Theoretical Analyses of Univariate Estimation-of-Distribution Algorithms

Martin Stefan Krejca

Publikationsbasierte Universitätsdissertation
zur Erlangung des akademischen Grades

doctor rerum naturalium
(Dr. rer. nat.)

in der Wissenschaftsdisziplin
Theoretische Informatik

eingereicht an der
Digital-Engineering-Fakultät
der Universität Potsdam

Datum der Disputation: 29. August 2019

Betreuer

Prof. Dr. Tobias Friedrich
Hasso Plattner Institute, University of Potsdam

Gutachter

Prof. Dr. Benjamin Doerr
École Polytechnique

Prof. Dr. Carsten Witt
Technical University of Denmark

Published online at the
Institutional Repository of the University of Potsdam:
https://doi.org/10.25932/publishup-43487
https://nbn-resolving.org/urn:nbn:de:kobv:517-opus4-434870

Abstract

Optimization is a core part of technological advancement and is usually heavily aided by computers. However, since many optimization problems are hard, it is unrealistic to expect an optimal solution within reasonable time. Hence, *heuristics* are employed, that is, computer programs that try to produce solutions of high quality quickly. One special class are *estimation-of-distribution algorithms* (EDAs), which are characterized by maintaining a probabilistic model over the problem domain, which they evolve over time. In an iterative fashion, an EDA uses its model in order to generate a set of solutions, which it then uses to refine the model such that the probability of producing good solutions is increased.

In this thesis, we theoretically analyze the class of *univariate* EDAs over the Boolean domain, that is, over the space of all length-n bit strings. In this setting, the probabilistic model of a univariate EDA consists of an n-dimensional probability vector where each component denotes the probability to sample a 1 for that position in order to generate a bit string.

My contribution follows two main directions: first, we analyze general inherent properties of univariate EDAs. Second, we determine the expected run times of specific EDAs on benchmark functions from theory. In the first part, we characterize when EDAs are unbiased with respect to the problem encoding. We then consider a setting where all solutions look equally good to an EDA, and we show that the probabilistic model of an EDA quickly evolves into an incorrect model if it is always updated such that it does not change in expectation.

In the second part, we first show that the algorithms cGA and MMAS-fp are able to efficiently optimize a noisy version of the classical benchmark function OneMax. We perturb the function by adding Gaussian noise with a variance of σ^2, and we prove that the algorithms are able to generate the true optimum in a time polynomial in σ^2 and the problem size n. For the MMAS-fp, we generalize this result to linear functions. Further, we prove a run time of $\Omega\big(n\log(n)\big)$ for the algorithm UMDA on (unnoisy) OneMax. Last, we introduce a new algorithm that is able to optimize the benchmark functions OneMax and LeadingOnes both in $\mathrm{O}\big(n\log(n)\big)$, which is a novelty for heuristics in the domain we consider.

Zusammenfassung

Optimierung ist ein Hauptbestandteil technologischen Fortschritts und oftmals computergestützt. Da viele Optimierungsprobleme schwer sind, ist es jedoch unrealistisch, eine optimale Lösung in angemessener Zeit zu erwarten. Daher werden *Heuristiken* verwendet, also Programme, die versuchen hochwertige Lösungen schnell zu erzeugen. Eine konkrete Klasse sind *Estimation-of-Distribution-Algorithmen* (EDAs), die sich durch das Entwickeln probabilistischer Modelle über dem Problemraum auszeichnen. Ein solches Modell wird genutzt, um neue Lösungen zu erzeugen und damit das Modell zu verfeinern, um im nächsten Schritt mit erhöhter Wahrscheinlichkeit bessere Lösungen zu generieren.

In dieser Arbeit untersuchen wir die Klasse *univariater* EDAs in der booleschen Domäne, also im Raum aller Bitstrings der Länge n. Das probabilistische Modell eines univariaten EDAs besteht dann aus einem n-dimensionalen Wahrscheinlichkeitsvektor, in dem jede Komponente die Wahrscheinlichkeit angibt, eine 1 an der entsprechenden Stelle zu erzeugen.

Mein Beitrag folgt zwei Hauptrichtungen: Erst untersuchen wir allgemeine inhärente Eigenschaften univariater EDAs. Danach bestimmen wir die erwartete Laufzeit gewisser EDAs auf Benchmarks aus der Theorie. Im ersten Abschnitt charakterisieren wir, wann EDAs unbefangen bezüglich der Problemcodierung sind. Dann untersuchen wir sie in einem Szenario, in dem alle Lösungen gleich gut sind, und zeigen, dass sich ihr Modell schnell zu einem falschen entwickelt, falls es immer so angepasst wird, dass sich im Erwartungswert nichts ändert.

Im zweiten Abschnitt zeigen wir, dass die Algorithmen cGA und MMAS-fp eine verrauschte Variante des klassischen Benchmarks OneMax effizient optimieren, bei der eine Gaussverteilung mit Varianz σ^2 hinzuaddiert wird. Wir beweisen, dass die Algorithmen das wahre Optimum in polynomieller Zeit bezüglich σ^2 und n erzeugen. Für den MMAS-fp verallgemeinern wir dieses Ergebnis auf lineare Funktionen. Weiterhin beweisen wir eine Laufzeit von $\Omega\big(n \log(n)\big)$ für den Algorithmus UMDA auf OneMax (ohne Rauschen). Zuletzt führen wir einen neuen Algorithmus ein, der die Benchmarks OneMax und LeadingOnes in $\mathrm{O}\big(n \log(n)\big)$ optimiert, was zuvor für noch keine Heuristik gezeigt wurde.

Acknowledgments

Pursuing a PhD involves far more than writing a thesis. In fact, I believe that the experience gained beforehand is what really constitutes a PhD, with the thesis being only the closing remarks. Looking back to the beginning of my journey, I see that I have grown as a person over all these years, which makes me very proud, and there are many people to thank for this achievement.

The most influential person in my process of becoming a scientist was Timo Kötzing. He constantly supported me, gave me great advice and new hope when I was stuck, and was around whenever I needed him. His lessons went far beyond writing good papers – they considered good research at its core. His enthusiasm for science and teaching was addicting and inspiring, he showed genuine interest in my progress, and hosted great game nights on a weekly basis which I happily attended. I owe you a lot, Timo, and I wholeheartedly want to say thank you for all you have done for me.

Another very important person was Tobias Friedrich, whom I first met when I was a Master's student. We immediately got along well, and he directly offered me to start a PhD under his supervision, which I am immensely grateful for. It was a great time being part of his research group Algorithm Engineering, since Tobias is a very permissive and lenient supervisor. He gave me a lot of freedom to conduct research in the areas I was interested in while also making sure that I was financially secure and giving me helpful advice. Thank you very much, Tobias, for your great support.

Many thanks also to all of the other recent and former members of the entire research group Algorithm Engineering, which I had the honor to be a part of. That is, thank you to Thomas Bläsius, Katrin Casel, Ankit Chauhan, Vanja Doskoč, Philipp Fischbeck, Andreas Göbel, Katrin Heinrich, Maximilian Katzmann, Anton Krohmer, Gregor Lagodzinski, Pascal Lenzner, Anna Melnichenko, Louise Molitor, Stefan Neubert, Francesco Quinzan, Manuel Rizzo, Ralf Rothenberger, Martin Schirneck, Karen Seidel, Andrew Sutton, and Christopher Weyand. You are awesome! The group was like a second home to me, and I was always happy coming to work. I highly enjoyed the daily board game sessions and all of the other plenty group activities like playing beach volleyball or basketball. I especially would like to thank Louise for being the best office mate I could imagine, to Andreas for the great times when we were designing new games, to Ankit for always being cheerful and supporting everyone, to Gregor and Martin for many inspiring scientific and philosophical discussions, to Philipp and Thomas for hosting fun game nights, and to Ralf and Vanja for sharing my joy for board games and having fun at the game club. Additionally, special thanks to Andrew for always answering the many questions I had, to Thomas for providing me with a template for this document, and to Andreas and Timo for giving me valuable feedback on my thesis.

Besides the research group, I would like to thank Benjamin Doerr and Carsten Witt for agreeing to review this thesis and for each to have offered me to visit them for some weeks in order to collaborate. I had a great time and learned a lot during my stays. Especially thank you to Carsten, who agreed to be my mentor without even knowing me at that point in time. Further, I would like to thank Ralf Teusner for being an awesome member of our gaming group at Timo's place.

Besides all of the wonderful people I met during my time as a PhD student, there are great people outside of my scientific community who deserve at least as much credit. Most important of all are my brother Matthias Krejca as well as my mother Anne Arends-Krejca and my father Grzesiu Krejca, who support me endlessly in everything I do and bring happiness into my life every day. I am glad to have you. Thank you so much for everything.

Last, I would like to say thank you to Emanuel Barth, Philipp Burkhardt, Patrick Fleischer, and Lukas Klimmasch for entertaining me during my spare time and being there for all these years. We may not have seen each other often lately, but it is always a great pleasure to have you around, and after every meeting, I am looking forward to the next.

Contents

1 Introduction

Optimization is at the core of advancing human technology. A notable example is the strong increase in performance and memory capacity of computers over the last decades, which has lead to their widespread use in many different domains. With this increase in computing power and its wide availability, optimization is nowadays heavily aided by computers, making it thus also an essential field of research in computer science. Unfortunately, for many real-world problems, it seems unlikely to get an optimal solution in a reasonable amount of time, as they are often NP-hard. Hence, *heuristics* are a necessity, that is, algorithms that aim at quickly generating good (but not necessarily optimal) solutions.

Traditional heuristics are designed with a single problem in mind and revolve around exploiting properties specific to that problem. For example, the algorithm of Christofides [Chr76] yields a 1.5-approximation for the well known NP-hard Traveling Salesman Problem in metric spaces. And the NP-hard Maximum Cut Problem can even be solved optimally in polynomial time when given a planar graph [Had75].

In contrast to such problem-specific approaches stand *metaheuristics,* which are targeted at optimizing a wider range of problems. One example is the well known algorithm Simulated Annealing [KGV83], which optimizes a problem by iteratively varying a current solution randomly and accepting the result with a probability that is dependent on the quality gain or loss by that variation and on the time passed; initially, the probability of accepting a worse solution is high but diminishing quickly over time such that the optimization process can converge.

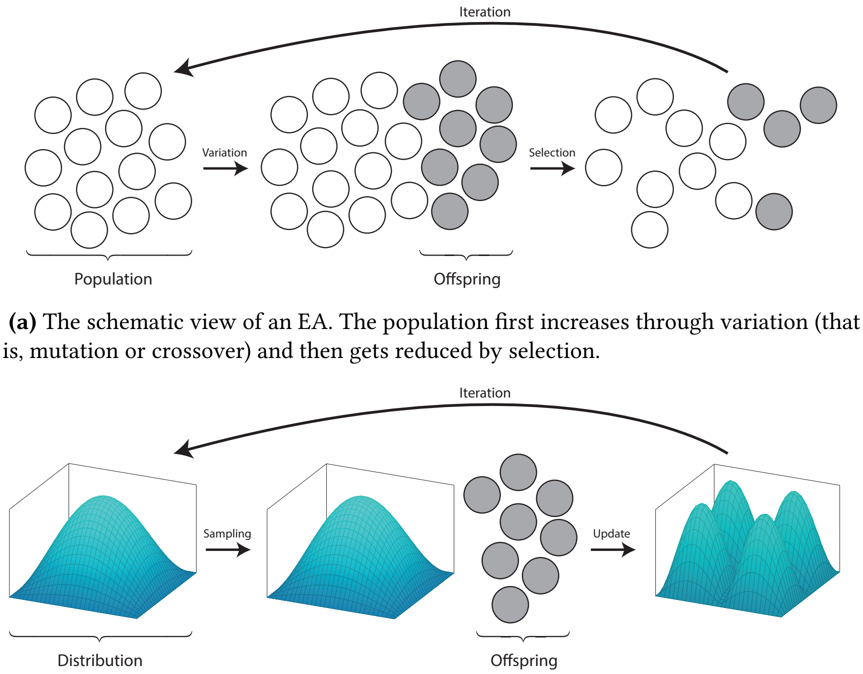
The general framework for metaheuristics consists of viewing the optimization problem as a *black box* and only interacting with it via some limited operations, such as modifying an instance or determining its quality. This approach makes metaheuristics widely applicable, especially since the problem itself does not need to be formalized – a subroutine for assessing the quality of a solution is already sufficient. Golovin et al. [Gol+17] even say that "Any sufficiently complex system acts as a black box when it becomes easier to experiment with than to understand”. For example, Arbonès et al. [Arb+16] optimize the energy gain from buoys submerged into the sea, which is a highly constrained problem, using a metaheuristic. The complexity of this problem is only communicated indirectly to the heuristic via a subroutine that simulates the energy gain of a given placement of buoys and thus determines its quality.

Many metaheuristics use operators inspired from real-world phenomena. For instance, Simulated Annealing is inspired by the physical process of annealing of metals. The field of *evolutionary computing* (EC [KP15, Part E]) considers metaheuristics that use operators inspired from phenomena in biology. One example are *evolutionary algorithms* (EAs [Sim13]), which use operators inspired by processes seen in natural evolution, such as mutation, recombination, and selection. These algorithms have a population of solutions (called *individuals*), which they use in order to create new individuals by modifying old solutions randomly. Figure 1.1 (a) sketches the outline of an EA.

A similar approach to EAs is taken by *estimation-of-distribution algorithms* (EDAs) [HP11; LL02; PHL15; PSC06], which are the focus of this thesis. The key feature of an EDA is its *probabilistic model* (instead of a population), which acts as a compact representation of a probability distribution over the search space. Initially, this model represents the uniform distribution. By sampling the model, the EDA generates individuals that act as an example of how well adapted the current model is to creating good solutions. Based on this observation, the EDA then updates its model in order to increase the quality of the samples. By iterating this process, the EDA evolves a model that shows how to generate good solutions. The outline of an EDA is depicted in Figure 1.1 (b), and the comparison between EAs and EDAs is described in Figure 1.1.

EDAs have been successfully applied to a wide range of combinatorial, high-dimensional problems with many dependencies and often outperformed competing algorithms [PHL15]. Next to their great performance, Pelikan et al. [PHL15] point out that another advantage of EDAs is the expressiveness of the probabilistic model created in the process, which allows for insights into the problem domain and acts as a justification for the created solutions. This expressiveness is a result of many EDAs using probabilistic graphical models (PGMs [KF09]) as their representation of a probability distribution over the problem domain, which are easy to interpret.

A PGM is a representation of a joint probability distribution via a graph, where each node denotes a random variable and each edge (directed or undirected) de-



(a) The schematic view of an EA. The population first increases through variation (that is, mutation or crossover) and then gets reduced by selection.

(b) The schematic view of an EDA. The algorithm samples offspring from a probabilistic model (depicted as a probability distribution) and then performs an update.

Figure 1.1: A comparison of the main differences between EAs and EDAs. An EA (Figure 1.1 (a)) works with an explicit population, whereas an EDA (Figure 1.1 (b)) uses a probabilistic model instead in order to create offspring. Afterward, both algorithms perform an update based on this new information.

notes a dependency among two random variables. A PGM can be generated from a problem by first determining relevant parameters (so-called *problem variables*) and then introducing one or more *decision variables* that the model should use in order to explain dependencies among the problem variables. The expressiveness of an EDA is then a direct result of the complexity of the underlying PGM. For example, a directed, acyclic graph results in a Bayesian network, whereas a directed path results in a chain. The simplest PGM is achieved if each problem variable is modeled by a single decision variable and no edges are allowed (that is, the graph is an independent set). Such a model is known as a *univariate* model, since the distribution of each problem variable is a function in only that problem variable (and no other variables).

1.1 Scope of this Thesis

I aim at the theoretical understanding of univariate EDAs. Although these algorithms only consider rather simple probabilistic models, I deem their analysis as important, as I do not think that more complicated EDAs can be well understood otherwise. In my opinion, univariate EDAs act as the natural first step to better understanding EDAs in general. Further, results for univariate EDAs are still rather scarce and their theoretical analysis is already quite challenging. Proof of this is that all theoretical results so far only considered univariate EDAs.

One advantage of univariate over more complex EDAs or population-based EAs is that they use very little memory [HP11]. In turn, this allows the algorithms to cope with bigger problem sizes and to run more iterations in a certain amount of time. Thus, they also provide a good alternative to switch to, once independent clusters of a problem have been found using other methods. Further, as we show in Chapter 10, the independence assumption does not mean that univariate EDAs cannot cope with dependencies at all.

In the following, we go into detail about some of the latest results for EDAs that show the complexity of the underlying research and the current focus. For a more complete picture on the research that has been conducted on EDAs, please refer to an article that is joint work with Carsten Witt [KW18b].

1.1.1 State of the Art

The current research of univariate EDAs is heavily centered around the *run time analysis* for the maximization of functions over bit strings (so-called *pseudo-Boolean* functions). Run time analysis is concerned with the expected number of function evaluations until an optimal solution is sampled for the first time. That is, it considers the evaluation of the objective function to be the most costly operation and ignores other operations that normally factor into the total run time. The aim of this line of research for EDAs is to compare their run time results to those of EAs, which have been studied in greater detail and also fall into the category of EC. For this purpose, certain benchmark functions are used, the most basic one being OneMax [Müh92], which is a common starting point for run time analysis. OneMax returns the number of 1s of a bit string and thus creates an easy slope: each bit string can be improved by changing any 0 into a 1. The intention of analyzing the performance of an algorithm on this function is to study its hill-climbing abilities, and a common run time for EAs is $\Theta\big(n\log(n)\big)$, where n denotes the length of the bit strings.[1]

Since the field of run time analysis of EDAs is rather young, only a few algorithms have been considered: most current results are concerned with the *compact genetic algorithm* (cGA [HLG99]) or the *univariate marginal distribution algorithm* (UMDA [MP96]) and mostly consider OneMax. For both of them, the tight run time bound of $\Theta\big(n\log(n)\big)$ has been proven (assuming optimal parameter choices). These results are remarkable, as they show that the cGA and the UMDA are able to optimize OneMax as efficiently as most EAs, although for completely different reasons. EAs exhibit this run time because the potential for improvement declines over time, as fewer 0s (which need to be changed to 1s) occur in solutions of better quality, slowing down the optimization process. For EDAs, the run time is a result of how fast the probabilistic model can be changed such that it generates an optimal solution.

Sudholt and Witt [SW16a] were the first to prove a tight run time bound of an EDA by analyzing how its update process slows down the run time. The authors considered the cGA, which is a univariate EDA and thus uses a vector of probabilities $\boldsymbol{p}$ (the *frequency vector*) as its probabilistic model, where each component $\boldsymbol{p}_i$ (a *frequency*) denotes the probability to sample 1 at position i. In each iteration, the cGA generates two solutions (bit strings) via its current frequency vector and ranks them by their quality. It then updates each frequency with respect to the bias viewed between the bits of the better and the worse solution. In more detail, let $\boldsymbol{x}$ denote the better solution and $\boldsymbol{y}$ the worse. For each frequency $\boldsymbol{p}_i$, if $\boldsymbol{x}_i = \boldsymbol{y}_i$, no update is performed; if $\boldsymbol{x}_i > \boldsymbol{y}_i$, then $\boldsymbol{p}_i$ is increased by $1/K$ (the *step size*), where K is an algorithm-specific parameter; and if $\boldsymbol{x}_i < \boldsymbol{y}_i$, then $\boldsymbol{p}_i$ is decreased by $1/K$. Usually the cGA has a *margin,* that is, the frequencies are restricted to the interval $[1/n, 1-1/n]$, where n is the length of the bit strings, in order to avoid having frequencies of 0 or 1, which would result in only sampling 0s or 1s at the respective position.

Sudholt and Witt [SW16a] proved that the cGA with a margin optimizes OneMax in $\Theta\big(n\log(n)\big)$ in expectation (assuming an optimal parameter choice). That is, for the analysis of the cGA, it is important to understand how fast the

[1] See also Table 10.1 for a detailed overview of run time results for many EAs and EDAs.

frequencies can get to $1 - 1/n$, since this increases the probability to sample the optimum. At a first glance, it may seem preferable to have a large step size, as this allows the frequencies to be increased quickly. However, due to the randomness in the sampling process, some frequencies may be decreased in the update process. If the step size is large, then a frequency reaches the wrong end of the spectrum ($1/n$) and it takes some time to recover from this mistake. Sudholt and Witt [SW16a] showed that if the step size is too large (that is, $K \in \mathrm{o}\big(\sqrt{n}\log(n)\big)$), too many frequencies reach $1/n$, which results in a lower bound in the order of $n\log(n)$. For smaller step sizes of $K \in \Omega\big(\sqrt{n}\log(n)\big)$, the process is slowed down by how long it now takes a frequency to reach the correct side of the spectrum, that is, $1 - 1/n$. Hence, the run time grows in the step size, and the overall lower bound is $\Omega\big(K\sqrt{n} + n\log(n)\big)$. The matching upper bound of $\mathrm{O}\big(K\sqrt{n}\big)$ for $K \in \mathrm{O}\big(\sqrt{n}\log(n)\big)$ was also proven by Sudholt and Witt [SW16a] by carefully analyzing how unlikely it is for a frequency too decrease too often if the step size is small.

As mentioned before, the other EDA with a similarly detailed analysis on OneMax is the UMDA, which is univariate like the cGA and thus also uses a frequency vector $\boldsymbol{p}$. However, the way the UMDA updates its frequency vector is rather different from the cGA: the UMDA generates λ solutions every iteration (instead of 2), where λ is an algorithm-specific parameter. It then ranks all solutions by their quality and chooses the best μ, where μ is another algorithm-specific parameter. Each frequency $\boldsymbol{p}_i$ is then *set* to the relative number of 1s at position i among the μ best individuals. Thus, similarly to the cGA, the frequencies of the UMDA can only take discrete values which are multiples of $1/\mu$. However, a strong difference to the cGA is that the UMDA does not use the old frequency value when performing an update. Since all solutions are generated independently, the change of a frequency during a single iteration tends to concentrate around its expected value though, since too large jumps are very unlikely. As for the cGA, the UMDA is usually considered with a margin, that is, its frequencies are restricted to $[1/n, 1 - 1/n]$.

Although the update process of the UMDA works differently from that of the cGA, the expected run time of the UMDA on OneMax was proven to be in $\Theta\big(n\log(n)\big)$ too (assuming optimal parameter choices), where the upper bound has been proven independently by Lehre and Nguyen [LN17] and Witt [Wit17].[2] The results for the cGA and the UMDA coincide even so far that the lower

[2] The lower bound is part of Chapter 9 and joint work with Carsten Witt.

and upper bounds of both algorithms take the same form (when assuming that $K = \lambda$), considering different parameter restrictions.[3] It is important to note though that this run time considers the expected number of function evaluations, not iterations. Since the UMDA performs λ function evaluations in each iteration, it takes only a λ-fraction of iterations when compared to the cGA but performs larger steps in expectation (assuming $K = \lambda$).

