrt{2}} \frac{\ln \lambda}{\sqrt{1.53}\sqrt{\ln \lambda}} \geq \frac{\sqrt{\ln \lambda}}{2\sqrt{3.06}} \geq \frac{\sqrt{\ln 2}}{2\sqrt{3.06}}.$$

Substituting this bound in our initial estimate, we have

$$1 - \Phi(x) \geq \frac{\sqrt{\ln 2}}{2\sqrt{3.06}\sqrt{2\pi}} \frac{1}{\lambda} \geq \frac{0.094}{\lambda}.$$

□

As mentioned in the introduction, Theorem 10 recovers the well-known $O(K\sqrt{n})$ bound, holding with high probability for $K \geq c\sqrt{n}\log n$ in the case $\lambda = 2$ [46] – however, in contrast to that paper, we do not bound the expected runtime here. We find that the proof of Theorem 10 uses simpler arguments than previous work since it does not inspect the stochastic process of single frequencies and does not use complicated moment-generating functions to obtain a high-probability statement.

5 A Function where Large Offspring Sizes May Be Beneficial

The results from the previous sections show a minor speedup for the cGA$_\lambda$ on OneMax, measured in the number of iterations. This speedup is only of order $\Theta(\log \lambda)$, while for simple $(1\overset{+}{,}\lambda)$ EAs the speedup in the number of iterations can be $\Theta(\lambda)$ [34]. However, in terms of the number of function evaluations, there is no real speedup, so the results are only relevant for models of computation where the offspring population can be evaluated in parallel, at least to some extent.

We are therefore interested in finding a multimodal example where larger selection pressure through an offspring population of the cGA$_\lambda$ is crucial and may even help to reduce the actual number of fitness evaluations. Such example is discussed in the following. It is based on the observation made above that the cGA$_\lambda$ may sample with a variance that is by a factor $\Theta(\log \lambda)$ larger than with the standard choice $\lambda = 2$. If larger variance allows the algorithm to find a trajectory towards the global optimum, while the standard variance is unlikely to find this trajectory and leads the algorithm towards a deceptive local optimum, a drastic runtime difference can be likely.

Consider the following function $f\colon \{0,1\}^n \to \mathbb{R}$ with parameter $m > 0$:

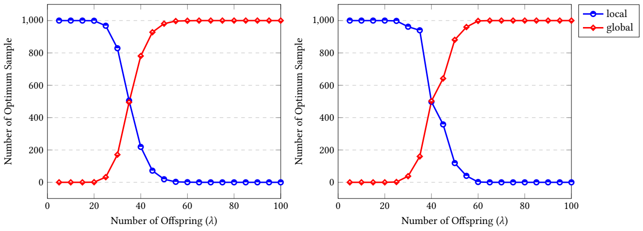
$$f_{\text{sampling}}(x) := \begin{cases} |x|_1 & \text{if } |x|_1 \geq n/2 + m, \\ n/2 + m - 1 - |x|_1 & \text{otherwise.} \end{cases}$$

This function has a global optimum in the all-ones string and a local optimum in all-zeros. It is similar to bimodal Trap [14] and to TwoMax functions studied previously in the literature (e. g., [27, 45]). However, the trajectory towards the global optimum is only visible if search points are sampled whose number of ones is by an additive term of m larger than the initial expected value of $n/2$. We conjecture the following: If we set m in the order of $\Theta(\sqrt{n\log \lambda})$, then the cGA$_\lambda$ will have a constant probability of sampling on the trajectory towards the global optimum, while the classic cGA will miss this trajectory with probability $1 - o(1)$ and quickly adjust its model towards the all-zeros string, the deceptive local optimum. In Section 6, we give experimental evidence for this conjecture.

We remark that on f_{sampling}, the probabilistic decision between the successful trajectory and the unsuccessful one is made right after initialization, so essentially through uniform random samples. It does not seem hard to define a function where the algorithm has to evolve its model for some time before a region is reached where the trajectory is split and only a larger sample size directs the cGA$_\lambda$ on the right path with high probability.