The analysis by Witt [Wit17] for the upper bound of the UMDA shows that the (optimal) run time of $\Theta\big(n \log(n)\big)$ is achieved for the two different parameter settings of $\lambda \in \Theta\big(\log(n)\big)$ and $\lambda \in \Theta\big(\sqrt{n} \log(n)\big)$ (always assuming that $\mu \in \Theta(\lambda)$). In comparison, for the cGA, Sudholt and Witt [SW16a] proved an upper bound of $\Theta\big(n \log(n)\big)$ only for the single parameter choice of $K \in \Theta\big(\sqrt{n} \log(n)\big)$. This begs the question of whether the regime of $K \in \Theta\big(\log(n)\big)$ for the cGA is also optimal. Although this question remains still unanswered, recent results by Lengler et al. [LSW18] suggest a bimodal behavior in the run time of both the cGA and the UMDA between the two parameter settings mentioned above, as they proved that the cGA has an expected run time of $\Omega\big(K^{1/3} n + n \log(n)\big)$ on OneMax for $K \in \mathrm{O}\big(\sqrt{n}/\log(n)^2\big)$. Note that this lower bound is strictly worse than $\Theta\big(n \log(n)\big)$ for $K \in \omega\big(\log(n)^3\big)$, which leads to the assumed bimodal behavior in run time (supposing that the cGA actually has a run time of $\Theta\big(n \log(n)\big)$ for $K \in \Theta\big(\log(n)\big)$).

1.2 Contribution and Outline

With this thesis, I contribute to the understanding of univariate EDAs and to the state-of-the-art results in two major ways: first, I investigate general, intrinsic properties of these algorithms. This shows certain possibilities and limits of EDAs, and it also provides insights that carry over to analyses with a different focus, such as run time analysis. Second, I analyze the run time behavior of certain EDAs on standard benchmark functions, while also using the insights about the intrinsic properties from the first part. These analyses complement the existing run time results and add to the bigger picture of EDAs. Additionally, all of the results use various tools for coping with random processes, and this thesis thus also serves as a demonstration of how to analyze EDAs theoretically.

Most of the chapters are based on joint work with different co-authors. If this is the case, it is mentioned at the beginning of the respective chapter. We now give an overview of all of the following chapters.

[3] Please refer to Table 10.1 for more details on the parameter restrictions.

Chapter 2 is concerned with the mathematical background for all of the analyses. We introduce important notation, the domain of our optimization problems, as well as the stochastic tools we use. As this thesis is focused on the analysis of randomized processes, we place emphasis on topics from probability theory that we consider more advanced, such as expected values conditional on a σ-algebra.

In Chapter 3, we go into detail about *drift theory,* which is a collection of useful theorems that provide bounds on the expected first-hitting times of processes by only bounding their local expected progress (the *drift*). Since these theorems are easy to use and widely applicable, they are a fundamental tool in the theory of EC, especially for run time analysis. We discuss the foundation of drift theory and prove drift theorems for different regimes in the up-to-date most general form. Overall, this chapter provides an overview about state-of-art drift theorems and educates the reader about their background.

In Chapter 4, we introduce the n-Bernoulli-λ-EDA (Algorithm 1) – a framework for univariate EDAs, which acts as a foundation for most of our results. An n-Bernoulli-λ-EDA is any univariate EDA that has a vector of probabilities (the *frequency vector*; each component being a *frequency*) that is updated iteratively only using the λ samples from the current iteration. In Chapters 5 and 6, we analyze general properties of the class of n-Bernoulli-λ-EDAs, and in the remaining chapters (besides Chapter 10), the algorithms we consider are special instances of the framework. Further, we show how the most commonly theoretically analyzed EDAs are all n-Bernoulli-λ-EDAs, and we give an overview about known results for these algorithms.

In Chapter 5, we consider the concept of *unbiasedness* introduced by Lehre and Witt [LW12], which has only been used in the context of EAs before, and we prove an alternative characterization for n-Bernoulli-λ-EDAs. An algorithm is said to be unbiased if an only if it performs the same when optimizing a perturbed objective function as when optimizing the unperturbed function, where a perturbation is any isometric automorphism of the discrete hypercube. Our characterization extends this notion to automorphisms of the continuous hypercube and shows how the invariance property of the original definition carries over to the frequency vector of n-Bernoulli-λ-EDAs. We then prove that almost all of the algorithms we consider in this thesis are unbiased. This result generalizes all time results for functions to closures of these functions under isometric automorphisms. In our case, all of our run time results hold for a class of functions instead of a single function, due to our results from Chapter 5.

In Chapter 6, we consider the update process of an n-Bernoulli-λ-EDA under a constant function, which models a setting in that the algorithm does not get any information on which parts of the search space are more beneficial. We say that the algorithm is *balanced* if and only if a frequency does not change in expectation after an update (that is, it is a martingale), and we say that the algorithm is t-*stable* if and only if the frequency only takes constant values for t iterations with high probability. Both properties are desirable in our considered setting, as we do not want the algorithm to introduce a bias into the update process (which would violate *balanced*), and we do not want a frequency to take values subconstantly close to 0 or 1 (which would violate *stable*), as it is hard to escape from these and the decision would be arbitrary. We prove that all commonly analyzed n-Bernoulli-λ-EDAs are balanced and that no n-Bernoulli-λ-EDA can be both balanced and stable. Hence, common EDAs are not stable, and we even show how fast their frequencies reach extremal values with a decent probability. Last, we introduce the scGA – a new n-Bernoulli-λ-EDA, which is stable (and thus not balanced) and prove that it optimizes the benchmark function LeadingOnes in $\mathrm{O}\big(n \log(n)\big)$. This is faster than the usual run time of $\Theta\big(n^2\big)$ for common EAs (see Table 10.1).

The remaining chapters are concerned with run time analysis. In Chapters 7 and 8, we consider noisy optimization, that is, every time the objective function is evaluated at a point, we add Gaussian noise to the original value. We then consider the number of function evaluations that are needed until the unnoisy optimum is sampled for the first time. We determine this run time with respect to the dimension of the problem space as well as the variance of the noise. If an algorithm has a run time that is polynomial in both of these quantities, we say that it *scales gracefully* with the considered noise. In Chapter 7, we show that the cGA scales gracefully on the benchmark function OneMax in our setting of additive posterior Gaussian noise, and in Chapter 8, we show that the algorithm MMAS-fp scales gracefully on the class of linear pseudo-Boolean functions. These results show that EDAs are well suited to optimize noisy functions. This stands in contrast to mutation-only EAs, which fail to optimize a noisy function with high probability once the variance of the noise is too large and thus do not scale gracefully [Fri+17].

In Chapter 9, we prove a lower bound of $\Omega\big(n \log(n)\big)$ of the UMDA on (unnoisy)

OneMax. This result mirrors a similar prior result by Sudholt and Witt [SW16a], who proved the same lower bound for the cGA and whose proof idea we adapt in our proof. Our analysis considers three different parameter regimes of the sample size λ of the UMDA. If λ is rather small or rather large, the run time bound follows by simple stochastic arguments. For the medium range of λ, we use more advanced tools, as we want to prove that frequencies reach the incorrect value of $1/n$ with a sufficiently large probability. In order to show this, we define a potential function that scales the search space such that the drift of the resulting process is independent of its expected value *and* its variance. This allows us to approximate this process after a sufficient number of iterations via a normal distribution, for which good tail bounds exit. We then use these bounds in order to show that it is sufficiently likely for frequencies to reach incorrect values, which overall slows down the optimization process. In combination with recently proven matching upper bounds [LN17; Wit17], we can conclude that the UMDA has a run time of $\Theta\big(n\log(n)\big)$ on OneMax (for optimal parameters).

Chapter 10 builds upon the concepts of *balanced* and *stable* from Chapter 6 and introduces the sig-cGA (Algorithm 2) – a univariate EDA that can be thought of as being both balanced and stable (for a polynomial number of iterations). Since n-Bernoulli-λ-EDAs cannot be both, the sig-cGA does not fall into this framework albeit still being univariate. The reason for not being an n-Bernoulli-λ-EDA is that the sig-cGA contains a history of all of the bit values of good solutions among *multiple* iterations. It uses this information in order to perform an update for a frequency if a statistical significance in a history is detected. We prove that the sig-cGA optimizes the two benchmark functions OneMax and LeadingOnes both in $\mathrm{O}\big(n\log(n)\big)$, which is the first time that such a result has been proven for any EDA or EA (see also Table 10.1). Additionally, we prove that the scGA from Chapter 6, which is an n-Bernoulli-λ-EDA and is also able to optimize LeadingOnes in $\mathrm{O}\big(n\log(n)\big)$, has an exponential run time on OneMax. This suggests that giving up the *balanced* property (as done for the scGA) in order to become stable does not seem beneficial, whereas leaving the n-Bernoulli-λ-EDA framework in order to be balanced and stable yields useful results.

Last, in Chapter 11, we conclude this thesis by emphasizing the most important lessons learned from our results and by giving an overview on open questions in the field of theory of EDAs.

2 Preliminaries

In this chapter, we introduce the notation that we use throughout this thesis, the benchmark functions that we consider in our analyses, as well as the stochastic tools we use in order to derive our results. We assume that the reader is familiar with basic probability theory, and we only provide details for topics we consider more advanced. For a more detailed introduction, we refer to the book by Mitzenmacher and Upfal [MU05].

2.1 Notation

We use bold upright letters to denote number sets. For example, $\mathbf{N}$ denotes the set of all natural numbers (including 0), and $\mathbf{R}$ denotes the set of all real numbers. For any $m, n \in \mathbf{N}$, we define $[m..n] \coloneqq [m, n] \cap \mathbf{N}$, that is, the discrete interval from m to n. Further, we define the special case of $[n] \coloneqq [1..n]$.

We use bold italic letters to denote vectors. For any $n \in \mathbf{N}$, any $i \in [n]$, and any vector $\boldsymbol{p} \in \mathbf{R}^n$, we denote the i-th component of $\boldsymbol{p}$ with $\boldsymbol{p}_i$. Vectors where each component is the same number are denoted by writing the respective number in bold, that is, for any $r \in \mathbf{R}$, we let $\mathbf{r}$ denote the all-r vector (of a dimension that has been established prior). For example, $\mathbf{1/2}$ denotes the all-$1/2$ vector.

In this thesis, we consider pseudo-Boolean functions, that is, functions mapping from the n-dimensional discrete hypercube (bit strings) to real numbers, formally $f : \{0, 1\}^n \to \mathbf{R}$, which we call *fitness functions*. We call the value of a fitness function *fitness*. If not stated otherwise, $n \in \mathbf{N}^+$ denotes the dimension of the domain of a fitness function. When performing run time calculations, we are interested in asymptotic results with respect to n.

For asymptotic notation, we follow the standard conventions of Landau notation ([Cor+09, Chapter 3]). That is, we do not use it for functions but function values instead. For example, $n \in \mathrm{O}(n^2)$ means that the function $n \mapsto n$ grows as most as fast as the function $n \mapsto n^2$, up to a constant factor; and $a \in \mathrm{o}(1)$ means that a is a sequence converging to 0 (in some variable that is not named; normally the problem size n). Further, we use $\mathrm{poly}(n)$ for any polynomial (in n).

For two bit strings $\boldsymbol{x}, \boldsymbol{y} \in \{0, 1\}^n$, let $d_{\mathrm{H}}(\boldsymbol{x}, \boldsymbol{y})$ denote their Hamming distance, that is, the number of positions that their differ in. More formally, we define $d_{\mathrm{H}}(\boldsymbol{x}, \boldsymbol{y}) := |\{i \in [n] \mid \boldsymbol{x}_i \neq \boldsymbol{y}_i\}|$. If two bit strings have a Hamming distance of 1, we say that they are neighbors. Further, let $\|\boldsymbol{x}\|_0$ denote the number of 0s of $\boldsymbol{x}$, and let $\|\boldsymbol{x}\|_1$ denote the number of 1s.

We denote the characteristic function of a set A with $\mathbf{1}_A$. We may use a proposition P instead of a set A and write $\mathbf{1}\{P\}$, meaning that the characteristic function is 1 if the proposition is true and 0 otherwise.

2.1.1 Fitness Functions

The run time results of this thesis consider the two, most commonly analyzed, fitness functions OneMax [Müh92] and LeadingOnes [Rud97].

OneMax

In its standard form, OneMax is a linear function that returns the number of 1s in a bit string. It is considered to be one of the easiest functions with a unique optimum for most EAs [Sud13; Wit13]. Formally, it is defined as

$$\textsc{OneMax}(\boldsymbol{x}) = \sum_{i \in [n]} \boldsymbol{x}_i \ . \tag{2.1}$$

A bit string with i 0s has i neighbors with strictly larger OneMax value. Hence, in order to maximize this function, it is sufficient to iteratively go from a bit string to any neighbor with a better fitness. This process of making local progress is known as *hill climbing,* and OneMax is used as a benchmark function to analyze how well an algorithm is able to perform such a hill-climbing task.

Note that the all-1s bit string $\mathbf{1}$ is the unique global optimum of OneMax. However, the function can easily be generalized to have an arbitrary bit string $\boldsymbol{a} \in \{0, 1\}^n$ as optimum. The function value is then defined to be the Hamming distance to $\boldsymbol{a}$, that is,

$$\textsc{OneMax}_{\boldsymbol{a}}(\boldsymbol{x}) = d_{\mathrm{H}}(\boldsymbol{x}, \boldsymbol{a}) \ .$$

For each $\boldsymbol{a} \in \{0, 1\}^n$, we get a different function $\textsc{OneMax}_{\boldsymbol{a}}$, and we call the set of all of these 2^n different functions the OneMax function class. In Chapter 5, we prove that it is sufficient for our considerations to only analyze OneMax as the representative function of this function class, since our algorithms of interest are unbiased with respect to the representation of the function's domain.

A common run time for EAs or EDAs on OneMax (or, usually, the entire OneMax function class) is $\Theta\big(n\log(n)\big)$ (see Table 10.1). For EAs, this run time is often a result from the coupon collector process [MU05, Section 2.4.1], as the n different bit values that need to be optimized can be thought of as n different coupons that need to be collected. For EDAs, the run time is a result of how fast the underlying probabilistic model converges to one that samples the optimal solution with high probability. We go more into detail about this in Chapters 9 and 10.

LeadingOnes

LeadingOnes is a function that returns the number of consecutive 1s in a bit string, starting from the leftmost position. Formally,

$$\textsc{LeadingOnes}(\boldsymbol{x}) = \sum_{i\in[n]}\prod_{j\in[i]} \boldsymbol{x}_i\ . \tag{2.2}$$

Different from OneMax, LeadingOnes is not linear, as the product introduces dependencies among the different bits. Now, each bit string (expect the global optimum $\mathbf{1}$) has exactly one neighbor with strictly better fitness. Hence, hill climbing is still sufficient, but, in order to progress locally, the correct neighbor has to be chosen in every iteration. Thus, optimizing LeadingOnes can be thought of as finding a hidden permutation [Afs+13].

Similar to OneMax, the definition of LeadingOnes can be generalized to an arbitrary optimum $\boldsymbol{a} \in \{0,1\}^n$ and an arbitrary permutation π of $\{0,1\}^n$. The optimum $\boldsymbol{a}$ determines which bit is the correct one for each position, and the permutation π determines the order in which the correct bits have to occur before the next bit contributes to the fitness at all. This results in the generalized definition of

$$\textsc{LeadingOnes}_{\boldsymbol{a},\pi}(\boldsymbol{x}) = \sum_{i\in[n]}\prod_{j\in[i]} \mathbb{1}\{\boldsymbol{x}_{\pi(j)} = \boldsymbol{a}_{\pi(j)}\}\ .$$

For each $\boldsymbol{a} \in \{0,1\}^n$ and each permutation π of $\{0,1\}^n$, we get a different function $\textsc{LeadingOnes}_{\boldsymbol{a},\pi}$, and we call the set of all of these $2^n n!$ different functions the LeadingOnes function class. Again in Chapter 5, we prove that it is sufficient to only consider LeadingOnes as the representative of this class for the algorithms we analyze.

A common run time for EAs or EDAs on LeadingOnes (or, usually, the entire LeadingOnes function class) is $\Theta(n^2)$ (see Table 10.1). The increase in the run time when compared to OneMax is, generally speaking, a result of determining the underlying permutation of the function. It is not sufficient to determine the n correct bits anymore; they also have to be determined in the correct order. In Chapter 10, we present an algorithm (Algorithm 2) that, albeit still relying in uncovering the bits in the correct order, optimizes LeadingOnes in $\mathrm{O}(n \log(n))$ by optimizing each position in time $\mathrm{O}(\log(n))$.

2.2 Probability Theory

In this section, we introduce the concepts of probability theory that we use in this thesis. We start with elementary topics and continue with more advanced topics such as *conditional expected values, filtrations,* and *stopping times.* All of the following definitions (and far more) can be found in the textbook by Grimmett and Stirzaker [GS01b].

2.2.1 Probability Spaces and Events

A *probability space* is a triple $(\Omega, \mathcal{A}, P)$, where the *sample space* $\Omega \neq \emptyset$ is the set of all elementary events, $\mathcal{A} \subseteq 2^{\Omega}$ is a *σ-algebra* containing all the events we assign a probability to, and $P\colon \mathcal{A} \to [0, 1]$ is a *probability measure* that assigns probabilities to all events from $\mathcal{A}$. Throughout this thesis, we do not state our probability spaces explicitly, since they are clear from context.

We use the notation $\Pr[A]$ to denote the probability of an event A of an implicit probability space, and we say that such an event occurs *with high probability* if and only if $\Pr[\overline{A}] \in \mathrm{O}(1/\mathrm{poly}(n))$, that is, the probability for A to not occur is at most an inverse polynomial. Further, we say that an event A occurs *with overwhelming probability* if and only if there is a $k \in \Omega(n)$ such that $\Pr[\overline{A}] = 2^{-k}$. Recall that n denotes the dimension of a fitness function.

For two events A and B, with $\Pr[B] \neq 0$, we denote the *probability of A conditional on B* with $\Pr[A \mid B] := \Pr[A \cap B]/\Pr[B]$, that is, the probability of A, assuming that B occurred. Effectively, we consider a new probability space with B as its new sample space. Conditional probabilities are a very important concept that most of our tools build upon.

The following theorem shows how conditional probabilities can be used to calculate a certain probability.

▶ **Theorem 2.1 (Law of Total Probability [GS01b, Chapter 1.4, Lemma 4]).** Let A be an event, let $I \subseteq \mathbf{N}$, and let $\{B_i\}_{i \in I}$ be a partition of Ω such that, for all $i \in I$, we have $\Pr[B_i] > 0$. Then

$$\Pr[A] = \sum_{i \in I} \Pr[A \mid B_i] \cdot \Pr[B_i] .$$ ◀

Theorem 2.1 determines $\Pr[A]$ via a case distinction with respect to the different events $\{B_i\}_{i \in I}$ that can occur.

We call two events A and B (with $\Pr[B] \neq 0$) *independent* if and only if $\Pr[A \mid B] = \Pr[A]$. Intuitively, knowing that B occurred does not change the probability of A.

For the rest of this chapter, we always assume that we are given a probability space $(\Omega, \mathcal{A}, \Pr)$.

2.2.2 Random Variables

In this thesis, we analyze quantities of algorithms that involve randomness, such as the run time of a randomized algorithm or the quality of a solution at a certain point in time. This concept is formally known as a *random variable.*

▶ **Definition 2.2 (Random Variable).** A *random variable* is any function $X : \Omega \to \mathbf{R}$. ◀

We only consider *discrete* random variables in this chapter, that is, random variables X with a countable range (denoted as $\mathrm{rng}(X)$). For continuous random variables, many of the following concepts apply similarly though.

The probability of an outcome x of a random variable X is determined by considering the probability of all preimages of x. More formally, the probability that X takes the value of $x \in \mathrm{rng}(X)$ is

$$\Pr\left[X^{-1}(x)\right] = \Pr\left[\{\omega \in \Omega \mid X(\omega) = x\}\right] .$$

We use the more conventional style and denote the above probability with $\Pr[X = x]$. Further, we denote the respective event with $\{X = x\}$, that is, we denote it as a set in order to highlight that we consider an event, not a relation. Consequently, the notation $\{X \geq x\}$ is used to denote the union of all events $\{X = y\}$ with $y \geq x$.

Whenever we compare two random variables X and Y, for example, when writing $X \geq Y$, we mean that this comparison is true for all elementary events, that is, for all $\omega \in \Omega$, it holds that $X(\omega) \geq Y(\omega)$. Note that such a statement is different from the event $\{X \geq Y\}$. As a special case, when we write $X = Y$, both random variables are identical.[4]

We say that two random variables X and Y are *independent* if and only if, for all $x \in \mathrm{rng}(X)$ and all $y \in \mathrm{rng}(Y)$, the events $\{X = x\}$ and $\{Y = y\}$ are independent.

Last, given two random variables X and Y, we say that Y *stochastically dominates* X, written as $Y \succeq X$, if and only if, for all $x \in \mathbf{R}$, it holds that $\Pr[Y \leq x] \leq \Pr[X \leq x]$. Note that, for all $x \in \mathbf{R}$, this is equivalent to $\Pr[Y > x] \geq \Pr[X > x]$.

For all of our considerations so far, we assumed that the events $\{X = x\}$ are actually part or our σ-algebra. However, note that this does not necessarily need to be true, which leads to the following definition.

▶ **Definition 2.3 (Measurable Random Variable).** Let $\mathcal{B} \subseteq \mathcal{A}$ be a σ-algebra. A (discrete) random variable X is *$\mathcal{B}$-measurable* if and only if, for all $x \in \mathrm{rng}(X)$, it holds that

$$\{X = x\} \in \mathcal{B} .$$ ◀

We only consider measurable random variables. However, the concept of a measurable random variable becomes relevant once we restrict the events that we want to assign a probability to.

Distributions

Given a random variable X, the function $D\colon \mathrm{rng}(X) \to [0, 1]$ with $D(x) = \Pr[X = x]$ is called the *distribution of* X. We use the standard notation of $X \sim D$ to denote that X follows distribution D. Examples of very common distributions are the following.

[4] We extend this notation to also apply when X and Y are equal *almost surely*, that is, for all positive probabilities. Analogously, when we speak of *unique* random variables, we also mean almost surely.

Uniform Distribution. For an $n \in \mathbf{N}^+$, we say that a random variable X that takes values in $[n]$ follows a uniform distribution, denoted as $X \sim \mathrm{Unif}(n)$, if and only if

$$\Pr[X = n] = \frac{1}{n} \ .$$

This means that every outcome of X is equally likely.

Bernoulli Distribution. We say that a random variable X that only takes the values 0 and 1 follows a Bernoulli distribution with parameter $p \in [0, 1]$, denoted as $X \sim \mathrm{Ber}(p)$, if and only if

$$\Pr[X = 1] = p \quad \text{and} \quad \Pr[X = 0] = 1 - p \ .$$

This means that a Bernoulli-distributed random variable denotes an indicator function for an event that occurs with probability p.

Binomial Distribution. When considering $n \in \mathbf{N}^+$ independent Bernoulli trials, all with the same success probability $p \in [0, 1]$, we get the Binomial distribution. We say that a random variable X that takes values in $[0..n]$ follows a Binomial distribution with parameters n and p, denoted as $X \sim \mathrm{Bin}(n, p)$, if and only if

$$\Pr[X = k] = \binom{n}{k} p^k (1 - p)^{n-k} \ .$$

Geometric Distribution. The Bernoulli distribution considers a single trial with success probability p, and the binomial distribution considers n independent trials with such probability. Similar to those, the geometric distribution also considers independent trials with the same success probability, but it considers a variable amount of trials – the amount until a success occurs for the first time. We say that a random variable X that takes values in $\mathbf{N}^+$ follows a geometric distribution with parameter $p \in [0, 1]$, denoted as $X \sim \mathrm{Geo}(p)$, if and only if

$$\Pr[X = k] = (1 - p)^{k-1} p \ .$$

In this thesis, we may use the terms *distribution* and *random variable* synonymously. For example, we may write $\Pr[\mathrm{Bin}(n, 1/2) = 2]$ to denote the probability of a binomially distributed random variable to take the value 2, without introducing a random variable.

For the expected values and variances of the distributions above, we refer to standard literature [GS01b; MU05].

2.2.3 Expected Values

A natural measure for random variables is their average value with respect to their distribution, called the *expected value* of a random variable. This concept plays a crucial role in our analyses.

▶ **Definition 2.4 (Expected Value).** Let X be a random variable. The *expected value of* X is the value

$$\begin{aligned} \mathrm{E}[X] &:= \sum_{x \in \mathrm{rng}(X)} x \cdot \Pr[X = x] \\ &= \sum_{\omega \in \Omega} X(\omega) \cdot \Pr[\{\omega\}] \,. \end{aligned}$$ ◀

Note that we do not state an order in which we add the terms. Hence, we assume that the above series is absolutely convergent, that is, $\mathrm{E}[|X|]$ converges. This allows us to reorder the terms arbitrarily. Whenever $\mathrm{E}[|X|]$ is not absolutely convergent, the expected value of X is not defined.

The first equality in Definition 2.4 is the more common and arguably more intuitive definition, which does not regard the sample space. However, it may get confusing when considering more complex expressions like $\mathrm{E}[XY]$, where the ranges of X and Y can be different. In these cases, the second equality is helpful, as it sums over the common domain of X and Y, which is Ω.

A very important property of the expected value is that it is linear.

▶ **Theorem 2.5 (Linearity of Expectation [GS01b, Chapter 3.3, Theorem 8]).** Let $a, b \in \mathbf{R}$, and let X and Y be random variables. Then

$$\mathrm{E}[aX + bY] = a\mathrm{E}[X] + b\mathrm{E}[Y] \,.$$ ◀

Theorem 2.5 gives us a rule of how the expected value of a complex random variable can be calculated by breaking it down into sums of simpler ones, regardless of any dependencies among them.

A concept related to the expected value of X is the *variance* of X – a measure for how largely X deviates from its expected value.

▶ **Definition 2.6 (Variance).** Let X be a random variable. The *variance of* X is the value

$$\begin{aligned}\mathrm{Var}[X] &\coloneqq \mathrm{E}\left[(X-\mathrm{E}[X])^2\right]\\ &= \mathrm{E}\left[X^2\right]-\mathrm{E}[X]^2\,. \end{aligned}$$ ◀

Note that the second equality follows from Theorem 2.5.

The variance is the expected value of a special transformation of a random variable. However, this transformation is sometimes useful, as we will see, for example, in Section 3.6.

Since the variance is defined via an expected value, all of the properties and definitions for expected values apply to the variance as well (when considering the respectively transformed random variable).

Conditional Expected Values

In our analyses, we often consider a certain state of an algorithm and determine what the expected next state is. In order to perform such calculations, we need to consider expected values conditional on that a certain event occurred.

As we already briefly discussed in Section 2.2.1, conditioning on an event A (with $\Pr[A] \neq 0$) can be thought of as considering a new probability space with sample space A. This leads to the following definition of a conditional expected value.

▶ **Definition 2.7 (Conditional Expected Value (on an Event)).** Let X be a random variable, and let A be an event with $\Pr[A] \neq 0$. The *expected value of* X *conditional on* A is the value

$$\begin{aligned}\mathrm{E}[X \mid A] &\coloneqq \sum_{x\in\mathrm{rng}(X)} x\cdot\Pr[X=x \mid A]\\ &= \sum_{\omega\in\Omega} X(\omega)\cdot\Pr[\{\omega\}\mid A]\,. \end{aligned}$$ ◀

Our aforementioned intuition of transforming the probability space gets obvious by noting that the second equality can be expressed as

$$\frac{1}{\Pr[A]} \sum_{\omega \in A} X(\omega) \cdot \Pr[\{\omega\}] ,$$

since $\Pr[\{\omega\} \mid A] = \Pr[\{\omega\}]/\Pr[A]$ if $\omega \in A$, and it is 0 otherwise. This means that we consider A as the new sample space and scale every probability accordingly. Note that this implies that Theorem 2.5 also holds for conditional expected values.

Definition 2.7 is useful when we know what event we are conditioning on, for example, when we know which state an algorithm is in. However, such information may be subject to randomness itself. Hence, we have to consider *all* possible outcomes of the condition. That is, we have to condition on a random variable.

▶ **Definition 2.8 (Conditional Expected Value (on a Random Variable)).** Let X and Y be random variables. The *expected value of X conditional on Y* is the *random variable* $\mathrm{E}[X \mid Y]$ such that, for all $\omega \in \Omega$, it holds that

$$\mathrm{E}[X \mid Y](\omega) = \mathrm{E}[X \mid Y = Y(\omega)] . \qquad ◀$$

Although mentioned in Definition 2.8, it is very important to stress that $\mathrm{E}[X \mid Y]$ is a random variable, not a number. This is due to the randomness in the condition. The expected value $\mathrm{E}[X \mid Y]$ can be thought of as a collection of all the conditional expected values of X with respect to the values that Y can take.

The following theorem states an important connection between an expected value conditional on a random variable and a (normal) expected value.

▶ **Theorem 2.9 (Law of Total Expectation [GS01b, Chapter 3.7, Theorem 4]).** Let X and Y be random variables. Then

$$\mathrm{E}\big[\mathrm{E}[X \mid Y]\big] = \mathrm{E}[X] . \qquad ◀$$

Theorem 2.9 is a powerful tool when calculating expected values. It allows us to consider any partition of the sample space (provided by the random variable Y), calculate the expected values conditional on each event of the partition (which is usually easier), and then taking the average. Since Y can be arbitrary, this gives us a great degree of freedom.

Note that we can express Theorem 2.9 in the following way, since we assume that Y is discrete.

▶ **Corollary 2.10 (Law of Total Expectation by Case Distinction [MU05, Lemma 2.5]).** Let X be a random variable, let $I \subseteq \mathbf{N}$, and let $\{A_i\}_{i \in I}$ be a partition of Ω. Then

$$\mathrm{E}[X] = \sum_{i \in I} \mathrm{E}[X \mid A_i\,] \cdot \Pr[A_i] \; . \qquad ◀$$

As we just mentioned, a random variable Y implicitly provides a partition of the sample space (with respect to the elementary events that are mapped to the same value). The σ-algebra of the probability space allows us then to consider all of the usual combinations of elementary events. However, different elementary events ω_1 and ω_2 may be mapped to the same value, meaning that we cannot differentiate between ω_1 and ω_2 with respect to Y. Thus, we basically consider a coarser probability space with the elementary events $\{Y = y\}$ for all $y \in \operatorname{rng}(Y)$, and the original σ-algebra may be needlessly fine-grained. This means, that we can remove Y from our considerations and instead immediately provide a σ-algebra of the desired granularity. This results in the following definition.

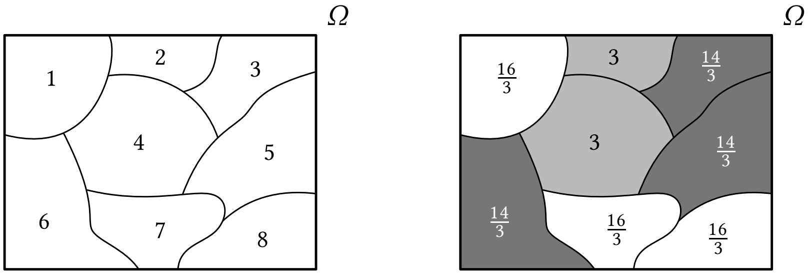
▶ **Definition 2.11 (Conditional Expected Value (on a σ-Algebra)).** Let X be a random variable and $\mathcal{B} \subseteq \mathcal{A}$ be a σ-algebra. The *expected value of* X *conditional on* $\mathcal{B}$ is the *random variable* $\mathrm{E}[X \mid \mathcal{B}\,]$ such that

(a) $\mathrm{E}[X \mid \mathcal{B}\,]$ is $\mathcal{B}$-measurable and,

(b) for all $B \in \mathcal{B}$, it holds that $\mathrm{E}\big[\mathrm{E}[X \mid \mathcal{B}\,] \cdot \mathbf{1}_B\big] = \mathrm{E}[X \cdot \mathbf{1}_B]$. ◀

We would like to mention that $\mathrm{E}[X \mid \mathcal{B}\,]$ is unique [GS01b, Chapter 7.9, Theorem 26]. For the following discussion, please also refer to Figure 2.1.

Definition 2.11 follows the aforementioned restriction of a probability space. The inclusion-minimal events of $\mathcal{B} \setminus \{\emptyset\}$ partition the original sample space Ω into new elementary events, which may result in a coarser view on the probability space, described by $\mathcal{B}$. Condition (a) now says that $\mathrm{E}[X \mid \mathcal{B}\,]$ is a random variable that can be expressed with respect to this coarser view, and condition (b) says that the expected values of $\mathrm{E}[X \mid \mathcal{B}\,]$ and X are the same when considering only events in $\mathcal{B}$. Thus, $\mathrm{E}[X \mid \mathcal{B}\,]$ can be thought of a random variable that recovers X as well as possible in terms of expectation when considering the σ-algebra $\mathcal{B}$.



(a) The random variable X. It takes a different value for each elementary event in Ω. Assume that it follows a uniform distribution.

(b) The random variable $\mathrm{E}[X \mid \mathcal{B}\,]$. Note that the expected value of $\mathrm{E}[X \mid \mathcal{B}\,]$ coincides with the expected value of X for each set of tiles with the same color.

Figure 2.1: An exemplary depiction of a random variable X (Figure 2.1 (a)) and its conditional expectation (Figure 2.1 (b)) with respect to a σ-algebra $\mathcal{B} \subset \mathcal{A}$. The eight tiles in each figure denote the elementary events of Ω. The differently colored tiles in Figure 2.1 (b) denote the elementary events with respect to $\mathcal{B}$, that is, the inclusion-minimal elements of $\mathcal{B} \setminus \{\emptyset\}$. Note that these elements partition Ω. Further note that X is not $\mathcal{B}$-measurable but $\mathrm{E}[X \mid \mathcal{B}\,]$ is.

Especially, note that $\mathrm{E}[X \mid \{\emptyset, \Omega\}\,] = \mathrm{E}[X]$,[5] that is, the expected value of X is a special case of the expected value of X when conditioning on the trivial σ-algebra $\{\emptyset, \Omega\}$ – the coarsest view possible. And when X is already $\mathcal{B}$-measurable, we get that $\mathrm{E}[X \mid \mathcal{B}\,] = X$.

We use Definition 2.11 to extend the concept of conditional probability. Let A be an event and $\mathcal{B}$ a σ-algebra. We define $\Pr[A \mid \mathcal{B}\,] \coloneqq \mathrm{E}[\mathbf{1}_A \mid \mathcal{B}\,]$, in analogy to how $\mathrm{E}[\mathbf{1}_A] = \Pr[A]$ holds for a normal expected value. Note that $\Pr[A \mid \mathcal{B}\,]$ is a random variable (taking values in $[0, 1]$).

The expected value from Definition 2.11 has all of the beneficial properties that the other expected values have.

▶ **Theorem 2.12 (Properties of the Expected Value Conditional on a σ-Algebra [GS01b, Chapter 7.9, Exercise 4][6]).** Let $a, b \in \mathbf{R}$, let X and Y be random variables, and let $\mathcal{B} \subseteq \mathcal{A}$ be σ-algebras. Then

[5] Formally, this equation is incorrect, since $\mathrm{E}[X]$ is not a random variable but a number. However, what we mean is that $\mathrm{E}[X]$ is a (deterministic) random variable that only takes the value $\mathrm{E}[X]$.

[6] The proofs can be found in the accompanying book by Grimmett and Stirzaker [GS01a].

(a) $\mathrm{E}[aX + bY \mid \mathcal{B}] = a\mathrm{E}[X \mid \mathcal{B}] + b\mathrm{E}[Y \mid \mathcal{B}]$,

(b) $\mathrm{E}\big[\mathrm{E}[X \mid \mathcal{B}]\big] = \mathrm{E}[X]$, and

(c) $\mathrm{E}\big[\mathrm{E}[X \mid \mathcal{A}] \bigm| \mathcal{B}\big] = \mathrm{E}[X \mid \mathcal{B}]$. ◀

Point (a) corresponds to Theorem 2.5; point (b) corresponds to Theorem 2.9; and point (c) says that when conditioning on multiple σ-algebras, the coarser algebra wins.

Last, we want to briefly formalize the connection between Definitions 2.8 and 2.11. In order to do so, note that σ-algebras are closed under intersection. Thus, for any random variable X, we define the σ-operator $\sigma(X)$ to be the coarsest σ-algebra such that X is still measurable, that is,

$$\sigma(X) := \bigcap \{\mathcal{B} \subseteq \mathcal{A} \mid \mathcal{B} \text{ is a } \sigma\text{-algebra and } X \text{ is } \mathcal{B}\text{-measurable}\} .$$

Note that this definition can be expanded to multiple random variables $X_0, \ldots, X_m$ by choosing the smallest σ-algebra such that all of these random variables are still measurable. Using the σ-operator, we can now interpret $\mathrm{E}[X \mid Y]$ as $\mathrm{E}[X \mid \sigma(Y)]$, which conforms with our initial motivation. Thus, we treat all conditional expected values that are random variables the same, no matter whether they are conditional on a random variable or a σ-algebra.

2.2.4 Filtrations, Adapted Processes, and Stopping Times

In the last section, we motivated the expected value of X conditional on a random variable Y by viewing it as a collection of normal conditional expected values of X, where each condition is one random outcome of Y. One example we had in mind was a single state change of a randomized algorithm. However, when analyzing a randomized algorithm, we usually look at a sequence of (random) states. At each state, we can consider the expected next state, conditional on the randomness of the current state. This results in a sequence of random outcomes that we want to condition on sequentially. Formally, this concept is modeled by a sequence of σ-algebras, where each σ-algebra denotes all of the random outcomes that could have taken place so far. Hence, this sequence is non-decreasing. The resulting sequence is known as a *filtration*.

▶ **Definition 2.13 (Filtration).** Let $(\mathcal{F}_t)_{t \in \mathbf{N}}$ be a sequence of σ-algebras. Then $\mathcal{F}$ is called a *filtration* if and only if, for all $t \in \mathbf{N}$, it holds that $\mathcal{F}_t \subseteq \mathcal{F}_{t+1}$. ◀

We call an element $\mathcal{F}_t$ of a filtration a *filter,* furthering the intuition that we take a coarser look the underlying probability space.

Continuing our example from above, let $(X_t)_{t \in \mathbf{N}}$ denote the sequence of states of a random algorithm. Using a filtration $\mathcal{F}$, we can formalize calculating the expected next state by considering $\mathrm{E}[X_{t+1} \mid \mathcal{F}_t]$. However, we are not guaranteed that $\mathcal{F}_t$ is fine-grained enough in order to entirely express the current state X_t, that is, we have no guarantee that X_t is $\mathcal{F}_t$-measurable, since we did not tie $\mathcal{F}$ to X in any way. The following definition ensures that a filtration has enough information to be able to reconstruct the current state.

▶ **Definition 2.14 (Adapted Process).** Let $(\mathcal{F}_t)_{t \in \mathbf{N}}$ be a filtration, and let $(X_t)_{t \in \mathbf{N}}$ be a sequence of random variables. Then X *is adapted to* $\mathcal{F}$ if and only if, for all $t \in \mathbf{N}$, the random variable X_t is $\mathcal{F}_t$-measurable. ◀

Note that, due to the non-decreasing nature of a filtration, Definition 2.14 implies that, for all $t \in \mathbf{N}$, the random variables $X_0, \ldots, X_t$ are $\mathcal{F}_t$-measurable.

Intuitively speaking, Definition 2.14 provides a framework for the analysis of random processes that produce a growing history of random events. It is important to note though that the definition does not prevent the filtration from having more information than necessary in order to measure X_t. For example, $(\mathcal{F}_t)_{t \in \mathbf{N}}$ where, for all $t \in \mathbf{N}$, we define $\mathcal{F}_t = \mathcal{A}$ is a filtration that every random process is adapted to, since $\mathcal{F}$ only consists of the σ-algebra $\mathcal{A}$ of the probability space. In order to restrict the information available for a certain element X_t, the respective filter has to be adjusted.

▶ **Definition 2.15 (Natural filtration).** Let $(X_t)_{t \in \mathbf{N}}$ be a sequence of random variables. The *natural filtration of* X is the filtration $\big(\sigma(X_0, \ldots, X_t)\big)_{t \in \mathbf{N}}$. ◀

The natural filtration of a process X is the smallest filtration possible for X in terms of possible σ-algebras for each X_t, due to the definition of the σ-operator. The natural filtration is useful when analyzing Markovian processes, where each state X_{t+1} is only dependent on the previous state, X_t.

Following our intuition of $(X_t)_{t \in \mathbf{N}}$ denoting states of a randomized algorithm, we can think of the index t as a point in time. We then may be interested in the first point in time T such that the algorithm reaches a certain state. For optimization algorithms, this may be a state where an optimal solution has been found. Since the algorithm is random, the states are random and, thus, the first point in time of reaching a certain state, that is, T is a random variable. Similar to how we tied a random process to a filtration by Definition 2.14, we connect the random variable T to a filtration too.

▶ **Definition 2.16 (Stopping Time).** Let $(\mathcal{F}_t)_{t \in \mathbf{N}}$ be a filtration, and let T be a random variable with $\mathrm{rng}(T) = \mathbf{N} \cup \{\infty\}$. Then T is called a *stopping time* (with respect to $\mathcal{F}$) if and only if, for all $t \in \mathbf{N}$, it holds that $\{T = t\} \in \mathcal{F}_t$. ◀

Note that, similar to Definition 2.14, due the non-decreasing property of a filtration, it follows that, for all $t \in \mathbf{N}$, we have $\{T \leq t\} \in \mathcal{F}_t$. Intuitively, a stopping time is any random variable denoting a point in time of an observation such that, at point t, it can be determined whether the observation occurred.

An important property of a stopping time T is that the decision of whether $\{T = t\}$ occurs can only be guaranteed relying on past information. For example, consider tossing a fair coin repeatedly. Let T be the random variable that denotes the first point in time such that the coin shows heads. Then T is a stopping time with respect to the natural filtration of the process, since we have, after each coin toss, the information of whether the coin shows heads or not. In contrast to that, the *last* point in time T' such that the coin shows heads is not a stopping time with respect to the natural filtration of the process, since, assuming that $\{T' = t'\}$ could be determined at t', the filter $\mathcal{F}_{t'}$ does *not* exclude the events that heads shows up again in the future (since none of these events are part of the history yet). This means that $\{T' = t'\} \notin \mathcal{F}_{t'}$. However, it is important to note that the choice of the filtration is important with respect to whether future information is allowed or not. For example, for our coin tossing example, when considering the filtration that is constantly $\mathcal{A}$, we have complete information from the very start and can fully determine T' immediately.

When we consider a stopping time T that denotes the *first* point in time such that a certain event occurs, we call T a *first-hitting time.* In Chapter 3, we go into detail about how the expected values of first-hitting times can be determined.

2.3 Probabilistic Inequalities

In this section, we state some important inequalities that we use throughout this thesis. The first inequality is useful for deriving an upper bound on the probability of the union of events. In such a case, it is sufficient to add the respective probabilities.

▶ **Theorem 2.17 (Union Bound [MU05, Lemma 1.2]).** Let $I \subseteq \mathbb{N}$, and let $\{E_i\}_{i \in I}$ be a family of events. Then

$$\Pr\left[\bigcup_{i \in I} E_i\right] \leq \sum_{i \in I} \Pr[E_i] \,. \qquad ◀$$

The next inequality states how likely it is for a nonnegative random variable to deviate from its expected value. The inequality does not necessarily yield a very strong bound, but it is widely applicable, since its only limitation is that the random variable must be nonnegative.

▶ **Theorem 2.18 (Markov's Inequality [MU05, Theorem 3.1]).** Let X be a nonnegative random variable. Then, for all $a > 0$, it holds that

$$\Pr\big[X \geq a \cdot \mathrm{E}[X]\big] \leq \frac{1}{a} \,. \qquad ◀$$

Theorem 2.18 can be used to derive far stronger bounds for more restricted scenarios. The arguably most prominent scenario considers the sum of independent, Bernoulli-distributed random variables.

▶ **Theorem 2.19 (Chernoff Bounds [MU05, Theorems 4.4 and 4.5]).** Let $n \in \mathbb{N}^+$, and let $(X_i)_{i \in [n]}$ be independent Bernoulli-distributed random variables. Further, let $X = \sum_{i \in [n]} X_i$. Then,

(a) for any $0 < \delta \leq 1$, we have $\Pr\big[X \geq (1+\delta)\mathrm{E}[X]\big] \leq \mathrm{e}^{-\frac{\mathrm{E}[X]\delta^2}{3}}$, and,

(b) for any $0 < \delta < 1$, we have $\Pr\big[X \leq (1-\delta)\mathrm{E}[X]\big] \leq \mathrm{e}^{-\frac{\mathrm{E}[X]\delta^2}{2}}$. ◀

Note that when we choose $\delta \in \Theta(1)$, Theorem 2.19 gives us exponentially low probabilities in $\mathrm{E}[X]$ to deviate from the expected value.