6 Empirical Analysis

We begin our empirical analysis by examining the performance of the cGA$_\lambda$ on the deceptive function f_{sampling} introduced in the previous section. The key question is whether a larger offspring size λ helps the algorithm find the trajectory to the global optimum and avoid following the trajectory to the local one. Figure 1 shows the number of runs (out of 1000) in which the cGA$_\lambda$ converged to either the local or the global optimum, depending on the offspring population size λ. In this experiment, we fixed n=100 and 200, K=100, and m=10 and 15. The results clearly demonstrate a phase transition:



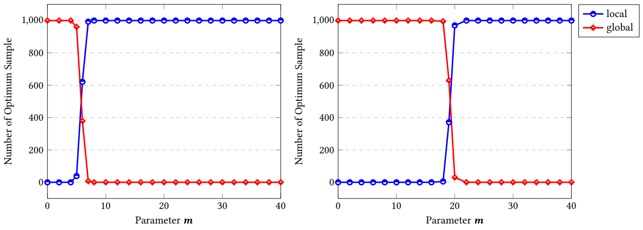
(a) m**=10**

(b) m**=15**

Figure 1: Number of Local optimum vs Global optimum out of 1000 runs; for n**=100 and** $m = 10$ **(left); for** n**=200 and** $m = 15$ **(right);** $K = 100$**,** $\lambda \in \{5, 10, \ldots, 100\}$**.**

for small λ, the algorithm almost always converges to the local optimum. As λ increases, the probability of reaching the global optimum increases rapidly. Beyond a certain threshold of λ, the success rate of global optimum saturates at almost 100%. These findings indicate that a sufficiently large λ improves the algorithm's ability to identify and follow the trajectory toward the global optimum.

Figure 2 further highlights this behavior by varying the parameter m, which controls the deceptiveness of the above function. With a fixed small $\lambda = 2$, the algorithm reliably finds the global optimum only for small values of m. As m increases, all runs end in the local optimum, confirming our conjecture that small offspring sizes are insufficient to overcome the deceptive structure of the function. For a large offspring population of $\lambda = 200$, the algorithm finds the global optimum not only for small values of m, but also for significantly larger ones. These results provide strong empirical support for the theoretical conjecture in Section 5.



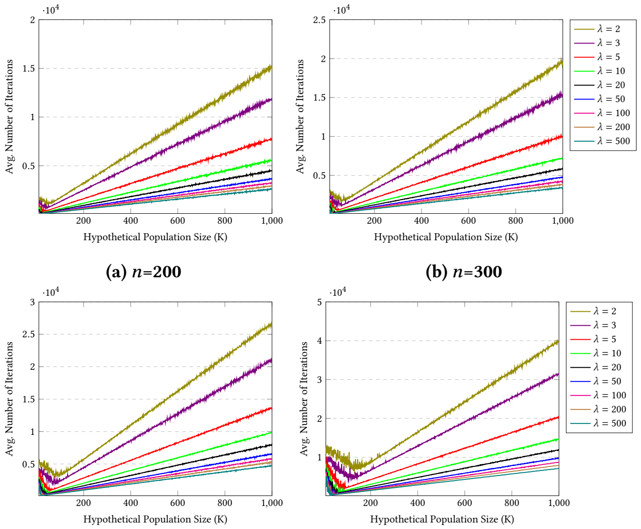
(a) λ**=2**

(b) λ**=200**

Figure 2: Number of Local optimum vs Global optimum out of 1000 runs; for λ**=2 and 200;** $n = 200$**,** $K = 100$**, and** $m = \{0, 2, \ldots, 40\}$**.**

We now present the results of experiments that evaluate the performance of cGA$_\lambda$ on the classical OneMax benchmark function. The goal of these experiments is to quantify the effect of the offspring population size λ on the average number of iterations required to reach the global optimum, and to compare this with the theoretical prediction of a $\Theta(\log \lambda)$ speedup in terms of iterations.