When considering the expected value of a random variable that is transformed by a convex function, the following inequality yields a useful bound.

▶ **Theorem 2.20 (Jensen's Inequality [MU05, Theorem 2.4]).** Let X be a random variable, and let f be a convex function. Then

$$\mathrm{E}[f(X)] \geq f\big(\mathrm{E}[X]\big) \,. \qquad ◀$$

The following theorem can be thought of as a generalization of the linearity of expectation (Theorem 2.5) when the number of random variables under consideration is subject to randomness itself.

▶ **Theorem 2.21 (Wald's Equation [GS01b, Chapter 10.2, Lemma 9]).** Let $(X_t)_{t \in \mathbf{N}^+}$ be a sequence of independent, identically distribute random variables over $\mathbf{R}$, and let T be a stopping time with respect to the natural filtration of X. Then

$$\mathrm{E}\left[\sum_{t=1}^{T} X_t\right] = \mathrm{E}[X_1] \cdot \mathrm{E}[T] \; . \qquad ◀$$

The following inequalities provide useful bounds on terms of the form $(1+p)^n$ with respect to the exponential function. Such expressions occur frequently when considering independent events.

▶ **Theorem 2.22 ([MR95, Proposition B.3]).**

(a) For all $p \in \mathbf{R}$, it holds that $1 + p \leq \mathrm{e}^p$.

(b) For all $x, n \in \mathbf{R}$ such that $n \geq 1$ and $|x| \leq n$,

$$\left(1 + \frac{x}{n}\right)^n \geq \left(1 - \frac{x^2}{n}\right)\mathrm{e}^x \; . \qquad ◀$$

Note that both cases combined imply for any $n \geq 1$ and any $x \in \mathbf{R}$ with $x^2 \in \mathrm{o}(n)$ that there exists an $a \in \mathrm{o}(1)$ such that

$$(1 - a)\mathrm{e}^{-x} \leq \left(1 - \frac{x}{n}\right)^n \leq \mathrm{e}^{-x} \; .$$

3 Drift Theory

This chapter is based on joint work with Timo Kötzing [KK18]. Different from the conference version, the results in this thesis have been phrased using filtrations in order to better align with the rest of this chapter. Further, Section 3.3.2 has been changed by simplifying the proof of Corollary 3.8 and adding a more general result (Theorem 3.7) as well as adjusting the discussion in this section with respect to these changes. Last, Theorem 3.3 as well as Sections 3.6 and 3.7 have been added.

In this chapter, we discuss *drift theory*, that is, theorems that are well suited for bounding the expected first-hitting times of random processes. We do not only aim at introducing the theorems we later use in this thesis but also at discussing the ideas behind the theorems and providing a reference of many useful drift theorems for different settings.

3.1 Introduction

Drift theory is a general term for a collection of theorems that consider random processes and bound their expected time to reach a certain value. The beauty and appeal of these theorems lie in them usually having few restrictions but yielding strong results. Intuitively speaking, in order to use a drift theorem, one only needs to estimate the expected change of a random process – the *drift* – at any given point in time. Hence, a drift theorem turns expected local changes of a process into expected first-hitting times. In other words, local information of the process is transformed into global information.

Drift theory gained traction in the theory of randomized search heuristics when it was introduced to the community by He and Yao [HY01; HY04] via the *additive drift theorem.* However, they were not the first to prove it. The result dates back to Hajek [Haj82], who stated the theorem in a fashion quite different from how it is phrased nowadays. According to Lengler [Len17], the theorem has been proven even prior to that various times. Since then, many different versions of drift theorems have been proven, the most common ones being the *variable drift theorem* [Joh10] and the *multiplicative drift theorem* [DJW12]. The different names refer to how the drift is bounded other than independent of time: *additive* means that the drift is bounded by the same value for all states; in a *multiplicative* scenario, the drift is bounded by a multiple of the current state of the process; and in the setting of *variable* drift, the drift is bounded by any monotone function with respect to the current state of the process.

Over time, the various drift theorems have been stated with different restrictions. At first, the theorems only applied to finite or discrete search spaces. However, these restrictions are rarely used in the proofs and thus not necessary, as pointed out, for example, by Lehre and Witt [LW14], who prove a general drift theorem without these restrictions. Nonetheless, one restriction that remained was a bounded search space, with a few notable exceptions that we briefly want to highlight:

- Lengler [Len17] mentions infinite search spaces and also gives a proof of the additive drift theorem in the setting of an unbounded search space. However, Lengler only considers discrete search spaces, and the drift condition only considers the previous state, not using a filtration.

- Corus et al. [Cor+14] provide a proof of an additive drift theorem over an unbounded search space in their appendix, using filtrations. However, the authors require that the expected first-hitting time of the process is finite, which is not always easy to prove and thus restricts the applicability of the theorem.

- Semenov and Terkel [ST03] state a Theorem very much like an additive drift theorem for unbounded search spaces, but they require the process to have a bounded variance, as they also prove concentration for their result.

In this section, we prove the most common drift theorems in the most general fashion up to date. Our most important results are our upper and lower bound of the classical additive drift theorem (Corollary 3.8 and Theorem 3.11, respectively), which we prove for unbounded[7] search spaces. These theorems are used as a foundation for the drift theorems in other settings. Most of our results for upper bounds require the random process to be nonnegative, which is a restriction that can be lifted, as we discuss in Section 3.3.2. In contrast to that, for the lower

[7] For the upper bound, we require the search space to be lower-bounded but not upper-bounded. We still refer to such a setting as unbounded.

bound of the additive drift theorem, we allow the process to be negative, but we have to bound the step size in return; Example 3.12 shows why our theorems fail otherwise.

Further, we prove an upper bound for the variable and the multiplicative drift theorem (Theorem 3.13 and Corollary 3.15, respectively) for unbounded search spaces, and we present a method that transforms a process with no drift into a process with positive drift (Theorem 3.18), in order to be able to still apply drift theory.

We would like to mention that drift theory has also brought forth other results than expected first-hitting times, namely, concentration bounds and negative drift, which are related. Both areas bound *the probability* of the first-hitting time taking certain values. Concentration bounds show how unlikely it is for a process to take much longer than the expected first-hitting time [DG13; Köt16]. On the other hand, negative drift bounds how likely it is for the process to reach the goal although the drift is going the opposite direction [Köt16; OW11]. Since we use negative-drift theorems in this thesis, Section 3.7 goes a bit more into detail about this topic.

3.2 Terms and Tools

We consider the expected first-hitting time T of a process $(X_t)_{t \in \mathbf{N}}$ over $\mathbf{R}$ adapted to a filtration $(\mathcal{F}_t)_{t \in \mathbf{N}}$, where T is a stopping time with respect to $\mathcal{F}$. That is, we are interested in the expected time it takes the process to reach a certain value for the first time, which we will refer to as the *target*. Usually, our target is the value 0, that is, we consider the random variable $T = \inf\{t \in \mathbf{N} \mid X_t \leq 0\}$, where we define $\inf \emptyset := \infty$.

We provide bounds on $\mathrm{E}[T \mid \mathcal{F}_0]$ with respect to the *drift* of X_t, which is defined as

$$X_t - \mathrm{E}[X_{t+1} \mid \mathcal{F}_t] \,.$$

Recall that $\mathrm{E}[T \mid \mathcal{F}_0]$ as well as $\mathrm{E}[X_{t+1} \mid \mathcal{F}_t]$ are both random variables. Because of the latter, the drift is a random variable, too. Further, note that if the drift is positive, X_t decreases its value in expectation over time when considering positive starting values. This is why 0 will be our target most of the time.

We are only interested in the process X_t until the time point T. That is, all of our requirements only need to hold for all $t < T$ (since we also consider

$t + 1$). While this phrasing is intuitive, it is formally inaccurate, as T is a random variable. We will continue to use it, however, formally, each of our inequalities in each of our requirements should be multiplied with the characteristic function of the event $\{t < T\}$. This way, the inequalities trivially hold once $t \geq T$ and, otherwise, are the inequalities we state. We go a bit more into detail about how a drift theorem without this convention looks like in Theorem 3.3 and the surrounding section.

Last, we state all of our results conditional on $\mathcal{F}_0$, that is, we bound $\mathrm{E}[T \mid \mathcal{F}_0]$. However, by the law of total expectation (Theorem 2.12, point (b)), one can easily derive a bound for $\mathrm{E}[T] = \mathrm{E}\big[\mathrm{E}[T \mid \mathcal{F}_0]\big]$.

Martingale Theorems

In this section, we state two theorems that we will use in order to prove our results in the next sections. Both theorems make use of *martingales*, a fundamental concept in the field of probability theory. A martingale is a random process with a drift of 0, that is, in expectation, it does not change over time. Further, a *supermartingale* has a drift of at least 0, that is, it decreases over time in expectation, and a *submartingale* has a drift of at most 0, that is, it increases over time in expectation.

The arguably most important theorem for martingales is the Optional-Stopping Theorem (Theorem 3.1). It is often only provided in a form that suits martingales. However, this result can be extended to super- and submartingales, as mentioned by Bhattacharya and Waymire [BW16, Remark 3.7].

▶ **Theorem 3.1 (Optional-Stopping Theorem [GS01b, Chapter 12.5, Theorem 9] and [BW16, Remark 3.7]).** Let $(X_t)_{t \in \mathbf{N}}$ be a sequence of random variables over $\mathbf{R}$ adapted to a filtration $(\mathcal{F}_t)_{t \in \mathbf{N}}$, and let T be a stopping time with respect to $\mathcal{F}$. Suppose that

(a) $\mathrm{E}[T] < \infty$ and that

(b) there is some value $c \geq 0$ such that, for all $t < T$,

$$\mathrm{E}\big[|X_{t+1} - X_t| \,\big|\, \mathcal{F}_t\big] \leq c\,.$$

Then the following holds.

1. If, for all $t < T$, we have $X_t - \mathrm{E}[X_{t+1} \mid \mathcal{F}_t] \geq 0$, then $\mathrm{E}[X_T] \leq \mathrm{E}[X_0]$.

2. If, for all $t < T$, we have $X_t - \mathrm{E}[X_{t+1} \mid \mathcal{F}_t] \leq 0$, then $\mathrm{E}[X_T] \geq \mathrm{E}[X_0]$. ◀

Theorem 3.1 allows us to bound $\mathrm{E}[X_T]$ independently of the filtration, which is why our drift results are independent of the filtration as well (except for $\mathcal{F}_0$).

Note that point 1 refers to supermartingales, whereas point 2 refers to submartingales. Intuitively, point 1 says that a supermartingale will have, in expectation, a lower value than it started with, which makes sense, as a supermartingale decreases over time in expectation. Point 2 is analogous for submartingales. For martingales, both cases can be combined in order to yield an equality.

Martingales are essential in the proofs of our theorems. We will frequently transform our process such that it results in a supermartingale or a submartingale in order to apply Theorem 3.1.

Another useful theorem for martingales is the following inequality, which is basically for martingales what a Chernoff bound (Theorem 2.19) is for binomial distributions of binary random variables. We state it in a fashion applicable for supermartingales, as mentioned by Warnke [War16, Remark 10].

▶ **Theorem 3.2 (Azuma-Hoeffding Inequality [GS01b, Chapter 12.2, Theorem 3] and [War16, Remark 10]).** Let $(X_t)_{t \in \mathbf{N}}$ be a sequence of random variables over $\mathbf{R}$ adapted to a filtration $(\mathcal{F}_t)_{t \in \mathbf{N}}$. Suppose that

(a) there is some value $c > 0$ such that, for all $t \in \mathbf{N}$, $|X_t - X_{t+1}| < c$.

If, for all $t \in \mathbf{N}$, we have $X_t - \mathrm{E}[X_{t+1} \mid \mathcal{F}_t] \geq 0$, then, for all $t \in \mathbf{N}$ and all $r > 0$,

$$\Pr[X_t - X_0 \geq r] \leq \mathrm{e}^{-\frac{r^2}{2tc^2}} . \quad ◀$$

3.3 Additive Drift

We speak of additive drift when the drift can be bounded by a value independent of the process itself. That is, the bound is spatially and time-homogeneous.

When considering the first-hitting time T of a random process $(X_t)_{t \in \mathbf{N}}$ whose drift is *lower*-bounded by a value $\delta > 0$, then $\mathrm{E}[T \mid \mathcal{F}_0]$ is *upper*-bounded by X_0/δ.[8] Interestingly, if the drift of X_t is *upper*-bounded by δ, $\mathrm{E}[T \mid \mathcal{F}_0]$ is *lower*-bounded by X_0/δ. Thus, if the drift of X_t is exactly δ, that is, we know how much expected progress X_t makes in each step, our expected first-hitting time is

[8] Note that we have access to X_0 because X is adapted to $\mathcal{F}$. Hence, X_0 is $\mathcal{F}_0$-measurable.

equal to X_0/δ. This result is remarkable, as it can be understood intuitively as follows: since we stop once X_t reaches 0, the distance from our start (X_0) to our goal (0) is exactly X_0, and we make an expected progress of δ each step. Thus, in expectation, we are done after X_0/δ steps.

3.3.1 A Very Formal Approach

Before we state our results, we provide a very formal version of our main theorem (Corollary 3.8) for $\mathrm{E}[T]$ (for the sake of simplicity). We do so because we are a bit informal in the following sections, as we already mentioned in Section 3.2. Hence, we want to provide a rigorous foundation here that shows how a drift theorem should look like if we were to mention every detail. Since many of these requirements seem unnecessarily verbose, we drop them afterward. We would like to mention that, to the best of our knowledge, the only versions of drift theorems that are this formal appear in the work of Corus et al. [Cor+14].

▶ **Theorem 3.3 (Upper Additive Drift, Unbounded, Very Formal).** Let $(X_t)_{t\in\mathbf{N}}$ be random variables over $\mathbf{R}$ adapted to a filtration $(\mathcal{F}_t)_{t\in\mathbf{N}}$, and let $T = \inf\{t \in \mathbf{N} \mid X_t \leq 0\}$. Furthermore, suppose that,

(a) there is some value $\delta > 0$ such that, for all $t \in \mathbf{N}$, it holds that

$$\mathrm{E}[(X_t - X_{t+1}) \cdot \mathbf{1}\{t < T\} \mid \mathcal{F}_t] \geq \delta \cdot \mathbf{1}\{t < T\}\text{ , and that,}$$

(b) for all $t \in \mathbf{N}$, it holds that $X_t \cdot \mathbf{1}\{t \leq T\} \geq 0$.

Then

$$\mathrm{E}[T] \leq \frac{\mathrm{E}[X_0]}{\delta}\ .$$ ◀

Note that the random variable $\mathbf{1}\{t < T\}$ is $\mathcal{F}_t$-measurable, since T is a stopping time with respect $\mathcal{F}$.

The proof of Theorem 3.3 closely follows Lengler [Len17, Theorem 1], who uses a very short and elegant method but does not consider drift in the setting of filtrations.

Proof of Theorem 3.3. We first aim at showing that the following inequality holds for all $t \in \mathbf{N}$:

$$\mathrm{E}[X_{t+1} \mid t < T] \leq \mathrm{E}[X_t \mid t < T] - \delta\ . \qquad (3.1)$$

Note that this inequality is *not* a simple transformation of condition (a), because inequality (3.1) considers expected values that are numbers, not random variables. We get

$$\begin{aligned}
\mathrm{E}[X_{t+1} \mid t < T] &= \mathrm{E}[X_{t+1} \cdot \mathbf{1}\{t < T\} \mid t < T] \\
&\overset{(*)}{=} \mathrm{E}\big[\mathrm{E}[X_{t+1} \cdot \mathbf{1}\{t < T\} \mid \mathcal{F}_t] \bigm| t < T\big] \\
&\overset{(\dagger)}{\leq} \mathrm{E}\big[\mathrm{E}[X_t \cdot \mathbf{1}\{t < T\} \mid \mathcal{F}_t] - \delta \cdot \mathbf{1}\{t < T\} \bigm| t < T\big] \\
&\overset{\ddagger}{=} \mathrm{E}[X_t \cdot \mathbf{1}\{t < T\} \mid t < T] - \mathrm{E}[\delta \cdot \mathbf{1}\{t < T\} \mid t < T] \\
&= \mathrm{E}[X_t \mid t < T] - \delta\,,
\end{aligned}$$

where steps (∗) and (‡) use Theorem 2.12, point (b), step (†) uses condition (a) of Theorem 3.3, and (‡) additionally uses Theorem 2.9.

By Corollary 2.10, we get, for all $t \in \mathbf{N}$, that

$$\begin{aligned}
\mathrm{E}[X_t] &= \mathrm{E}[X_t \mid t < T] \cdot \Pr[t < T] + \overbrace{\mathrm{E}[X_t \mid t \geq T]}^{=0} \cdot \Pr[t \geq T] \\
&= \mathrm{E}[X_t \mid t < T] \cdot \Pr[t < T]\,. \qquad (3.2)
\end{aligned}$$

Using Corollary 2.10, inequality (3.1) and equation (3.2), we get

$$\begin{aligned}
\mathrm{E}[X_{t+1}] &= \mathrm{E}[X_{t+1} \mid t < T] \cdot \Pr[t < T] + \overbrace{\mathrm{E}[X_{t+1} \mid t \geq T]}^{=0} \cdot \Pr[t \geq T] \\
&\overset{(3.1)}{\leq} (\mathrm{E}[X_t \mid t < T] - \delta) \cdot \Pr[t < T] \\
&\overset{(3.2)}{=} \mathrm{E}[X_t] - \delta \cdot \Pr[t < T]\,. \qquad (3.3)
\end{aligned}$$

Since T is a nonnegative integer random variable, it holds for its expected value that $\mathrm{E}[T] = \sum_{t=0}^{\infty} \Pr[t < T]$ [MR95, Proposition C.7]. By using the definition of an infinite sum, we get

$$\begin{aligned}
\delta \cdot \Pr[t < T] &\overset{\tau \to \infty}{\longleftarrow} \sum_{t=0}^{\tau} \delta \cdot \Pr[t < T] \\
&\overset{(3.3)}{\leq} \sum_{t=0}^{\tau} (\mathrm{E}[X_t] - \mathrm{E}[X_t + 1])
\end{aligned}$$

$$
\begin{aligned}
&= \mathrm{E}[X_0] - \overbrace{\mathrm{E}[X_{\tau+1}]}^{\geq 0} \\
&\leq \mathrm{E}[X_0] \ .
\end{aligned}
$$

Dividing by δ finishes the proof. ■

In the following, we omit the indicator random variables and only mention them indirectly via universal quantifiers.

3.3.2 Upper Bounds

We give a proof for the additive drift theorem, originally published (in a more restricted version) by He and Yao [HY01; HY04]. We start by reproving the original theorem (which requires a bounded search space) but in a simpler, more elegant and educational manner. We then extend this result by generalizing it to processes with a bounded step size. Finally, we lift also this restriction.

Commonly, additive drift theorems are stated for nonnegative processes. However, we show that this restriction is not necessary. Nonetheless, we state all of our results only for nonnegative processes, except for our most general result (Theorem 3.7). The reason for this is that the result for nonnegative processes takes an easier form than the more general one. We continue to discuss details of this difference throughout this section

In all of the following proofs, we transforms the process we consider into a supermartingale and then use Theorem 3.1 (or a variant). However, in order to apply Theorem 3.1, we have to make sure to fulfill condition (a), which is the hardest part.

▶ **Theorem 3.4 (Upper Additive Drift, Bounded).** Let $(X_t)_{t\in\mathbf{N}}$ be random variables over $\mathbf{R}$ adapted to a filtration $(\mathcal{F}_t)_{t\in\mathbf{N}}$, and let $T = \inf\{t \in \mathbf{N} \mid X_t \leq 0\}$. Furthermore, suppose that

(a) there is some value $\delta > 0$ such that, for all $t < T$, it holds that

$$X_t - \mathrm{E}[X_{t+1} \mid \mathcal{F}_t] \geq \delta \text{ , that,}$$

(b) for all $t \leq T$, it holds that $X_t \geq 0$, and that

(c) there is some value $c \geq 0$ such that, for all $t < T$, it holds that $X_t \leq c$.

Then

$$\mathrm{E}[T \mid \mathcal{F}_0] \leq \frac{X_0}{\delta} \,. \qquad \blacktriangleleft$$

Note that condition (b) means that T can be rewritten as $\inf\{t \in \mathbf{N} \mid X_t = 0\}$, that is, we have to hit 0 exactly in order to stop. As we discuss after the proof, this restriction is not necessary when adjusting the result of the theorem slightly.

Condition (a) bounds the expected progress we make each time step. The larger δ, the lower the expected first-hitting time. However, due to condition (b), note that small values of X_t create a natural upper bound for δ, as the progress for such values can be at most $|X_t - 0| = X_t$.

Condition (c) means that we are considering random variables over the interval $[0, c]$. It is a restriction that all previous additive drift theorems have but that is actually not necessary, as we show with Corollary 3.8. In the following proof, we use this condition in order to show that $\mathrm{E}[T] < \infty$, which is necessary when applying Theorem 3.1.

Proof of Theorem 3.4. We want to use point 1 of Theorem 3.1. Thus, we define, for all $t < T$, the process $Y_t := X_t + \delta t$, which is a supermartingale, since

$$\begin{aligned} Y_t - \mathrm{E}[Y_{t+1} \mid \mathcal{F}_t] &= X_t + \delta t - \mathrm{E}[X_{t+1} + \delta(t+1) \mid \mathcal{F}_t] \\ &= X_t - \mathrm{E}[X_{t+1} \mid \mathcal{F}_t] - \delta \\ &\geq 0 \,, \end{aligned}$$

as we assume that $X_t - \mathrm{E}[X_{t+1} \mid \mathcal{F}_t] \geq \delta$ for all $t < T$.

We now show that $\mathrm{E}[T \mid \mathcal{F}_0] < \infty$ holds in order to apply Theorem 3.1. Let $r > 0$, and let a be any value such that $\Pr[X_0 \leq a] > 0$. We condition on the event $\{X_0 \leq a\}$, and we consider a time point $t' = (a + r)/\delta$ and want to bound the probability that $X_{t'}$ has not reached 0 yet, that is, we consider the event $\{X_{t'} > 0\}$. We rewrite this event as $\{X_{t'} - a > -a\}$, which is equivalent to $\{Y_{t'} - a > -a + \delta t' = r\}$, by definition of Y and t'.

Note that, for all $t < T$, we have $|Y_t - Y_{t+1}| < c + \delta + 1$, as we assume that $X_t \leq c$. Thus, the differences of Y_t are bounded and we can apply Theorem 3.2 as follows, noting that $Y_0 = X_0 \leq a$, due to our condition on $\{X_0 \leq a\}$:

$$\begin{aligned} \Pr[Y_{t'} - a > r \mid X_0 \leq a] &\leq \Pr[Y_{t'} - Y_0 \geq r \mid X_0 \leq a] \\ &\leq \mathrm{e}^{-\frac{r^2}{2t'(c+\delta+1)^2}} \,. \end{aligned}$$

If we choose $r \geq a$, we get $t' \leq 2r/\delta$ and, thus,

$$\Pr[Y_{t'} - Y_0 > r \mid X_0 \leq a] \leq \mathrm{e}^{-\frac{r\delta}{4(c+\delta+1)^2}} .$$

This means that the probability that $X_{t'}$ has not reached 0 goes exponentially fast toward 0 as t' (and, hence, r) goes toward ∞. Thus, the expected value of T is finite.

Now we can use point 1 of Theorem 3.1 in order to get $\mathrm{E}[Y_T \mid \mathcal{F}_0] \leq \mathrm{E}[Y_0 \mid \mathcal{F}_0]$. In particular, noting that $X_T = 0$ by definition,

$$\begin{aligned} X_0 &= \mathrm{E}[X_0 \mid \mathcal{F}_0] \\ &= \mathrm{E}[Y_0 \mid \mathcal{F}_0] \\ &\geq \mathrm{E}[Y_T \mid \mathcal{F}_0] \\ &= \mathrm{E}[X_T + \delta T \mid \mathcal{F}_0] \\ &= \mathrm{E}[X_T \mid \mathcal{F}_0] + \delta \mathrm{E}[T \mid \mathcal{F}_0] \\ &= \delta \mathrm{E}[T \mid \mathcal{F}_0] . \end{aligned}$$

Thus, we get the desired bound by dividing by δ. ■

Note that the proof reveals that condition (b) of Theorem 3.4 can be ignored in order to get the result

$$\mathrm{E}[T \mid \mathcal{F}_0] \leq \frac{X_0 - \mathrm{E}[X_T \mid \mathcal{F}_0]}{\delta} . \tag{3.4}$$

This result is useful when one can get a lower bound on $\mathrm{E}[X_T \mid \mathcal{F}_0]$. However, determining this value may not be easy. Hence, the result is simplified to the form seen in Theorem 3.4. Further note that we are able to achieve this general result because none of the tools we use (that is, Theorems 3.1 and 3.2) require the process to be nonnegative.

Since the arguments in the proof of Theorem 3.4 only need the property of bounded differences in order to apply Theorem 3.2, we can relax the condition of a bounded state space into bounded step size, which can be seen in the following theorem.

▶ **Theorem 3.5 (Upper Additive Drift, Bounded Step Size).** Let $(X_t)_{t \in \mathbf{N}}$ be random variables over $\mathbf{R}$ adapted to a filtration $(\mathcal{F}_t)_{t \in \mathbf{N}}$, and let $T = \inf\{t \in \mathbf{N} \mid X_t \leq 0\}$. Furthermore, suppose that

(a) there is some value $\delta > 0$ such that, for all $t < T$, it holds that

$$X_t - \mathrm{E}[X_{t+1} \mid \mathcal{F}_t] \geq \delta \text{ , that,}$$

(b) for all $t \leq T$, it holds that $X_t \geq 0$, and that

(c) there is some value $c \geq 0$ such that, for all $t < T$, we have $|X_{t+1} - X_t| \leq c$.

Then

$$\mathrm{E}[T \mid \mathcal{F}_0] \leq \frac{X_0}{\delta} \text{ .}$$ ◀

Proof. The proof of Theorem 3.4 is also applicable here. ■

Note that the result of Theorem 3.5 can also be generalized to inequality (3.4), since the proof is the same as the one of Theorem 3.4.

We now state an even more general additive drift theorem, which drops condition (c) from Theorem 3.5. However, this means that we cannot rely on Theorem 3.2 anymore and need another proof strategy. In the paper that this chapter is based on [KK18], we used Markov's inequality (Theorem 2.18) and bounded the step size probabilistically in order to apply Theorem 3.5. While this approach is feasible, it is quite involved, and there is an easier proof, which we are going to present here. This proof makes use of the following special form of the Optional-Stopping Theorem (Theorem 3.1).

▶ **Theorem 3.6 (Optional-Stopping Theorem for Nonnegative Supermartingales [Dur19, Theorem 4.8.4]).** Let $(X_t)_{t\in\mathbf{N}}$ be a sequence of random variables over $\mathbf{R}$ adapted to a filtration $(\mathcal{F}_t)_{t\in\mathbf{N}}$, and let T be a stopping time with respect to $\mathcal{F}$. Suppose that,

(a) for all $t < T$, we have $X_t - \mathrm{E}[X_{t+1} \mid \mathcal{F}_t] \geq 0$, and that,

(b) for all $t \leq T$, it holds that $X_t \geq 0$.

Then $\mathrm{E}[X_T] \leq \mathrm{E}[X_0]$. ◀

Note that the two conditions of Theorem 3.6 state that the considered process is a nonnegative supermartingale. The advantage of this theorem over Theorem 3.1 is that we neither have to show first that the expected first-hitting time is finite nor that the process has a bounded expected step size. This allows us to easily prove the following theorem, which is the most general additive-drift theorem up to date. Different to most common drift theorems, this theorem allows to overshoot the target by a certain amount. In return, a penalty term is added to the expected first-hitting time.

▶ **Theorem 3.7 (General Upper Additive Drift, Unbounded).** Let $a \leq 0$, let $(X_t)_{t \in \mathbf{N}}$ be random variables over $\mathbf{R}$ adapted to a filtration $(\mathcal{F}_t)_{t \in \mathbf{N}}$, and let $T = \inf\{t \in \mathbf{N} \mid X_t \leq 0\}$. Furthermore, suppose that

(a) there is some value $\delta > 0$ such that, for all $t < T$, it holds that

$$X_t - \mathrm{E}[X_{t+1} \mid \mathcal{F}_t] \geq \delta \text{ and that,}$$

(b) for all $t \leq T$, it holds that $X_t \geq a$.

Then

$$\begin{aligned} \mathrm{E}[T \mid \mathcal{F}_0] &\leq \frac{X_0 - \mathrm{E}[X_T \mid \mathcal{F}_0]}{\delta} \\ &\leq \frac{X_0 - a}{\delta} . \end{aligned}$$ ◀

Proof. We follow the proof idea of Theorem 3.4 but use Theorem 3.6 instead of Theorem 3.1. Thus, we do not have to show that $\mathrm{E}[T]$ is finite.

We define for all $t < T$ the process $Y_t := X_t - a + \delta t$, which is a supermartingale, since

$$\begin{aligned} Y_t - \mathrm{E}[Y_{t+1} \mid \mathcal{F}_t] &= X_t - a + \delta t - \mathrm{E}[X_{t+1} - a + \delta(t+1) \mid \mathcal{F}_t] \\ &= X_t - \mathrm{E}[X_{t+1} \mid \mathcal{F}_t] - \delta \\ &\geq 0 , \end{aligned}$$

as we assume that $X_t - \mathrm{E}[X_{t+1} \mid \mathcal{F}_t] \geq \delta$ for all $t < T$.

Note that Y is nonnegative, as, for all $t \leq T$, it holds due to condition (b) that

$$\begin{aligned} Y_t &= X_t - a + \delta t \\ &\geq X_t - a \\ &\geq 0 . \end{aligned}$$

Hence, we can apply Theorem 3.6 and get

$$\begin{aligned} X_0 - a &= \mathrm{E}[X_0 - a \mid \mathcal{F}_0] \\ &= \mathrm{E}[Y_0 \mid \mathcal{F}_0] \\ &\geq \mathrm{E}[Y_T \mid \mathcal{F}_0] \\ &= \mathrm{E}[X_T - a + \delta T \mid \mathcal{F}_0] \\ &= \mathrm{E}[X_T \mid \mathcal{F}_0] - a + \delta \mathrm{E}[T \mid \mathcal{F}_0] . \end{aligned}$$

Solving this inequality for $\mathrm{E}[T]$ yields the first result of the theorem. Using the bound $X_T \geq a$ and thus $\mathrm{E}[X_T \mid \mathcal{F}_0] \geq a$ yields the second result and concludes the proof. ■

As before, we state Theorem 3.7 with respect to nonnegative processes by choosing $a = 0$, which is the usual form for additive drift theorems.

▶ **Corollary 3.8 (Upper Additive Drift, Unbounded).** Let $(X_t)_{t \in \mathbf{N}}$ be random variables over $\mathbf{R}$ adapted to a filtration $(\mathcal{F}_t)_{t \in \mathbf{N}}$, and let $T = \inf\{t \in \mathbf{N} \mid X_t \leq 0\}$. Furthermore, suppose that

(a) there is some value $\delta > 0$ such that, for all $t < T$, it holds that

$$X_t - \mathrm{E}[X_{t+1} \mid \mathcal{F}_t] \geq \delta \text{ and that,}$$

(b) for all $t \leq T$, it holds that $X_t \geq 0$.

Then

$$\mathrm{E}[T \mid \mathcal{F}_0] \leq \frac{X_0}{\delta} .$$ ◀

Note that any random processs X may be transformed into one only taking nonnegative values by defining $Y = \max\{X, 0\}$. Since Corollary 3.8 does not have the additional term of $-a$ in its result, it may seem that it makes more sense to always transform a process into a nonnegative one and then apply Corollary 3.8 instead of Theorem 3.7. However, it is important to note that such a transformation may affect the drift. In general, the drift of X will always be a lower bound for the drift of Y, since the former can take values less than 0 whereas the latter is bounded by 0. Thus the expected differences of X may be smaller than those of Y. Consequently, the result of Theorem 3.7 benefits of this (potentially) lower bound on the drift when compared to Corollary 3.8. The following example highlights this difference.

▶ **Example 3.9.** Let $n > 0$, and let $(X_t)_{t \in \mathbf{N}}$ be a random process with $X_0 = 1$ and, for all $t \in \mathbf{N}$, $X_{t+1} = X_t$ with probability $1 - 1/n$, and $X_{t+1} = -n + 1$ otherwise. Let T denote the first point in time t such that the event $\{X_t \leq 0\}$ occurs. Since T follows a geometric distribution with success probability $1/n$, we have $\mathrm{E}[T \mid X_0] = n$. We now upper-bound $\mathrm{E}[T]$ using Theorem 3.7 and Corollary 3.8. We start with the former.

We have, for all $t < T$, that $X_t - \mathrm{E}[X_{t+1} \mid X_0, \ldots, X_t] = 1$ and, thus, $\mathrm{E}[T \mid X_0] \leq (1 - (-n + 1))/1 = n$. We now consider Corollary 3.8.

For all $t < T$, let $Y_t = \max\{X_t, 0\}$. Thus, we get $Y_t - \mathrm{E}[Y_{t+1} \mid Y_0, \ldots, Y_t] = 1/n$ and consequently $\mathrm{E}[T \mid X_0] \leq 1/(1/n) = n$. This is the same bound as before. However, in the second case, the drift changed but no additional term was added in the result. ◀

Further, it is important to note that cutting off a process at 0 may result in Corollary 3.8 not being applicable at all whereas Theorem 3.7 is, as the following example shows.

▶ **Example 3.10.** Let $(X_t)_{t \in \mathbf{N}}$ be a random process with $X_0 = 1$ and, for all $t \in \mathbf{N}$, $X_{t+1} = X_t/2$ with probability $1/2$, and $X_{t+1} = (3/2)X_t - 2$ otherwise. Let T denote the first point in time t such that the event $\{X_t \leq 0\}$ occurs. Since T follows a geometric distribution with success probability $1/2$, we have $\mathrm{E}[T \mid X_0] = 2$.

Note that if we consider $Y := \max\{X, 0\}$, Corollary 3.8 is not applicable, since this process approaches 0 in the limit. However, Theorem 3.7 is applicable to X, as the process has a drift of 1. By noting that the process can never go below -2, we get $\mathrm{E}[T \mid X_0] \leq (1 - (-2))/1 = 3$, which is worse than the actual value of $\mathrm{E}[T \mid X_0]$, since the bound of -2 is rather pessimistic. Noting that, for all $t \in \mathbf{N}^+$, the random variable X_T takes the value $(3/2) \cdot (1/2)^{t-1} - 2$ with probability $(1/2)^t$, we see that $\mathrm{E}[X_T] = -1$ and consequently $\mathrm{E}[T \mid X_0] \leq (1 - (-1))/1 = 2$, which is the actual value. Overall, Theorem 3.7 can bound the expected first-hitting time tightly, whereas Corollary 3.8 is not applicable. ◀

It remains an open problem whether condition (b) of Theorem 3.7 is necessary or not. We were not able to find any counterexample. However, showing this more general result proves difficult, as introducing a lower bound (and then applying Theorem 3.7) can affect the drift, as the two examples above show.

3.3.3 Lower Bound

In this section, we provide a lower bound for the expected first-hitting time under additive drift. In order to do so, we need an upper bound for the drift. Since we now lower-bound the first-hitting time, a large upper bound of the drift makes the result bad. Thus, we do not require a lower bound on the search space, as larger differences can only increase the drift's upper bound. However, we need to have some restriction on the step size in order to make sure not to move away from the target. We provide an example (Example 3.12) showing this necessity at the end of this section.

▶ **Theorem 3.11 (Lower Additive Drift, Expected Bounded Step Size).** Let $(X_t)_{t \in \mathbf{N}}$ be random variables over $\mathbf{R}$ adapted to a filtration $(\mathcal{F}_t)_{t \in \mathbf{N}}$, and let $T = \inf\{t \in \mathbf{N} \mid X_t \leq 0\}$. Furthermore, suppose that

(a) there is some value $\delta > 0$ such that, for all $t < T$, it holds that

$$X_t - \mathrm{E}[X_{t+1} \mid \mathcal{F}_t] \leq \delta \text{ , and that}$$

(b) there is some value $c \geq 0$ such that, for all $t < T$, it holds that

$$\mathrm{E}\big[|X_{t+1} - X_t| \,\big|\, \mathcal{F}_t\big] \leq c \ .$$

Then

$$\mathrm{E}[T \mid \mathcal{F}_0] \geq \frac{X_0}{\delta} \ .$$ ◀

Proof. We make a case distinction with respect to $\mathrm{E}[T \mid \mathcal{F}_0]$ being finite. If $\mathrm{E}[T \mid \mathcal{F}_0]$ is infinite, then the theorem trivially holds. Thus, we now assume that $\mathrm{E}[T \mid \mathcal{F}_0] < \infty$.

Similar to the proof of Theorem 3.4, we define, for all $t < T$, $Y_t = X_t + \delta t$, which is a submartingale, since

$$\begin{aligned} Y_t - \mathrm{E}[Y_{t+1} \mid \mathcal{F}_0] &= X_t - \delta t - \mathrm{E}[X_{t+1} - \delta(t+1) \mid \mathcal{F}_t] \\ &= X_t - \mathrm{E}[X_{t+1} \mid \mathcal{F}_t] - \delta \\ &\leq 0 \ , \end{aligned}$$

as we assume that $X_t - \mathrm{E}[X_{t+1} \mid \mathcal{F}_t] \leq \delta$ for all $t < T$.

Since we now assume that $\mathrm{E}[T \mid \mathcal{F}_0] < \infty$ and that $\mathrm{E}\big[|X_{t+1} - X_t| \bigm| \mathcal{F}_t\big] \le c$ for all $t < T$, we can directly apply point 2 of Theorem 3.1 and get that $\mathrm{E}[Y_T \mid \mathcal{F}_0] \geq \mathrm{E}[Y_0 \mid \mathcal{F}_0]$. This yields, noting that $X_T \le 0$,

$$\begin{aligned} X_0 &= \mathrm{E}[X_0 \mid \mathcal{F}_0] \\ &= \mathrm{E}[Y_0 \mid \mathcal{F}_0] \\ &\le \mathrm{E}[Y_T \mid \mathcal{F}_0] \\ &= \mathrm{E}[X_T + \delta T \mid \mathcal{F}_0] \\ &= \mathrm{E}[X_T \mid \mathcal{F}_0] + \delta \mathrm{E}[T \mid \mathcal{F}_0] \\ &\le \delta \mathrm{E}[T \mid \mathcal{F}_0] . \end{aligned}$$

Thus, we get the desired bound by dividing by δ. ■

Note that, similar to what we discussed after Theorem 3.4, the prove reveals that we can alternatively state the bound

$$\mathrm{E}[T \mid \mathcal{F}_0] \ge \frac{X_0 - \mathrm{E}[X_T \mid \mathcal{F}_0]}{\delta} .$$

Thus, any upper bound on $\mathrm{E}[X_T \mid \mathcal{F}_0]$ yields a valid result, Theorem 3.11 only chooses the most pessimistic bound.

As we already mentioned, the step size of the process has to be bounded in some way for a lower bound for the expected first-hitting time, as the following example shows.

▶ **Example 3.12.** Let $\delta \in (0, 1)$, and let $(X_t)_{t\in\mathbf{N}}$ be a random process with $X_0 = 2$ and, for all $t \in \mathbf{N}$, $X_{t+1} = 0$ with probability $1/2$ and $X_{t+1} = 2X_t - 2\delta$ otherwise. Further, let T denote the first point in time t such that $X_t = 0$. Then T follows a geometric distribution with success probability $1/2$, which yields $\mathrm{E}[T] = 2$. However, we have that $X_t - \mathrm{E}[X_{t+1} \mid X_0, \ldots, X_t] = \delta$. If Theorem 3.11 could be applied to this process (by neglecting the condition of the bounded step size), the theorem would yield that $\mathrm{E}[T] \ge 2/\delta$, which is not true. ◀

3.4 Variable Drift

In contrast to additive drift, *variable* drift means that the drift can depend on the current state of the process (while still being bounded independently of the time). Interestingly, these more flexible drift theorems can be derived by using additive drift. Intuitively, the reasoning behind this approach is to scale the search space such that the information relevant to the process's history cancels out. As a side note, we would like to mention that Lengler [Len17] notes that considering the process $X_t \coloneqq \mathrm{E}[T \mid \mathcal{F}_t] - t$ always yields a drift of 1.[9] This means that additive drift (in theory) can always be used in order to get a tight result. However, determining the value of $\mathrm{E}[T \mid \mathcal{F}_t] - t$ is usually hard. Hence, the variable drift theorem can be thought of as a handy approximation of the process defined above.

The first variable drift theorem was proven by Johannsen [Joh10] and, independently in a different version, by Mitavskiy et al. [MRC09]. It was later refined by Rowe and Sudholt [RS14]. In all of these versions, bounded search spaces were used. Due to Corollary 3.8, we can drop this restriction.

It is important to note that variable drift theorems are commonly phrased such that the first-hitting time T denotes the first point in time such that the random process drops *below* a certain value (our target) – it is not enough to hit that value. However, this restriction is not always necessary. Thus, we also consider the setting from Section 3.3, where T denotes the first point in time such that we *hit* our target. In this section, our target is no longer 0 but a value $x_{\min}$.

In all of our theorems in this section, we make use of a set D. This set contains (at least) all possible values that our process can take while not having reached the target yet. It is a formal necessity in order to calculate the bound of the first-hitting time (via an integral). However, when applying the theorem, it is usually sufficient to choose $D = \mathbf{R}$ or $D = \mathbf{R}_{\geq 0}$.

3.4.1 Below the Target

The following version of the theorem assumes that the process has to drop below the target, denoted by $x_{\min}$. We provide the other version afterward.

▶ **Theorem 3.13 (Upper Variable Drift, Unbounded, Below Target).** Let $(X_t)_{t \in \mathbf{N}}$ be random variables over $\mathbf{R}$ adapted to a filtration $(\mathcal{F}_t)_{t \in \mathbf{N}}$, let $x_{\min} > 0$, and let $T = \inf\{t \in \mathbf{N} \mid X_t < x_{\min}\}$. Additionally, let D denote the smallest real interval that contains at least all values $x \geq x_{\min}$ that, for all $t \leq T$, any X_t can take. Furthermore, suppose that

[9] This can be checked using point (c) of Theorem 2.12.

(a) there is a monotonically increasing function $h\colon D \longrightarrow \mathbf{R}^+$ such that, for all $t < T$, it holds that

$$X_t - \mathrm{E}[X_{t+1} \mid \mathcal{F}_t] \geq h(X_t)\text{ , that}$$

(b) $X_0 \geq x_{\min}$ and, for all $t \leq T$, it holds that $X_t \geq 0$.

Then

$$\mathrm{E}[T \mid \mathcal{F}_0] \leq \frac{x_{\min}}{h(x_{\min})} + \int_{x_{\min}}^{X_0} \frac{1}{h(z)} \mathrm{d}z\ . \qquad ◀$$

Proof. The proof follows the one given by Rowe and Sudholt [RS14] very closely. We define a function $g\colon D \cup [0, x_{\min}] \longrightarrow \mathbf{R}_{\geq 0}$ as follows:

$$g(x) = \begin{cases} 0 & \text{if } x < x_{\min}, \\ \frac{x_{\min}}{h(x_{\min})} + \int_{x_{\min}}^{x} \frac{1}{h(z)} \mathrm{d}z & \text{else.} \end{cases}$$

Note that g is well-defined, since $1/h$ is monotonically decreasing and every monotone function is integrable over all compact intervals of its domain [Rud76, Theorem 6.9]. Further, $g(X_t) = 0$ holds if and only if $X_t < x_{\min}$. Thus, both processes have the same first-hitting time.

Assume that $x \geq y \geq x_{\min}$. We get

$$\begin{aligned} g(x) - g(y) &= \int_{y}^{x} \frac{1}{h(z)} \mathrm{d}z \\ &\geq \frac{x-y}{h(x)}\ , \end{aligned}$$

since h is monotonically increasing. Assuming $y \geq x \geq x_{\min}$, we get, similar to before,

$$\begin{aligned} g(x) - g(y) &= -\int_{x}^{y} \frac{1}{h(z)} \mathrm{d}z \\ &\geq -\frac{y-x}{h(x)} \\ &= \frac{x-y}{h(x)}\ . \end{aligned}$$

Thus, we can write, for $x \geq x_{\min}$ and $y \geq x_{\min}$,

$$g(x) - g(y) \geq \frac{x-y}{h(x)} \,.$$

Further, for $x \geq x_{\min} > y \geq 0$, we get

$$\begin{aligned} g(x) - g(y) &= \frac{x_{\min}}{h(x_{\min})} + \int_{x_{\min}}^{x} \frac{1}{h(z)} \mathrm{d}z \\ &\geq \frac{x_{\min}}{h(x)} + \frac{x - x_{\min}}{h(x)} \\ &= \frac{x}{h(x)} \\ &\geq \frac{x-y}{h(x)} \,. \end{aligned}$$

Overall, for $x \geq x_{\min}$ (including $X_0 \geq x_{\min}$) and $y \in \mathbf{R}_{\geq 0}$, we can estimate

$$g(x) - g(y) \geq \frac{x-y}{h(x)} \,.$$

We use this to determine the drift of the process $g(X_t)$ as follows:

$$\begin{aligned} g(X_t) - \mathrm{E}[g(X_{t+1}) \mid \mathcal{F}_t] &= \mathrm{E}[g(X_t) - g(X_{t+1}) \mid \mathcal{F}_t] \\ &\geq \frac{\mathrm{E}[X_t - X_{t+1} \mid \mathcal{F}_t]}{h(X_t)} \\ &\geq 1 \,, \end{aligned}$$

where we used the condition on the drift of X.