To evaluate the effect of λ offspring, we conducted simulations using different problem sizes. For each problem size, we tested a range of hypothetical population sizes K and varied the offspring population size λ between values $\lambda = 2$, 3, 5, 10, 50, 100, 200 and 500. Each experiment was repeated in multiples of 200 times to ensure statistical robustness, and we measured the average number of iterations required for the algorithm to reach the global optimum.



(a) n**=200**

(b) n**=300**

(c) n**=500**

(d) n**=1000**

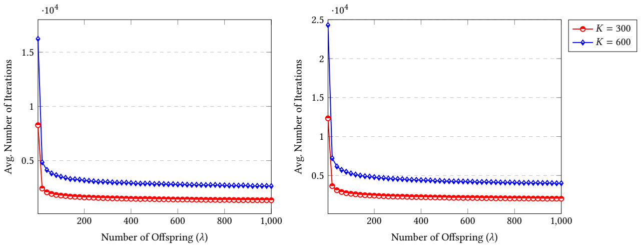
Figure 3: Empirical runtime of the cGA$_\lambda$ **on OneMax; for** n**=200, 300, 500 and 1000;** $\lambda \in \{2, 3, 5, 10, 20, 50, 100, 200, 500\}$**,** $K \in \{10, \ldots, 1000\}$ **and averaged over 200 runs.**

Figure 3 summarizes the results for $n = 200$, 300, 500 and 1000. The x-axis represents the hypothetical population size K and the y-axis shows the average number of iterations. Each curve corresponds to a different λ value.

Across all plots, we observe a consistent trend: larger offspring sizes lead to fewer iterations required to find the optimum. However, the benefits of increasing λ are not uniform throughout the range. Significant gains are observed when increasing λ from smaller values, for example, from $\lambda = 2$ to 5 or 10 where the iteration count decreases notably. In contrast, for larger values such as $\lambda = 100$ to 200, the benefit of increasing λ becomes marginal. This trend is further supported by Figure 4, which plots the average number of iterations required to reach the optimum against the number of offspring λ, for fixed K. The figures show that increasing λ initially yields substantial reductions in the iteration count; however, the rate of improvement decreases as λ becomes larger. This is in line with the factor $\Theta(1/\sqrt{\log \lambda})$ from the theoretical analysis.

In addition to measuring the number of iterations, we also analyzed the performance of the algorithm in terms of the total number of function evaluations, calculated as the product of the number of iterations and the size of the offspring population λ. Figure 5 illustrates the average number of function evaluations required to reach the optimum for $n = 200$ and 500, plotted against varying values of the hypothetical population size K.

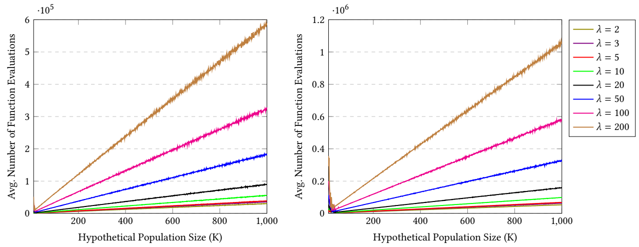
Unlike the results based on the number of iterations in Figure 3, where increasing λ led to faster convergence, the curves in Figure 5 show an opposite pattern. Here, increasing λ actually results in more function evaluations. This highlights an important trade-off: using more offspring can reduce the number of iterations needed, but they incur a higher computational cost when offspring are evaluated sequentially.



(a) n=500

(b) n=1000

Figure 4: Empirical runtime of the cGA$_\lambda$ on OneMax; for n=500 and 1000; $\lambda \in \{2, 20, 40, 60, \ldots, 1000\}$, K=300 and 600, and averaged over 200 runs.



(a) n=200

(b) n=500

Figure 5: Average number of function evaluations for the cGA$_\lambda$ on OneMax; for n=200 and 500; $\lambda \in \{2, 3, 5, 10, 20, 50, 100, 200\}$, $K \in \{10, \ldots, 1000\}$ and averaged over 200 runs.

These results confirm that there is no real speedup in terms of function evaluations. However, cGA$_\lambda$ offers practical advantages in environments where multiple offspring can be evaluated in parallel.