An application of Corollary 3.8 completes the proof. ■

3.4.2 Hitting the Target

As mentioned before, it is not always necessary to drop below the target. For the additive drift, for example, we are interested in the first time reaching the target. Interestingly, the proof for the following theorem is straightforward, as it is almost the same as the proof of Theorem 3.13. Intuitively, the waiting time for getting below the target, once it is reached, is eliminated from the expected first-hitting time. However, it is important to note that it is now not allowed to get below the target.

▶ **Theorem 3.14 (Upper Variable Drift, Unbounded, Hitting Target).** Let $(X_t)_{t \in \mathbf{N}}$ be random variables over $\mathbf{R}$ adapted to a filtration $(\mathcal{F}_t)_{t \in \mathbf{N}}$, let $x_{\min} \geq 0$, and let $T = \inf\{t \in \mathbf{N} \mid X_t \leq x_{\min}\}$. Additionally, let D denote the smallest real interval that contains at least all values $x \geq x_{\min}$ that, for all $t \leq T$, any X_t can take. Furthermore, suppose that

(a) there is a monotonically increasing function $h\colon D \to \mathbf{R}^+$ such that, for all $t < T$, it holds that

$$X_t - \mathrm{E}[X_{t+1} \mid \mathcal{F}_t] \geq h(X_t) \text{ and that,}$$

(b) for all $t \leq T$, it holds that $X_t \geq x_{\min}$..

Then

$$\mathrm{E}[T \mid \mathcal{F}_0] \leq \int_{x_{\min}}^{X_0} \frac{1}{h(z)} \mathrm{d}z\ . \qquad ◀$$

Proof. This proof is almost identical to the proof of Theorem 3.13. The difference is that we define our potential function $g\colon D \to \mathbf{R}_{\geq 0}$ as follows:

$$g(x) = \begin{cases} 0 & \text{if } x \leq x_{\min}, \\ \int_{x_{\min}}^{x} \frac{1}{h(z)} \mathrm{d}z & \text{else.} \end{cases}$$

As for $g(x) - g(y)$, the case $x \geq x_{\min} > y$ does not exist anymore, since we cannot get below $x_{\min}$. Thus, the potential difference is the same in all cases, and nothing changes in the rest of the proof. ■

3.5 Multiplicative Drift

A special case of variable drift is *multiplicative* drift, where the drift can be bounded by a multiple of the most recent value in the history of the process. As before, we provide upper bounds in the two versions of either dropping below the target or hitting it. In this setting, it can be intuitively argued why the version of dropping below the target is useful: consider a sequence of nonnegative numbers that halves its current value each time step. This process will never reach 0 within finite time. However, it drops below any value greater than 0.

Both upper bounds we state are simple applications of the corresponding variable drift theorems from Section 3.4.

3.5.1 Below the Target

Corollary 3.15 has first been stated by Doerr et al. [DJW12] using finite state spaces. Afterward, it has been proven multiple times for processes not requiring an upper bound (although this is not always stated) [DG13; LS18; LW13].

▶ **Corollary 3.15 (Upper Multiplicative Drift, Unbounded, Below Target).** Let $(X_t)_{t \in \mathbf{N}}$ be random variables over $\mathbf{R}$ adapted to a filtration $(\mathcal{F}_t)_{t \in \mathbf{N}}$, let $x_{\min} > 0$, and let $T = \inf\{t \in \mathbf{N} \mid X_t < x_{\min}\}$. Furthermore, suppose that

(a) there is some value $\delta > 0$ such that, for all $t < T$, it holds that
$$X_t - \mathrm{E}\big[X_{t+1} \,\big|\, \mathcal{F}_t\big] \geq \delta X_t \text{ , that}$$

(b) $X_0 \geq x_{\min}$ and, for all $t \leq T$, it holds that $X_t \geq 0$.

Then
$$\mathrm{E}[T \mid \mathcal{F}_0] \leq \frac{1 + \ln\Big(\frac{X_0}{x_{\min}}\Big)}{\delta} \text{ .}$$ ◀

Proof. We define a function $h\colon [x_{\min}, \infty) \to \mathbf{R}^+$ with $h(x) = \delta x$. Note that h is monotonically increasing and that, by construction, for all $t < T$, we have $X_t - \mathrm{E}[X_{t+1} \mid \mathcal{F}_t] \geq h(X_t)$. Thus, by applying Theorem 3.13, we get

$$\begin{aligned}\mathrm{E}[T \mid \mathcal{F}_0] &\leq \frac{x_{\min}}{h(x_{\min})} + \int_{x_{\min}}^{X_0} \frac{1}{h(z)} \mathrm{d}z \\ &= \frac{x_{\min}}{\delta x_{\min}} + \frac{\ln\Big(\frac{X_0}{x_{\min}}\Big)}{\delta} \text{ ,}\end{aligned}$$

which completes the proof. ■

3.5.2 Hitting the Target

By applying Theorem 3.14 instead of Theorem 3.13, we get the following theorem. As in the case of Theorem 3.14, the process now has to be lower-bounded by $x_{\min}$.

▶ **Corollary 3.16 (Upper Multiplicative Drift, Unbounded, Hitting Target).** Let $(X_t)_{t \in \mathbf{N}}$ be random variables over $\mathbf{R}$ adapted to a filtration $(\mathcal{F}_t)_{t \in \mathbf{N}}$, let $x_{\min} > 0$, and let $T = \inf\{t \in \mathbf{N} \mid X_t \leq x_{\min}\}$. Furthermore, suppose that

(a) there is some value $\delta > 0$ such that, for all $t < T$, it holds that

$$X_t - \mathrm{E}\big[X_{t+1} \,\big|\, \mathcal{F}_t\big] \geq \delta X_t \text{ , and that,}$$

(b) for all $t \leq T$, it holds that $X_t \geq x_{\min}$.

Then

$$\mathrm{E}[T \mid \mathcal{F}_0] \leq \frac{\ln\left(\frac{X_0}{x_{\min}}\right)}{\delta} \,. \qquad ◀$$

Proof. We define the same potential as in the proof of Corollary 3.15 but apply Theorem 3.14 instead. ■

As before, we provide an example that shows that the upper bounds are as tight as possible, up to constant factors, for the range of processes we consider.

▶ **Example 3.17.** Let $\delta \in (0, 1)$ be a value bounded away from 1. Consider the process $(X_t)_{t \in \mathbf{N}}$, with $X_0 > 1$, that decreases each step deterministically such that $X_{t+1} = (1 - \delta)X_t$ holds. Let T denote the first point in time such that the process drops below 1. Thus, we get $T \in \Theta\big(-\log_{(1-\delta)}(X_0)\big) = \Theta\big(-\ln(X_0)/\ln(1-\delta)\big) = \Theta\big(\ln(X_0)/\delta\big)$, where the last equation makes use of the Taylor expansion of $\ln(1 - \delta) \in \Theta(-\delta)$, as $1 - \delta$ does not converge to 0, by assumption. ◀

3.6 Drift Without Drift

In order for any of the above drift theorems to be applicable, the process needs to have a positive drift toward the target. However, sometimes one is interested in the first-hitting time of unbiased processes, that is, processes with a drift 0. The classical example for that is the Gambler's Ruin process [MU05, Section 7.2.1], which describes a fair random walk.

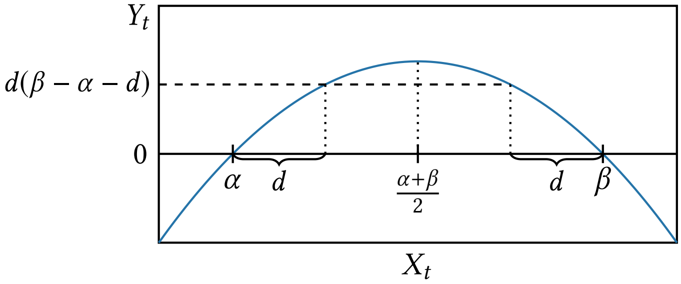


Figure 3.1: A depiction of how Theorem 3.18 transforms a random variable X_t. The resulting random variable Y_t is positive over (α, β), has the roots α and β, and takes its maximum $(\beta - \alpha)^2/4$ at $(\alpha + \beta)/2$. Note that in the interval $[\alpha, \beta]$, the random variable X_t is within a distance of $d \in [0, (\beta - \alpha)/2]$ to α or β if and only if $Y_t \leq d(\beta - \alpha - d)$.

In this section, we focus on such unbiased processes, that is, martingales. We show that in these cases the variance (which is nonnegative by definition; Definition 2.6) can be used in order to apply a drift theorem. Since the variance of a process is 0 if and only if the process is deterministic, we get a framework applicable to any unbiased random process.

We start by providing a transformation of a martingale into another random process that has positive drift. The underlying method of this transformation is known as *predictable quadratic variation,* although it is not always referred to under this name. For more information, please refer to the books of Durrett [Dur19, Chapter 4.5] or Williams [Wil91, Chapter 12.11]. The way the martingale is transformed is depicted in Figure 3.1.

▶ **Theorem 3.18 (Martingale Drift Transformation).** Let $(X_t)_{t \in \mathbf{N}}$ be random variables over $\mathbf{R}$ adapted to a filtration $(\mathcal{F}_t)_{t \in \mathbf{N}}$, let $[\alpha, \beta] \subset \mathbf{R}$ be an interval, and let $T = \inf\{t \in \mathbf{N} \mid X_t \notin (\alpha, \beta)\}$. Furthermore, suppose that,

(a) for all $t < T$, it holds that $\mathrm{Var}[X_{t+1} \mid \mathcal{F}_t] > 0$ and that,

(b) for all $t < T$, it holds that $\mathrm{E}[X_{t+1} \mid \mathcal{F}_t] = X_t$.

Then the process $(Y_t)_{t \in \mathbf{N}}$ with

$$Y_t = (X_t - \alpha)(\beta - X_t)$$

is, for all $t < T$, a positive random process with drift $\mathrm{Var}[X_{t+1} \mid \mathcal{F}_t]$ toward 0. ◀

Proof. Since, for all $t \in \mathbf{N}$, the random variable Y_t is a concave quadratic function with the roots α and β, it holds, for all $t' < T$, that $Y_{t'} > 0$, that is, the process is positive then. We now determine the drift of Y_t (toward 0) for all $t < T$. Note that Y_t is $\mathcal{F}_t$-measurable, since it is expressed by constants and by X_t, which is $\mathcal{F}_t$-measurable by assumption.

$$\begin{aligned} Y_t - \mathrm{E}[Y_{t+1} \mid \mathcal{F}_t] &= (X_t - \alpha)(\beta - X_t) + \alpha - \mathrm{E}[(X_{t+1} - \alpha)(\beta - X_{t+1}) + \alpha \mid \mathcal{F}_t] \\ &= -X_t^2 + (\alpha + \beta)X_t - \alpha\beta - \mathrm{E}\big[-X_{t+1}^2 + (\alpha + \beta)X_{t+1} - \alpha\beta \,\big|\, \mathcal{F}_t\big] \\ &= -\overbrace{X_t}^{= \mathrm{E}[X_{t+1} \mid \mathcal{F}_t]}{}^2 + \mathrm{E}\big[X_{t+1}^2 \,\big|\, \mathcal{F}_t\big] + (\alpha + \beta)X_t - (\alpha + \beta)\overbrace{\mathrm{E}[X_{t+1} \mid \mathcal{F}_t]}^{=X_t} \\ &= \mathrm{E}\big[X_{t+1}^2 \,\big|\, \mathcal{F}_t\big] - \mathrm{E}[X_{t+1} \mid \mathcal{F}_t]^2 \\ &= \mathrm{Var}[X_{t+1} \mid \mathcal{F}_t]\,, \end{aligned}$$

which is positive by assumption. This concludes the proof. ■

Since the transformed process Y described in Theorem 3.18 is positive as long as $X_t \in (\alpha, \beta)$, it follows that T also denotes the first-hitting time of $Y_t \leq 0$. Hence, Y_t can be used in order to apply any drift theorem.

► **Corollary 3.19 (Martingale Upper Additive Drift).** Let $(X_t)_{t \in \mathbf{N}}$ be random variables over $[\alpha, \beta] \subset \mathbf{R}$ adapted to a filtration $(\mathcal{F}_t)_{t \in \mathbf{N}}$, and let $T = \inf\big\{t \in \mathbf{N} \,\big|\, X_t \in \{\alpha, \beta\}\big\}$. Furthermore, suppose that

(a) there is some value $\delta > 0$ such that, for all $t < T$, it holds that

$$\mathrm{Var}[X_{t+1} \mid \mathcal{F}_t] \geq \delta \text{ and that,}$$

(b) for all $t < T$, it holds that $\mathrm{E}[X_{t+1} \mid \mathcal{F}_t] = X_t$.

Then

$$\mathrm{E}[T \mid \mathcal{F}_0] \leq \frac{(X_0 - \alpha)(\beta - X_0)}{\delta}\,. \qquad ◄$$

Proof. We use Theorem 3.18 to transform X into the process Y, which has a drift of at least δ, by assumption. Note that, for all $t \leq T$, it holds that $Y_t \geq 0$. Applying Corollary 3.8 completes the proof. ■

For the lower bound drift theorem, the martingale itself does not have to be bounded but only the search space. Due to the boundedness of the search space, we do not require a restriction on the expected step size.

▶ **Corollary 3.20 (Martingale Lower Additive Drift).** Let $(X_t)_{t\in\mathbf{N}}$ be random variables over $\mathbf{R}$ adapted to a filtration $(\mathcal{F}_t)_{t\in\mathbf{N}}$, let $[\alpha,\beta] \subset \mathbf{R}$ be an interval, and let $T = \inf\{t \in \mathbf{N} \mid X_t \notin (\alpha,\beta)\}$. Furthermore, suppose that,

(a) there is some value $\delta > 0$ such that, for all $t < T$, it holds that

$$\mathrm{Var}[X_{t+1} \mid \mathcal{F}_t] \leq \delta \text{ and that,}$$

(b) for all $t < T$, it holds that $\mathrm{E}[X_{t+1} \mid \mathcal{F}_t] = X_t$.

Then

$$\mathrm{E}[T \mid \mathcal{F}_0] \geq \frac{(X_0-\alpha)(\beta-X_0)}{\delta} \,. \qquad ◀$$

Proof. We use Theorem 3.18 and want to apply Theorem 3.11. For this, we can argue analogously as in the proof of Corollary 3.19. However, we still need to check the expected bounded step size of Y.

Note that Y_t is a convex function in X_t and is maximal for $X_t = (\alpha+\beta)/2$, resulting in $|Y_t| \leq (\alpha\beta/4)^2$. Thus, in order to bound

$$\mathrm{E}\big[|Y_{t+1} - Y_t| \,\big|\, \mathcal{F}_t\big] \leq \mathrm{E}\big[|Y_{t+1}| \,\big|\, \mathcal{F}_t\big] + |Y_t|$$

we are left with bounding

$$\begin{aligned}\mathrm{E}\big[|Y_{t+1}| \,\big|\, \mathcal{F}_t\big] &= \mathrm{E}\big[|(X_{t+1}-\alpha)(\beta-X_{t+1})| \,\big|\, \mathcal{F}_t\big] \\ &\leq |\alpha+\beta|\mathrm{E}\big[|X_{t+1}| \,\big|\, \mathcal{F}_t\big] + \mathrm{E}\big[X_{t+1}^2 \,\big|\, \mathcal{F}_t\big] + |\alpha\beta| \,.\end{aligned}$$

Since we can bound $\mathrm{Var}[X_{t+1} \mid \mathcal{F}_t] = \mathrm{E}\big[X_{t+1}^2 \,\big|\, \mathcal{F}_t\big] - \mathrm{E}[X_{t+1} \mid \mathcal{F}_t]^2$ by assumption, we can bound the two expected values $\mathrm{E}\big[X_{t+1}^2 \,\big|\, \mathcal{F}_t\big]$ and $\mathrm{E}[|X_{t+1}| \mid \mathcal{F}_t]$ and therefore $\mathrm{E}\big[|Y_{t+1} - Y_t| \,\big|\, \mathcal{F}_t\big]$. Applying Theorem 3.11 finishes the proof. ■

The other drift theorems follow analogously, using Theorem 3.18. However, the resulting theorems get more complicated, since the process Y is more complex than X. One consequence of this is that the drift of Y has to be stated with respect to $(X_t-\alpha)(\beta-X_t)$ instead of X_t, which may inhibit the applicability of such a theorem. Nonetheless, we would like to mention that we state a multiplicative version of a drift theorem for martingales in Corollary 6.6, and we successfully apply it in Section 6.3.

As we mentioned at the beginning of this section, Theorem 3.18 allows to generalize from processes like the Gambler's Ruin. In the following, we want to use the above theorems in order to get the exact first-hitting time of said process.

▶ **Example 3.21 (Gambler's Ruin [MU05, Section 7.2.1]).** Let $n \in \mathbf{N}$, and let $(X_t)_{t \in \mathbf{N}}$ be a random process over $\{0, \ldots, 2n\}$ such that, for all $t \in \mathbf{N}$, it holds that

(a) if $X_t = x \notin \{0, 2n\}$, then $\Pr[X_{t+1} = x - 1] = \Pr[X_{t+1} = x + 1] = \frac{1}{2}$, and

(b) if $X_t = x \in \{0, 2n\}$, then $\Pr[X_{t+1} = x] = 1$.

Further, let $T = \inf \left\{t \in \mathbf{N} \,\middle|\, X_t \in \{0, 2n\}\right\}$.

Note that, for all $t \in \mathbf{N}$, it holds that $\mathrm{E}[X_{t+1} \mid X_t] = X_t$. Hence, we use Theorem 3.18 and bound, for all $t < T$,

$$\mathrm{Var}[X_{t+1} \mid X_t] = \frac{1}{2} \cdot (X_t - 1 - X_t)^2 + \frac{1}{2} \cdot (X_t + 1 - X_t)^2 = 1\,.$$

Applying both Corollary 3.19 and 3.20 yields $\mathrm{E}[T \mid X_0] = X_0(2n - X_0)$. Especially, for $X_0 = n$, we get $\mathrm{E}[T] = n^2$. ◀

3.7 Negative Drift

In order to provide a more complete picture of drift, we also want to mention two common negative-drift theorems and derive a useful corollary from one of them, which we use in our results.

Negative drift considers a setting where the drift is directed *away* from the target, that is, the drift toward the target is *negative.* Negative-drift theorems are then used to derive superpolynomial lower bounds on the probability that first-hitting time of the process reaching the target is polynomial.

The first theorem considers random processes that are allowed to make arbitrarily large jumps as long as the probability of such jumps decreases exponentially in their width. It is also important to note that the drift has to be bounded by a constant.

▶ **Theorem 3.22 (Negative Drift for an Interval [OW11; OW12]).** Let $(X_t)_{t \in \mathbf{N}}$ be random variables over $\mathbf{R}$ adapted to a filtration $(\mathcal{F}_t)_{t \in \mathbf{N}}$, and let $b \in \mathbf{R}$ such that $X_0 \geq b$. Suppose that there exist an interval $[a, b] \subseteq \mathbf{R}$, two constants

δ, $\varepsilon > 0$, and, possibly depending on $\ell := b - a$, a function $r(\ell)$ satisfying $1 \leq r(\ell) \in \mathrm{o}\big(\ell/\log(\ell)\big)$ such that, for all $t \in \mathbf{N}$, it holds that

(a) $\mathrm{E}\big[(X_{t+1} - X_t) \cdot \mathbf{1}\{a < X_t < b\} \bigm| \mathcal{F}_t\big] \geq \varepsilon \cdot \mathbf{1}\{a < X_t < b\}$ and,

(b) for all $j \in \mathbf{N}$, it holds that

$$\Pr\big[|X_{t+1} - X_t| \cdot \mathbf{1}\{X_t > a\} \geq j \cdot \mathbf{1}\{X_t > a\} \bigm| \mathcal{F}_t\big] \leq \frac{r(\ell)}{(1+\delta)^j} .$$

Then there is a constant $c > 0$ and a function $m(\ell) \in \Omega\big(\ell/r(\ell)\big)$ such that, for $T := \inf\{t \in \mathbf{N} \mid X_t \leq a\}$, it holds that

$$\Pr\Big[T \leq 2^{\frac{c\ell}{r(\ell)}}\Big] = 2^{-m(\ell)} . \qquad ◀$$

Note that the indicator random variables in Theorem 3.22 say that the respective condition only has to hold if the proposition of the indicator function is true. Otherwise, the condition is trivially satisfied.

Theorem 3.22 is very useful, since it only assumes a negative drift over a part of the search space. However, when considering a process with bounded step size, condition (b) may be harder to check. In such a case, it may make sense to use the following theorem that works for processes that are deterministically bounded in their step size. Note that both the drift and the bound on the step size do not have to be constants in this theorem.

▶ **Theorem 3.23 (Negative Drift; Bounded Step Size [Köt16]).** Let $(X_t)_{t\in\mathbf{N}}$ be random variables over $\mathbf{R}$ adapted to a filtration $(\mathcal{F}_t)_{t\in\mathbf{N}}$. Further, let $X_0 \leq 0$, let $b > 0$, and let $T = \inf\{t \in \mathbf{N} \mid X_t \geq b\}$. Suppose that there are values $c \in (0, b)$ and $\varepsilon < 0$ such that, for all $t < T$, it holds that

(a) $\mathrm{E}[X_{t+1} - X_t \mid \mathcal{F}_t] \leq \varepsilon$ and that

(b) $|X_t - X_{t+1}| < c$.

Then, for all $t \in \mathbf{N}$, it holds that

$$\Pr[T \leq t] \leq t \cdot \mathrm{e}^{-\frac{b|\varepsilon|}{2c^2}} . \qquad ◀$$

Theorem 3.23 comes with the drawback that it assumes the drift to be negative as long as the target has not been reached. This means that the drift has to be negative over the whole search space below the target, not only over a part of it. This is especially bad when considering bounded search spaces, where the drift cannot be negative once the lower bound is reached. Hence, we propose the following corollary that circumvents this problem.

▶ **Corollary 3.24 (Negative Drift for an Interval; Bounded Step Size).** Let $(X_t)_{t \in \mathbf{N}}$ be random variables over $\mathbf{R}$ adapted to a filtration $(\mathcal{F}_t)_{t \in \mathbf{N}}$. Further, let $X_0 \leq 0$, let $b > 0$, and let $T = \inf\{t \in \mathbf{N} \mid X_t \geq b\}$. Suppose that there are values $a \leq 0$, $c \in (0, b)$, and $\varepsilon < 0$ such that, for all $t < T$, it holds that

(a) $\mathrm{E}[(X_{t+1} - X_t) \cdot \mathbf{1}\{X_t \geq a\} \mid \mathcal{F}_t] \leq \varepsilon \cdot \mathbf{1}\{X_t \geq a\}$, that

(b) $|X_t - X_{t+1}| \cdot \mathbf{1}\{X_t \geq a\} < c \cdot \mathbf{1}\{X_t \geq a\} + \mathbf{1}\{X_t < a\}$, and that

(c) $X_{t+1} \cdot \mathbf{1}\{X_t < a\} \leq 0$.

Then, for all $t \in \mathbf{N}$, it holds that

$$\Pr[T \leq t] \leq t^2 \cdot \mathrm{e}^{-\frac{b|\varepsilon|}{2c^2}} .$$ ◀

Proof. We consider two different phases of a run of X: one phase where Theorem 3.23 is applicable and one phase where it is not because we left the interval $[a, n)$ where negative drift is guaranteed. In the former case, we can bound the probability, in the latter case, we need to invest at least one step in order to be able to reach n, which can be viewed as a restart. We then recurse until no more steps are left.

Let A denote the event that X_0 can reach n without getting below a. For all $t \in \mathbf{N}$, we can then write

$$\Pr[T \leq t] = \Pr[\{T \leq t\} \cap A] + \Pr\left[\{T \leq t\} \cap \overline{A}\right] ,$$

which is basically Theorem 2.1 but also allows for $A = \emptyset$.

Note that we can use Theorem 3.23 to bound $\Pr[\{T \leq t\} \cap A]$, since the indicator random variables in conditions (a) and (b) are 1 in this case.

In order to bound $\Pr\left[\{T \leq t\} \cap \overline{A}\right]$, we make the following observation: X must go below a at one point in time (due to $\overline{A}$) and then can only reach n by getting back into the interval $[a, 0]$, due to condition (c). This costs at least one step. Hence, for all $t \in \mathbf{N}$, we get

$$\Pr\left[\{T \leq t\} \cap \overline{A}\right] \leq \Pr[T \leq t - 1] \,.$$

We now use the same arguments as before in order to bound $\Pr[T \leq t - 1]$. Thus, for all $t \in \mathbf{N}$, we get

$$\begin{aligned}\Pr[T \leq t] &\leq \sum_{i=1}^{t} i \cdot \mathrm{e}^{-\frac{b|\varepsilon|}{2c^2}}\\ &\leq t^2 \cdot \mathrm{e}^{-\frac{b|\varepsilon|}{2c^2}} \,,\end{aligned}$$

which concludes the proof. ■

Corollary 3.24 is very pessimistic in its bound but, in return, does not make many more assumptions on the process than Theorem 3.23. And since the goal of using a negative-drift theorem usually is to get a superpolynomial upper bound on the first-hitting time being polynomial, the extra factor of t in Corollary 3.24 has no impact then.

4 The n-Bernoulli-λ-EDA Framework

This chapter is based on joint work with Tobias Friedrich and Timo Kötzing [FKK16], where the framework was first introduced. Section 4.2 contains additional definitions, which are based on another joint work with Tobias Friedrich and Timo Kötzing [FKK18].

In this chapter, we introduce a framework for a class of estimation-of-distribution algorithms (EDAs) that all of our results fall into, with the exception of Algorithm 2 from Chapter 10. We call this framework the *n-Bernoulli-λ-EDA* (see Algorithm 1).

We start by discussing different frameworks that contain some EDAs. Afterward, we introduce our framework and show that all commonly theoretically analyzed EDAs are subsumed by it.

4.1 Introduction

A framework is an algorithm that is flexible enough to yield other algorithms as special cases when using certain parameter values. Hence, we use the term *framework* synonymously with the class of algorithms that it subsumes.

The main reason for considering a framework is to prove propositions for an entire class of algorithms at once instead of showing them for every single algorithm. Further, a framework provides the same terminology for all algorithms of the class. Hence, even when properties are only proven for certain algorithms, one can ask oneself whether this statement generalizes to the entire framework.

The framework we introduce in this chapter – the n-Bernoulli-λ-EDA (Algorithm 1) – is motivated by the two aforementioned premises: first, we use it in order to prove general properties of a special class of EDAs in Chapters 5 and 6. Second, we prove run time results for certain algorithms in Chapters 7 to 10 and discuss how these relate to known run time results for similar EDAs.

As we discussed in Chapter 1, we consider univariate EDAs optimizing pseudo-Boolean functions. Thus, our framework encompasses only such algorithms.

That is, their probabilistic model is fully determined by a probability vector where each component denotes the probability to sample a 1 at that position; and the only parameters of the framework are the number of samples drawn each iteration and a function that says how the probabilistic model is updated (equation (4.1)). In Section 4.3, we show how commonly analyzed EDAs are subsumed by this framework.

We use the n-Bernoulli-λ-EDA as our framework, as it best captures the algorithms we are interested in. However, other frameworks exist that also subsume univariate EDAs, and we would like to discuss them briefly.

Ollivier et al. [Oll+17] propose the method of Information-Geometric Optimization (IGO), which is a very general method defined for arbitrary search spaces and uses an update method known as *infinitesimal maximum likelihood.* Applying the IGO to the Boolean domain results in an algorithm that also subsumes all of the algorithm we mention in Section 4.3. However, it does not encapsulate univariate EDAs that do not follow such an update rule. Further, due to its applicability to any domain (and it being mainly motivated by continuous search spaces), the IGO framework is not very well suited for our discrete setting, that is, it is too vast.

Another framework was introduced by Paixão et al. [Pai+15], which is intended to model any evolutionary process. Hence, this very general framework subsumes all univariate EDAs. However, similar to the IGO, due to its generality, it does not specifically focus on EDAs and is not well-suited for their analysis.

Other approaches have been examined by Shapiro [Sha06], who analyzed EDAs that use a maximum-likelihood update, and by Dang and Lehre [DL15], who proposed a class of EDAs that update their distributions only with information from the current samples, not with information from the current distribution. Again, these approaches do not capture all univariate EDAs.

4.2 The n-Bernoulli-λ-EDA

The n-Bernoulli-λ-EDA generalizes EDAs that use an n-fold product of a Bernoulli distribution as their probabilistic model. This means that their model is represented by a probability vector $\boldsymbol{p} \in [0, 1]^n$, and an individual $\boldsymbol{x} \in \{0, 1\}^n$ is sampled with respect to $\boldsymbol{p}$ such that

$$\forall i \in [n]\colon \Pr[\boldsymbol{x}_i = 1] = \boldsymbol{p}_i \wedge \Pr[\boldsymbol{x}_i = 0] = 1 - \boldsymbol{p}_i\ .$$

Algorithm 1: The n-Bernoulli-λ-EDA framework with a given update scheme φ, optimizing f.

```
t ← 0;
$\boldsymbol{p}^{(t)} \leftarrow \frac{1}{2}$;
repeat
    $D \leftarrow \emptyset$;
    for $j \in [\lambda]$ do
        $\boldsymbol{x} \leftarrow$ offspring sampled with respect to $\boldsymbol{p}^{(t)}$;
        $D \leftarrow D \cup \{\boldsymbol{x}\}$;
    $\boldsymbol{p}^{(t+1)} \leftarrow \varphi\big(\boldsymbol{p}^{(t)}, (\boldsymbol{x}, f(\boldsymbol{x}))_{\boldsymbol{x} \in D}\big)$;
    $t \leftarrow t + 1$;
until termination criterion met;
```

We call $\boldsymbol{p}$ the *frequency vector* of the algorithm and its components *frequencies.* The initial frequency vector is given by $\boldsymbol{p}^{(0)} = \mathbf{1/2}$.

Until a termination criterion is met, the algorithm iteratively samples $\lambda \in \mathbf{N}^+$ offspring and updates each frequency according to the *update scheme* of the algorithm, which is a function

$$\varphi\colon [0,1]^n \times \big(\{0,1\}^n \times \mathbf{R}\big)^\lambda \to [0,1]^n \tag{4.1}$$

that takes the current frequency vector $\boldsymbol{p}^{(t)}$, an offspring population[10] D of λ individuals sampled according to $\boldsymbol{p}^{(t)}$, and their respective fitness, and it yields the frequencies of $\boldsymbol{p}^{(t+1)}$ for the following iteration. The whole framework is summarized in Algorithm 1.

Given an offspring population D, we say that $\boldsymbol{x} \in D$ has *rank* $i \in [\lambda]$ (denoted as $\boldsymbol{x}^{(i)}$) if $\boldsymbol{x}$ has the i-th best fitness in D.[11] Since we want ranks to be unique, we assume that ties are broken uniformly at random if not stated otherwise.

Given an n-Bernoulli-λ-EDA A, let $P^{(t)}(A)$ denote the set of all possible frequency vectors of A in iteration t, and let $P(A) := \bigcup_{t=0}^{\infty} P^{(t)}(A)$ denote the set of all possible frequency vectors of A. Analogously, $P_i^{(t)}(A) \subseteq [0,1]$ denotes the set

[10] We assume that the population is a multiset, that is, we allow duplicate entries.

[11] Note that *best* is relative to whether the objective is maximization or minimization. For maximization, for example, the best fitness would be the largest value.

of possible values of the frequency $\boldsymbol{p}_i^{(t)}$, and $P_i(A)$ denotes the set of all possible values of $\boldsymbol{p}_i$. We say that A is *ρ-bounded* if, for all $i \in [n]$ and all $p \in P(A)$, we have $|\boldsymbol{p}_i - \varphi(\boldsymbol{p}, \cdot)_i| \leq \rho$, that is, an update can change a frequency by at most ρ.

4.2.1 Special Update Schemes

Equation (4.1) allows arbitrary interactions between different positions and individuals or even fitness values. Normally, EDAs that fall into the n-Bernoulli-λ-EDA framework are not that permissive when performing an update. Thus, in the following, we describe more restricted update schemes.

Decomposable EDAs

Commonly, n-Bernoulli-λ-EDAs consider each frequency independently when performing an update. The following definition formalizes this behavior.

▶ **Definition 4.1 (Decomposable EDAs).** Let A be an n-Bernoulli-λ-EDA with update scheme φ. We say that A is *decomposable* if and only if there exists a set of n functions $\dot{\varphi}_i \colon [0, 1] \times \left(\{0, 1\} \times \mathbf{R}\right)^{\lambda} \to \mathbf{R}$, indexed by $i \in [n]$ such that, for all fitness functions f, all possible frequency vectors $\boldsymbol{p} \in P(A)$, all offspring populations D that can be sampled by $\boldsymbol{p}$, and all $i \in [n]$, it holds that

$$\varphi\big(\boldsymbol{p}, \big(\boldsymbol{x}, f(\boldsymbol{x})\big)_{\boldsymbol{x} \in D}\big)_i = \dot{\varphi}_i\big(\boldsymbol{p}_i, \big(\boldsymbol{x}_i, f(\boldsymbol{x})\big)_{\boldsymbol{x} \in D}\big) \ .$$

If there is a single function $\dot{\varphi}$ such that, for all $i \in [n]$, $\dot{\varphi} = \dot{\varphi}_i$, we say that A is *singularly decomposable*. We then drop the index i. ◀

As we will see in Section 4.3, all EDAs we present are singularly decomposable. Hence, they treat every position equally and do not introduce a bias toward certain positions. In Section 5.4, we go more into detail about this property.

Locally Updating EDAs

Some singularly decomposable n-Bernoulli-λ-EDAs restrict their update scheme by performing an update to a frequency that is based solely on the current frequency value and the information of whether it should be increased, decreased, or not changed at all. We call such EDAs *locally updating*, since they only have a very local view on all of the potential information when performing an update.

▶ **Definition 4.2 (Locally Updating EDAs).** Let A be an n-Bernoulli-λ-EDA with update scheme φ. We say that A is *locally updating* if and only if there are two functions

- $\mathrm{move}\colon \big(\{0,1\} \times \mathbf{R}\big)^{\lambda} \to \{\mathrm{up}, \mathrm{stay}, \mathrm{down}\}$ and
- $\mathrm{set}\colon [0,1] \to [0,1]$, which is monotonically increasing,

such that, for all fitness functions f, all possible frequency vectors $\boldsymbol{p} \in P(A)$, all offspring populations D that can be sampled by $\boldsymbol{p}$, and all $i \in [n]$, abbreviating $v_i \coloneqq \mathrm{move}\big((\boldsymbol{x}_i, f(\boldsymbol{x}))_{\boldsymbol{x} \in D}\big)$, it holds that

$$\varphi\big(\boldsymbol{p}, (\boldsymbol{x}, f(\boldsymbol{x}))_{\boldsymbol{x} \in D}\big)_i = \begin{cases} \mathrm{set}(\boldsymbol{p}_i) & \text{if } v_i = \mathrm{up}, \\ \boldsymbol{p}_i & \text{if } v_i = \mathrm{stay}, \\ 1 - \mathrm{set}(1 - \boldsymbol{p}_i) & \text{if } v_i = \mathrm{down}. \end{cases}$$ ◀

Note that the definition enforces that an increase and a decrease are centrally symmetric around $(1/2, 1/2)$, that is, an increase of $\boldsymbol{p}_i$ is the same as a decrease of $1 - \boldsymbol{p}_i$. We go more into detail about why this is beneficial in Section 5.5.

We assume that $\mathrm{set}(\boldsymbol{p}_i) \geq \boldsymbol{p}_i$ in order to enforce that ›up‹ does not decrease the frequency. Note that this also entails that ›down‹ does not increase the frequency. Without this assumption, ›down‹, for example, could take the role of ›up‹, which would result in convoluted case distinctions in proofs. By assuming $\mathrm{set}(\boldsymbol{p}_i) \geq \boldsymbol{p}_i$, we avoid such overlaps.

4.2.2 Margins

When a frequency $\boldsymbol{p}_i^{(t)}$ is 0 or 1, then all bits sampled at position i in iteration t are either 0 or 1, respectively. Usually, the update scheme of an n-Bernoulli-λ-EDA is phrased such that the updated frequency $\boldsymbol{p}_i^{(t+1)}$ is 0 or 1 again.[12] This means that the algorithm is stuck at this position and cannot change its frequency anymore. In order to prevent this and to always be able to sample 0s and 1s at each position, many algorithms cap their frequency vector beforehand. We then speak of a *margin.*

Formally, a margin is a number $b \in (0, 1/2]$ such that, for all $\boldsymbol{p} \in P(A)$, it holds that $\boldsymbol{p} \in [b, 1-b]^n$, and we call b and $1 - b$ the *lower* and *upper border,*

[12] Note that the update scheme could, in fact, check whether a frequency is 0 or 1 and then set it to another value. However, we do not know of any EDA that does this.

respectively. A common way to ensure that an n-Bernoulli-λ-EDA has a margin of b is to change line 8 of Algorithm 1 as follows:

$$\boldsymbol{p}_i^{(t+1)} = \min\Big\{\max\Big\{\varphi\big(\boldsymbol{p}^{(t)}, \big(\boldsymbol{x}, f(\boldsymbol{x})\big)_{\boldsymbol{x}\in D}\big)_i, b\Big\}, 1-b\Big\} . \tag{4.2}$$

This way, borders can easily be enforced on an existing update scheme without changing the scheme itself.

We would like to note that a common margin for n-Bernoulli-λ-EDA is $1/n$, which is also the typical mutation rate for classical EAs [Doe+13; Jan13].

4.3 Classifying Existing EDAs

In this section, we show how existing EDAs are subsumed by the n-Bernoulli-λ-EDA framework. We only focus on those algorithms that have been theoretically analyzed, and we mention corresponding theoretical results. If not stated otherwise, the results always assume a margin of $1/n$.

4.3.1 PBIL

The Population-based Incremental Learning algorithm (PBIL) was introduced by Baluja [Bal94]. It is a ρ-bounded, singularly decomposable n-Bernoulli-λ-EDA with parameters ρ (the *learning rate*) and μ (the *population size*) with $\mu \leq \lambda$. The update scheme is, for all $i \in [n]$,

$$\varphi\big(\boldsymbol{p}, \big(\boldsymbol{x}, f(\boldsymbol{x})\big)_{\boldsymbol{x}\in D}\big)_i = (1-\rho)\boldsymbol{p}_i + \rho\frac{\sum_{k=1}^{\mu}\boldsymbol{x}_i^{(k)}}{\mu} .$$

This algorithm yields other well-known algorithms for the extreme cases of $\mu = 1$ or $\rho = 1$, as we will show next.

Höhfeld and Rudolph [HR97] were the first to prove convergence results for the PBIL without a margin. Very recently, Lehre and Nguyen [LN18] proved an expected run time of $\mathrm{O}\big(n\lambda\log(\lambda) + n^2\big)$, for $\lambda \in \Omega\big(\log(n)\big)$, on LeadingOnes and the binary-value function BinVal.

4.3.2 UMDA

The Univariate Marginal Distribution Algorithm (UMDA) was introduced by Mühlenbein and Paaß [MP96]. It is a special case of the PBIL for $\rho = 1$. Hence, it is still a singularly decomposable n-Bernoulli-λ-EDA. However, it is not longer bounded. The update scheme is, for all $i \in [n]$,

$$\varphi\big(\boldsymbol{p}, (\boldsymbol{x}, f(\boldsymbol{x}))_{\boldsymbol{x} \in D}\big)_i = \frac{\sum_{k=1}^{\mu} \boldsymbol{x}_i^{(k)}}{\mu} .$$

This means that a frequency is set to the relative number of 1s of the μ best individuals. This way, a frequency can, in theory, make large changes in a single iteration, for example, from $1/\mu$ to 1. However, such large changes are very unlikely. We go into detail about the UMDA and its update scheme in Chapter 9.

The UMDA has been extensively analyzed in the past years. Initially, Chen et al. [Che+09a; Che+09b] considered it on non-standard functions they defined in the papers. Then, Chen et al. [Che+10] proved a polynomial run time of the UMDA on the benchmark function LeadingOnes. This result was followed up by Dang and Lehre [DL15], who improved the upper bound on LeadingOnes and proved an upper bound on OneMax. Krejca and Witt [KW18a] proved a lower bound of $\Omega\big(n\log(n)\big)$ on OneMax, for certain parameter settings.[13] The matching upper bound was proven independently by Witt [Wit17] and Lehre and Nguyen [LN17].

4.3.3 λ-MMAS$_{\text{IB}}$

The MAX-MIN Ant System algorithm with iteration-best update (λ-MMAS$_{\text{IB}}$) was introduced by Neumann et al. [NSW10]. The algorithm builds upon the more general MMAS framework by Stützle and Hoos [SH00], which considers virtual ants traversing edges of a graph, laying pheromone, in order to represent a probabilistic model on graphs. An ant chooses an edge proportional to its pheromone value with respect to the sum of all pheromone values of outgoing edges of that node. When considering a directed path of length n with two edges between two adjacent nodes, the resulting pheromone vector can be interpreted as a frequency vector (see Figure 4.1). Hence, such special cases are univariate EDAs.

[13] Chapter 9 is based on this result.

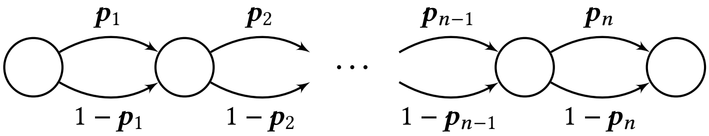


Figure 4.1: A graph whose pheromone trail can be interpreted as a frequency vector. A virtual ant is supposed to traverse the graph from left to right. Each iteration, it has to choose one of two edges per node. If it chooses the top edge, this can be interpreted as sampling a 1 in a univariate EDA model. If the ant traverses the bottom edge, this can be interpreted as sampling a 0.

The λ-MMAS$_{\text{IB}}$ is a special case of the PBIL for $\mu = 1$, that is, only the best individual of an iteration is considered for an update. The λ-MMAS$_{\text{IB}}$ is a ρ-bounded, locally updating n-Bernoulli-λ-EDA with the following two functions:

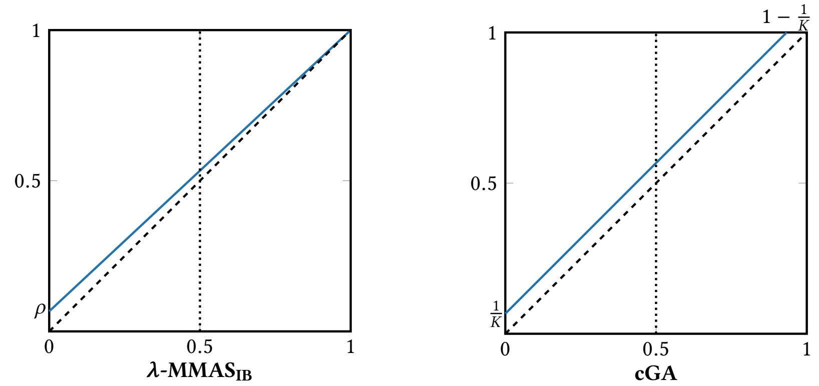
$$\operatorname{move}\Big(\big(\boldsymbol{x}_i, f(\boldsymbol{x})\big)_{\boldsymbol{x}\in D}\Big) = \begin{cases} \text{up} & \text{if } \boldsymbol{x}_i^{(1)} = 1\,, \\ \text{down} & \text{if } \boldsymbol{x}_i^{(1)} = 0\,, \text{ and} \end{cases}$$
$$\operatorname{set}(\boldsymbol{p}_i) = (1-\rho)\,\boldsymbol{p}_i + \rho\ .$$

The ›set‹ function is depicted in Figure 4.2 (a). Note that if a frequency $\boldsymbol{p}_i$ is decreased, it takes the value $(1-\rho)\boldsymbol{p}_i$. Hence, for all $i \in [n]$, the update scheme of the λ-MMAS$_{\text{IB}}$ can also be written as

$$\varphi\big(\boldsymbol{p}, \big(\boldsymbol{x}, f(\boldsymbol{x})\big)_{\boldsymbol{x}\in D}\big)_i = (1-\rho)\boldsymbol{p}_i + \boldsymbol{x}_i^{(1)}\rho\ ,$$

that is, the current frequency $\boldsymbol{p}_i$ is always reduced by a factor of $1-\rho$ and only increased if the bit at position i of the best individual is 1.