7 Conclusions

In this paper, we presented the first runtime analysis of the cGA$_\lambda$, a Compact Genetic Algorithm using a larger offspring population to increase selection pressure. Using rigorous analyses of the drift of its probabilistic model, we proved that the number of iterations required to optimize OneMax is $\Theta(K\sqrt{n/\log \lambda})$ with high probability, with the lower bound holding for all functions with a unique optimum. The proof of the upper bound includes new probabilistic tools and is arguably simpler than previous upper bounds for the case $\lambda = 2$. We also defined a multimodal function where a large offspring population size seems to be crucial, and verified this experimentally. A theoretical analysis of the cGA$_\lambda$ on this function is left for future work, as are further studies on the impact of the offspring population size on multimodal problems.

Acknowledgments

This work has been supported by the Danish Council for Independent Research through grant 10.46540/2032-00101B. The authors thank the participants and organizers of Dagstuhl seminar 25092 for facilitating discussions that supported developing the ideas discussed in this work. This work also benefited from COST Action CA22137 "Randomised Optimisation Algorithms Research Network" (ROAR-NET), supported by COST (European Cooperation in Science and Technology).

References

[1] Sumit Adak and Carsten Witt. 2024. Runtime Analysis of a Multi-valued Compact Genetic Algorithm on Generalized OneMax. In *Proc. of PPSN 2024*. Springer, 53–69.

[2] Sumit Adak and Carsten Witt. 2025. Improved Runtime Analysis of a Multi-Valued Compact Genetic Algorithm on Two Generalized OneMax Problems. In *Proc. of GECCO 2025*. ACM Press, 1585–1593.

[3] Sumit Adak and Carsten Witt. 2025. A Runtime Analysis of the Multi-valued Compact Genetic Algorithm on Generalized LeadingOnes. In *Proc. EvoCOP 2025*. Springer, 1–17.

[4] Chang Wook Ahn and Rudrapatna S. Ramakrishna. 2003. Elitism-based compact genetic algorithms. *IEEE Transactions on Evolutionary Computation* 7, 4 (2003), 367–385.

[5] Chang Wook Ahn and Rudrapatna S. Ramakrishna. 2004. Augmented Compact Genetic Algorithm. In *Proc. of Parallel Processing and Applied Mathematics (PPAM 2003)*, Vol. 3019. Springer, 560–565.

[6] Adetunji David Ajimakin and V. Susheela Devi. 2023. The Competing Genes Evolutionary Algorithm: Avoiding Genetic Drift Through Competition, Local Search, and Majority Voting. *IEEE Transactions on Evolutionary Computation* 27, 6 (2023), 1678–1689.

[7] Denis Antipov and Benjamin Doerr. 2021. A Tight Runtime Analysis for the $(\mu + \lambda)$ EA. *Algorithmica* 83, 4 (2021), 1054–1095.

[8] Béla Bollobás. 2001. *Random graphs* (2nd ed.). Number 73 in Cambridge studies in advanced mathematics. Cambridge University Press.

[9] Marcel Chwiałkowski, Benjamin Doerr, and Martin S. Krejca. 2025. Runtime Analysis of the Compact Genetic Algorithm on the LeadingOnes Benchmark. *IEEE Transactions on Evolutionary Computation* (2025).

[10] Dogan Corus, Andrei Lissovoi, Pietro S. Oliveto, and Carsten Witt. 2021. On Steady-State Evolutionary Algorithms and Selective Pressure: Why Inverse Rank-Based Allocation of Reproductive Trials Is Best. *ACM Transactions on Evolutionary Learning and Optimization* 1, 1 (2021), 1–38.

[11] Duc-Cuong Dang, Per Kristian Lehre, and Phan Trung Hai Nguyen. 2019. Level-Based Analysis of the Univariate Marginal Distribution Algorithm. *Algorithmica* 81, 2 (2019), 668–702.

[12] Duc-Cuong Dang and Per Kristian Lehre. 2015. Simplified runtime analysis of estimation of distribution algorithms. In *Proc. of GECCO 2015*. ACM Press, 513–518.

[13] Axel de Perthuis de Laillevault, Benjamin Doerr, and Carola Doerr. 2015. Money for Nothing: Speeding Up Evolutionary Algorithms Through Better Initialization. In *Proc. of GECCO 2015*. ACM Press, 815–822.

[14] Kalyanmoy Deb and David E. Goldberg. 1993. Analyzing Deception in Trap Functions. In *Proc. of FOGA 1992*. Elsevier, 93–108.

[15] Benjamin Doerr. 2018. An elementary analysis of the probability that a binomial random variable exceeds its expectation. *Statistics & Probability Letters* 139 (2018), 67–74.

[16] Benjamin Doerr. 2020. Probabilistic Tools for the Analysis of Randomized Optimization Heuristics. In *Theory of Evolutionary Computation: Recent Developments in Discrete Optimization*, Benjamin Doerr and Frank Neumann (Eds.). Springer, 1–87.

[17] Benjamin Doerr. 2021. The Runtime of the Compact Genetic Algorithm on Jump Functions. *Algorithmica* 83, 10 (2021), 3059–3107.

[18] Benjamin Doerr and Timo Kötzing. 2021. Multiplicative Up-Drift. *Algorithmica* 83, 10 (2021), 3017–3058.

[19] Benjamin Doerr and Martin S. Krejca. 2020. Significance-Based Estimation-of-Distribution Algorithms. *IEEE Transactions on Evolutionary Computation* 24, 6 (2020), 1025–1034.

[20] Benjamin Doerr and Martin S. Krejca. 2021. A simplified run time analysis of the univariate marginal distribution algorithm on LeadingOnes. *Theoretical Computer Science* 851 (2021), 121–128.

[21] Benjamin Doerr and Martin S. Krejca. 2021. The Univariate Marginal Distribution Algorithm Copes Well with Deception and Epistasis. *Evolutionary Computation* 29, 4 (2021), 543–563.

[22] Benjamin Doerr and Martin S. Krejca. 2023. Bivariate estimation-of-distribution algorithms can find an exponential number of optima. *Theoretical Computer Science* 971 (2023), 114074.

[23] Benjamin Doerr and Weijie Zheng. 2020. Sharp Bounds for Genetic Drift in Estimation of Distribution Algorithms. *IEEE Transactions on Evolutionary Computation* 24, 6 (2020), 1140–1149.

[24] Stefan Droste. 2006. A rigorous analysis of the compact genetic algorithm for linear functions. *Natural Computing* 5 (2006), 257–283.

[25] William Feller. 1968. *An Introduction to Probability Theory and Its Applications*. Vol. 1. Wiley.

[26] Cella Florescu, Marc Kaufmann, Johannes Lengler, and Ulysse Schaller. 2024. Faster Optimization Through Genetic Drift. In *Proc. of PPSN 2024*. Springer-Verlag, Berlin, Heidelberg, 70–85.

[27] Tobias Friedrich, Pietro S. Oliveto, Dirk Sudholt, and Carsten Witt. 2009. Analysis of Diversity-Preserving Mechanisms for Global Exploration. *Evolutionary*

Computation 17, 4 (2009), 455–476.

[28] Leon Jay Gleser. 1975. On the Distribution of the Number of Successes in Independent Trials. *The Annals of Probability* 3, 1 (1975), 182–188.

[29] Spencer Greenberg and Mehryar Mohri. 2014. Tight lower bound on the probability of a binomial exceeding its expectation. *Statistics & Probability Letters* 86 (2014), 91–98. Issue C.

[30] Ryoki Hamano, Kento Uchida, Shinichi Shirakawa, Daiki Morinaga, and Youhei Akimoto. 2024. Tail Bounds on the Runtime of Categorical Compact Genetic Algorithm. *Evolutionary Computation* (12 2024), 1–49.

[31] Georges R Harik, Fernando G Lobo, and David E Goldberg. 1999. The compact genetic algorithm. *IEEE Transactions on Evolutionary Computation* 3, 4 (1999), 287–297.

[32] Václav Hasenöhrl and Andrew M Sutton. 2018. On the runtime dynamics of the compact genetic algorithm on jump functions. In *Proc. of GECCO 2018*. ACM Press, 967–974.

[33] Wassily Hoeffding. 1956. On the Distribution of the Number of Successes in Independent Trials. *The Annals of Mathematical Statistics* 27, 3 (1956), 713–721.

[34] Thomas Jansen, Kenneth A. De Jong, and Ingo Wegener. 2005. On the Choice of the Offspring Population Size in Evolutionary Algorithms. *Evolutionary Computation* 13, 4 (2005), 413–440.

[35] Martin S. Krejca and Carsten Witt. 2020. Lower bounds on the run time of the Univariate Marginal Distribution Algorithm on OneMax. *Theoretical Computer Science* 832 (2020), 143–165.

[36] Martin S. Krejca and Carsten Witt. 2020. Theory of Estimation-of-Distribution Algorithms. In *Theory of Evolutionary Computation: Recent Developments in Discrete Optimization*, Benjamin Doerr and Frank Neumann (Eds.). Springer, 405–442.

[37] Per Kristian Lehre and Phan Trung Hai Nguyen. 2018. Level-Based Analysis of the Population-Based Incremental Learning Algorithm. In *Proc. of PPSN 2018*, Vol. 11102. Springer, 105–116.

[38] Per Kristian Lehre and Phan Trung Hai Nguyen. 2021. Runtime Analyses of the Population-Based Univariate Estimation of Distribution Algorithms on LeadingOnes. *Algorithmica* 83, 10 (2021), 3238–3280.

[39] Johannes Lengler. 2020. Drift analysis. In *Theory of Evolutionary Computation: Recent Developments in Discrete Optimization*, Benjamin Doerr and Frank Neumann (Eds.). Springer, 89–131.

[40] Johannes Lengler, Dirk Sudholt, and Carsten Witt. 2021. The complex parameter landscape of the compact genetic algorithm. *Algorithmica* 83 (2021), 1096–1137.

[41] Frank Neumann, Dirk Sudholt, and Carsten Witt. 2010. A few ants are enough: ACO with iteration-best update. In *Proc. of GECCO 2010*. ACM Press, 63–70.

[42] Frank Neumann, Dirk Sudholt, and Carsten Witt. 2025. The Compact Genetic Algorithm Struggles on Cliff Functions. *Algorithmica* 87, 4 (2025), 507–536.

[43] Jonathan E. Rowe and Dirk Sudholt. 2014. The choice of the offspring population size in the (1, λ) evolutionary algorithm. *Theoretical Computer Science* 545 (2014), 20–38.

[44] Eric V. Slud. 1977. Distribution Inequalities for the Binomial Law. *The Annals of Probability* 5, 3 (1977), 404–412.

[45] Dirk Sudholt. 2020. The Benefits of Population Diversity in Evolutionary Algorithms: A Survey of Rigorous Runtime Analyses. In *Theory of Evolutionary Computation: Recent Developments in Discrete Optimization*, Benjamin Doerr and Frank Neumann (Eds.). Springer International Publishing, 359–404.

[46] Dirk Sudholt and Carsten Witt. 2019. On the choice of the update strength in estimation-of-distribution algorithms and ant colony optimization. *Algorithmica* 81 (2019), 1450–1489.

[47] Wenpin Tang and Fengmin Tang. 2023. The Poisson Binomial Distribution – Old & New. *Statistical Science* 38 (2023), 108–119. Issue 1.

[48] Carsten Witt. 2023. How majority-vote crossover and estimation-of-distribution algorithms cope with fitness valleys. *Theoretical Computer Science* 940 (2023), 18–42.

[49] Zijun Wu, Michael Kolonko, and Rolf H. Möhring. 2017. Stochastic runtime analysis of the cross-entropy algorithm. *IEEE Transactions on Evolutionary Computation* 21, 4 (2017), 616–628.

[50] Zijun Wu, Rolf H. Möhring, and Jianhui Lai. 2018. Stochastic runtime analysis of a Cross-Entropy algorithm for traveling salesman problems. *Theoretical Computer Science* 724 (2018), 69–86.