Interestingly, λ-MMAS$_{\text{IB}}$ has mostly been analyzed with $\lambda = 2$. The first results date back to Neumann et al. [NSW10], who proved an upper bound of $\mathrm{O}\big(n\log(n)\big)$ for the 2-MMAS$_{\text{IB}}$ with $\rho \in \Theta\big(1/(\sqrt{n}\log(n))\big)$ on OneMax. Further, they showed that the λ-MMAS$_{\text{IB}}$ takes, with overwhelming probability, expected exponential time on any pseudo-Boolean function with a unique optimum if $\lambda/\rho \leq \big(\ln(n)\big)/244$. Sudholt and Witt [SW16a] proved a matching lower bound of $\Omega\big(n\log(n)\big)$ of 2-MMAS$_{\text{IB}}$ on OneMax.



(a) The ›set‹ function of the λ-MMAS$_{\text{IB}}$. Note that the value 1 cannot be reached within finite time when increasing a frequency. Consequently, the same is true when decreasing a frequency.

(b) The ›set‹ function of the cGA. In contrast to the λ-MMAS$_{\text{IB}}$, the cGA can reach 1 within finite time. In more detail, if a frequency with a value of at least $1 - 1/K$ is increased, it is set to 1.

Figure 4.2: The ›set‹ functions of the λ-MMAS$_{\text{IB}}$ (Figure 4.2 (a)) and of the cGA (Figure 4.2 (b)) depicted as solid (blue) lines. The dashed line represents the identity function, and the distance of the solid line to the identity depicts the change made by a single update. Recall that, for a frequency p, we demand $\text{set}(p) \geq p$. Hence, ›set‹ cannot be below the identity. When mirroring the ›set‹ function at $(1/2, 1/2)$ both horizontally and vertically, one gets the function that is used for decreasing p, that is, $1 - \text{set}(1 - p)$.

4.3.4 cGA

The Compact Genetic Algorithm (cGA) was introduced by Harik. et al. [HLG99]. It is a $1/K$-bounded, locally updating n-Bernoulli-2-EDA with parameter K, the so-called *population size*. Its ›move‹ and ›set‹ functions are as follows:

$$\text{move}\Big(\big(\boldsymbol{x}_i, f(\boldsymbol{x})\big)_{\boldsymbol{x}\in D}\Big) = \begin{cases} \text{up} & \text{if } \boldsymbol{x}_i^{(1)} > \boldsymbol{x}_i^{(2)}\,, \\ \text{down} & \text{if } \boldsymbol{x}_i^{(1)} < \boldsymbol{x}_i^{(2)}\,, \\ \text{stay} & \text{if } \boldsymbol{x}_i^{(1)} = \boldsymbol{x}_i^{(2)}\,, \text{ and} \end{cases}$$

$$\text{set}(\boldsymbol{p}_i) = \boldsymbol{p}_i + \frac{1}{K}\,.$$

The ›set‹ function is depicted in Figure 4.2 (b). Note that if a frequency $\boldsymbol{p}_i$ is decreased, it takes the value $\boldsymbol{p}_i - 1/K$. Hence, for all $i \in [n]$, the update scheme of the cGA can also be written as

$$\varphi\big(\boldsymbol{p}, \big(\boldsymbol{x}, f(\boldsymbol{x})\big)_{\boldsymbol{x} \in D}\big)_i = \boldsymbol{p}_i + (\boldsymbol{x}_i^{(1)} - \boldsymbol{x}_i^{(2)})\frac{1}{K} ,$$

that is, the current frequency $\boldsymbol{p}_i$ is shifted by $1/K$ with respect to the sign of the bit difference at position i of both individuals.

The first theoretical analysis of the cGA was conducted by Droste [Dro06], who analyzed it without a margin on the class of all linear pseudo-Boolean functions. Droste showed that the cGA optimizes different functions in a different expected time, with OneMax being the easiest and BinVal the hardest function. Further, he showed a general lower bound of $\Omega\big(K\sqrt{n}/\log(n)\big)$ for the cGA on any pseudo-Boolean function if $K = \mathrm{poly}(n)$. In Chapter 7, we consider the cGA without a margin in a noisy setting. Sudholt and Witt [SW16a] proved a tight expected run time bound of $\Theta\big(n\log(n)\big)$ on OneMax. Recently, Lengler et al. [LSW18] proved a lower bound of $\Omega\big(K^{1/3}n + n\log(n)\big)$ on OneMax for the cGA if $K = \mathrm{O}\big(\sqrt{n}\log(n)^2\big)$, suggesting a bimodal behavior of the run time, and Witt [Wit18] proved a lower bound of $\Omega\big(n^2\big)$ for the cGA without a margin on LeadingOnes.

5 Unbiasedness of $\boldsymbol{n}$-Bernoulli-$\boldsymbol{\lambda}$-EDAs

This chapter is based on joint work with Tobias Friedrich and Timo Kötzing [FKK18]. Some definitions of this paper have been moved to Chapter 4.

In this chapter, we show when an n-Bernoulli-λ-EDA is unbiased with respect to the encoding of the objective function and when not. We start by introducing the concept of *unbiasedness*. We then define it formally via automorphisms of the discrete hypercube. Afterward, we state our main theorem, Theorem 5.5, which provides a general characterization of an n-Bernoulli-λ-EDA being unbiased. Last, we refine this characterization for decomposable and locally updating n-Bernoulli-λ-EDAs, and we conclude.

The results of this chapter imply that all of the run time results in the following chapters are always applicable for the entire function class, not only for the specific function we consider.

5.1 Introduction

In this thesis, we view EDAs as well as randomized search heuristics (RSHs) in general as problem-agnostic solvers for pseudo-Boolean optimization, and their only way to get any information about the problem at hand is to query the fitness function, seen as a subroutine. Thus, the problem itself can be seen as a black box to the algorithm, and this scenario of optimization is called *black-box optimization.*

When optimizing in a black-box scenario, the number of calls to the subroutine is of great importance, since the other operations of an RSH are usually very cheap. Thus, the subroutine calls dominate the overall cost of the respective algorithm. Coming from this point of view, Droste et al. [DJW06] introduced a new complexity theory for RSHs: the so-called black-box complexity (BBC). In this model, a randomized search heuristic is assumed to solve a problem by querying the subroutine for potential solutions (modeled as bit strings) and then receiving feedback in form of the solution's quality. The worst-case complexity of a specific algorithm on a problem class is defined as the expected number of queries to the subroutine on a worst-case instance of this class; and the BBC of a class is the best possible worst-case complexity, taken over all black-box algorithms.

Although this definition captures the initial idea of subroutine calls very well, it does not prohibit the algorithm to be highly problem-specific. Droste et al. [DJW02], for example, showed that the Max-Clique problem, which is NP-hard in classical complexity theory, has a polynomial BBC. This is due to the respective algorithm learning the instance via queries and then performing costly offline computations, which do not increase the BBC. This highlights that the BBC alone does not necessarily provide sufficient information about the true complexity of a problem. In addition to that, Anil and Wiegand [AW09] showed that the black-box complexity of OneMax is $\Theta\big(n/\log(n)\big)$,[14] whereas traditional EAs, such as the (1 + 1) EA, usually have an expected complexity of $\Theta\big(n\log(n)\big)$ on OneMax [DJW02] (see Table 10.1).

In order to focus on solvers that are more problem-agnostic, Lehre and Witt [LW12] restricted the BBC model introduced by Droste et al. [DJW06] such that the algorithms considered can only make use of *unbiased* variation operators when querying the next bit string, and the algorithms are not allowed to see the representational structure of solutions during selection. This restricted variant is called *unbiased black-box complexity. Unbiased* means that the respective algorithm performs the same when given two problem instances where one is a perturbation of the other, restricted to permutations and bit flips. Hence, the algorithm is unbiased with respect to 0s and 1s as well as their positions, and it has to treat them the same way. Rowe and Vose [RV11] proposed a more general definition of *unbiased*, which focuses on the sequence of queried bit strings (the *trace*) of the algorithm under consideration. They call an algorithm unbiased if it behaves the same under any permutation of the search space that the problem class is closed under. When considering problem classes closed under the aforementioned perturbations, this definition results in the unbiased BBC from Lehre and Witt [LW12] without the restriction on the selection process. Further, Rowe and Vose [RV11] showed that, for any black-box algorithm on a problem class, there exists an unbiased algorithm (in their sense) that is at least as good. This means that the BBC of a problem class is the same as its unbiased BBC. Hence,

[14] Actually, this result dates back to Erdős and Rényi [ER63], who proved it in the context of information theory.

it suffices to study unbiased algorithms when one is interested in the BBC of a class.

Doerr et al. [DDK14] showed that the unbiased BBC, in the sense of Lehre and Witt [LW12], of different subproblems of the NP-complete Partition problem is polynomial when using higher-arity operators. This has lead to other restricted BBCs [DW14a; DW14b] or combinations of them [DL17]. However, the original unbiased BBC model is still considered an important complexity measure up to this date [BLS14; DDK15; DDY16b; Doe+11a; DW14c].

Surprisingly, all of the results for unbiased BBC so far only considered classical EAs although EDAs fit very well into the black-box model too. We combine the up to now unrelated fields of unbiased BBC and EDAs, and we characterize unbiasedness for the entire class of n-Bernoulli-λ-EDAs. This characterization (Theorem 5.5) shows strong similarities to the original definition of unbiased algorithms, introduced by Lehre and Witt [LW12], and it provides insights into how the properties of an unbiased population-based algorithm extend to EDAs. We then consider more restricted update schemes, leading to characterizations that are easier to verify than the general one. We especially find a very concise characterization for locally updating n-Bernoulli-λ-EDAs. We also prove that all of the EDAs mentioned in Section 4.3 are unbiased.

Interestingly, to the best of our knowledge, all common EDAs are unbiased. One possible explanation for this may be that the sample procedure is unbiased and that the samples are not altered afterward. Thus, a bias could only be introduced via the way the probability distribution is updated. However, as we show in Section 5.4, at least for the n-Bernoulli-λ-EDA framework, such an update usually exhibits a property that strongly favors unbiasedness.

5.2 Automorphisms of the Hypercube

Lehre and Witt [LW12] define unbiasedness with respect to variation operators over $\{0, 1\}^n$. These operators draw bit strings according to probability distributions that are invariant under automorphisms of $\{0, 1\}^n$: permutations and $\oplus$ (XOR) operations on bit strings. We extend these functions to $[0, 1]^n$ to be able to use them directly on the frequency vectors of n-Bernoulli-λ-EDAs.

Let $\overline{\sigma}$ be a permutation of $[n]$. We overload this notation and call a function $\sigma\colon [0, 1]^n \to [0, 1]^n$ a *permutation* if and only if it rearranges the elements of its input according to $\overline{\sigma}$ such that, for all $\boldsymbol{p} \in [0, 1]^n$ and all $i, j \in [n]$ with

$\overline{\sigma}(i) = j$, we have $\sigma(\boldsymbol{p})_j = \boldsymbol{p}_i$. In this case, we say that σ maps position i to position j. Further, we call a function $\chi\colon [0,1]^n \to [0,1]^n$ a *complementation* if and only if, for each position, it either takes the complement of the value at this position or does not change it. That is, for all $\boldsymbol{p} \in [0,1]^n$ and all $i \in [n]$, we have $\chi(\boldsymbol{p})_i = 1 - \boldsymbol{p}_i$ or $\chi(\boldsymbol{p})_i = \boldsymbol{p}_i$.

We call any bijection $\alpha\colon \{0,1\}^n \to \{0,1\}^n$ that preserves the Hamming distance d_{H} a *Hamming automorphism*. It is well-known that there are exactly $2^n n!$ Hamming automorphisms.[15] Note that this means that any Hamming automorphism α can be denoted as the composition of a permutation σ and a complementation χ over $\{0,1\}^n$, that is, $\alpha = \chi \circ \sigma$. We are now interested in the superspace $[0,1]^n$ of $\{0,1\}^n$ and use the metric d defined as follows:

$$\forall \boldsymbol{x}, \boldsymbol{y} \in [0,1]^n \colon d(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{n} |\boldsymbol{x}_i - \boldsymbol{y}_i| \,.$$

This corresponds to the 1-norm. It naturally extends the Hamming distance on $\{0,1\}^n$.

We say that $\alpha\colon [0,1]^n \to [0,1]^n$ is an *isometric automorphism* (of $[0,1]^n$) if it is bijective and *distance-preserving*, that is, for all $\boldsymbol{x}, \boldsymbol{y} \in [0,1]^n$, we have $d(\boldsymbol{x}, \boldsymbol{y}) = d\big(\alpha(\boldsymbol{x}), \alpha(\boldsymbol{y})\big)$.

The following lemma shows that the Hamming automorphisms are in a one-to-one correspondence with the isometric automorphisms.

▶ **Lemma 5.1.** For any isometric automorphism α, let α_{H} denote α restricted to $\{0,1\}^n$. Let two isometric automorphisms α and β be given. It then holds that

1. α_{H} is a Hamming automorphism,
2. if $\alpha_{\mathrm{H}} = \beta_{\mathrm{H}}$, then $\alpha = \beta$, and that,
3. for every Hamming automorphism η, there exists an isometric automorphism α such that $\alpha_{\mathrm{H}} = \eta$. ◀

Proof. **Regarding point 1,** it suffices to show that α maps any $\boldsymbol{x} \in \{0,1\}^n$ to $\{0,1\}^n$. Let $\overline{\boldsymbol{x}}$ be the component-wise complement of $\boldsymbol{x}$. We then have $d(\boldsymbol{x}, \overline{\boldsymbol{x}}) = n$.

[15] This can be seen as follows: there are 2^n possible choices for $\alpha(\boldsymbol{0})$. After $\alpha(\boldsymbol{0})$ is fixed, the n 1-neighbors of $\boldsymbol{0}$ can be mapped in an arbitrary (but bijective) manner to the n 1-neighbors of $\alpha(\boldsymbol{0})$, for which there are $n!$ choices. After these mappings are determined, the Hamming automorphism is completely determined.

Thus, $d\big(\alpha(\boldsymbol{x}), \alpha(\overline{\boldsymbol{x}})\big) = n$. Since only pairs of points from $\{0,1\}^n$ can be of distance n in $[0,1]^n$, we get that $\alpha(\boldsymbol{x}) \in \{0,1\}^n$, as desired.

Regarding point 2, for all $j \in [n]$, we use $\mathbf{e}^{(j)}$ to denote the bit string which is 0 everywhere except in position j. We now show the following claim:

$$\forall \boldsymbol{x}, \boldsymbol{y} \in [0,1]^n \colon \big(\forall z \in \{0,1\}^n \colon d(\boldsymbol{x}, z) = d(\boldsymbol{y}, z)\big) \rightarrow \boldsymbol{x} = \boldsymbol{y} \; . \tag{5.1}$$

Let $\boldsymbol{x}, \boldsymbol{y} \in [0,1]^n$ such that, for all $z \in \{0,1\}^n$, it holds that $d(\boldsymbol{x}, z) = d(\boldsymbol{y}, z)$. We now get for all $\boldsymbol{a} \in [0,1]^n$ and $j \in [n]$ that

$$\begin{aligned} d(\boldsymbol{a}, \mathbf{0}) - d\big(\boldsymbol{a}, \mathbf{e}^{(j)}\big) &= \sum_{i=1}^{n} \boldsymbol{a}_i - \left(\sum_{i=1}^{j-1} \boldsymbol{a}_i + (1 - \boldsymbol{a}_j) + \sum_{i=j+1}^{n} \boldsymbol{a}_i\right) \\ &= 2\boldsymbol{a}_j - 1 \; . \end{aligned}$$

Thus, we get for all $j \in [n]$ that

$$\begin{aligned} 2\boldsymbol{x}_j - 1 &= d(\boldsymbol{x}, \mathbf{0}) - d\big(\boldsymbol{x}, \mathbf{e}^{(j)}\big) \\ &= d(\boldsymbol{y}, \mathbf{0}) - d\big(\boldsymbol{y}, \mathbf{e}^{(j)}\big) \\ &= 2\boldsymbol{y}_j - 1 \; , \end{aligned}$$

which shows $\boldsymbol{x} = \boldsymbol{y}$.

Now, suppose that $\alpha_{\mathrm{H}} = \beta_{\mathrm{H}}$. Let $\boldsymbol{x} \in [0,1]^n$ be given. We have for all $z \in \{0,1\}^n$ that $d\big(\alpha(\boldsymbol{x}), \alpha(z)\big) = d(\boldsymbol{x}, z) = d\big(\beta(\boldsymbol{x}), \beta(z)\big)$, because α and β are isometric. Since $\alpha_{\mathrm{H}} = \beta_{\mathrm{H}}$ are Hamming automorphisms, we have for all $z \in \{0,1\}^n$ that $d\big(\alpha(\boldsymbol{x}), z\big) = d\big(\beta(\boldsymbol{x}), z\big)$. Using proposition 5.1, we see that $\alpha(\boldsymbol{x}) = \beta(\boldsymbol{x})$, as desired.

Regarding point 3, let η be given. We construct a distance-preserving extension α of η to $[0,1]^n$ such that $\alpha_{\mathrm{H}} = \eta$. For all $\boldsymbol{x} \in [0,1]^n$, let

$$\alpha(\boldsymbol{x}) = \eta(\mathbf{0}) + \sum_{i=1}^{n} \big(\eta\big(\mathbf{e}^{(i)}\big) - \eta(\mathbf{0})\big) \cdot \boldsymbol{x}_i \; . \tag{5.2}$$

Note that $\eta\big(\mathbf{e}^{(i)}\big)$ differs from $\eta(\mathbf{0})$ in exactly one position, since η is distance-preserving. This position does not necessarily have to be i, since η can perform a permutation. Assume that this position is $j \in [n]$. Then, for any $k \in [n]$, the value $\big(\eta\big(\mathbf{e}^{(i)}\big) - \eta(\mathbf{0})\big)_k$ is 1 or -1 if $k = j$ and 0 otherwise. Note further that, for each unit vector, a component different from all other unit vectors is non-zero in $\eta\big(\mathrm{e}^{(i)}\big) - \eta(\mathbf{0})$, since η is bijective. Overall (assuming that position i is mapped to position j), for any $\boldsymbol{x} \in [0,1]^n$, this means that $\alpha(\boldsymbol{x})_j$ is completely determined by $\eta(\mathbf{0})_j + \big(\eta\big(\mathrm{e}^{(i)}\big) - \eta(\mathbf{0})\big)_j \cdot \boldsymbol{x}_i$. This shows that α is bijective (albeit not necessarily on $[0,1]^n$), since the inverse can be computed uniquely, due to the unique partition of the sum into the different components and due to η being a bijection.

We now show that **$\boldsymbol{\alpha}$ maps bijectively to $[0, 1]^n$.** Again, assume that position i is mapped to position j, and let $\boldsymbol{x} \in [0,1]^n$. Recall that η can only perform complementations additionally to permutations. We show that

$$\eta(\mathbf{0})_j + \big(\eta\big(\mathrm{e}^{(i)}\big) - \eta(\mathbf{0})\big)_j \cdot \boldsymbol{x}_i$$

is a bijection of $[0,1]$. It then follows that α is a bijection of $[0,1]^n$, since the components do not interfere with one another.

We make a case distinction with respect to whether η takes the complement at position j (after performing the permutation) or not.

Case $\boldsymbol{\eta(x)_j = x_i}$. Then, $\eta(\mathbf{0})_j = 0$ and $\big(\eta\big(\mathrm{e}^{(i)}\big) - \eta(\mathbf{0})\big)_j = 1$. Thus, $\alpha(\boldsymbol{x})_j = \boldsymbol{x}_i$, which is a bijection from $[0,1]$ to $[0,1]$.

Case $\boldsymbol{\eta(x)_j = 1 - x_i}$. Then, $\eta(\mathbf{0})_j = 1$ and $\big(\eta\big(\mathrm{e}^{(i)}\big) - \eta(\mathbf{0})\big)_j = -1$. Thus, $\alpha(\boldsymbol{x})_j = 1 - \boldsymbol{x}_i$, which is also a bijection from $[0,1]$ to $[0,1]$.

Combining both cases yields that $\alpha(\boldsymbol{x})_i$ is a bijection of $[0,1]$ for every $\boldsymbol{x}$ and every i. Thus, α is a bijection of $[0,1]^n$, as we discussed before.

Now, we show that **$\boldsymbol{\alpha}$ is distance-preserving.** For this, let $\boldsymbol{x}, \boldsymbol{y} \in [0,1]^n$, and let $\overline{\sigma}$ be the permutation of indices that η realizes. We get

$$\begin{aligned} d\big(\alpha(\boldsymbol{x}), \alpha(\boldsymbol{y})\big) &= \sum_{i=1}^{n} \big|\alpha(\boldsymbol{x})_i - \alpha(\boldsymbol{y})_i\big| \\ &= \sum_{i=1}^{n} \Big|\eta(\mathbf{0})_{\overline{\sigma}(i)} + \big(\eta\big(\mathrm{e}^{(i)}\big) - \eta(\mathbf{0})\big)_{\overline{\sigma}(i)} \cdot \boldsymbol{x}_i \\ &\qquad - \eta(\mathbf{0})_{\overline{\sigma}(i)} - \big(\eta\big(\mathrm{e}^{(i)}\big) - \eta(\mathbf{0})\big)_{\overline{\sigma}(i)} \cdot \boldsymbol{y}_i\Big| \\ &= \sum_{i=1}^{n} \Big|\big(\eta\big(\mathrm{e}^{(i)}\big) - \eta(\mathbf{0})\big)_{\overline{\sigma}(i)}(\boldsymbol{x}_i - \boldsymbol{y}_i)\Big| \end{aligned}$$

$$= \sum_{i=1}^{n} |\boldsymbol{x}_i - \boldsymbol{y}_i| ,$$

since $d\big(\eta(\mathbf{e}^{(i)}), \eta(\mathbf{0})\big) = d\big(\mathbf{e}^{(i)}, \mathbf{0}\big) = 1$.

We now show that $\boldsymbol{\alpha}_{\mathrm{H}}$ **is equal to $\boldsymbol{\eta}$,** that is, for all $\boldsymbol{x} \in \{0,1\}^n$, it holds that $\alpha(\boldsymbol{x}) = \eta(\boldsymbol{x})$. We do so by showing that the components are the same. We handle this case very similarly to the one where we showed that α is bijective on $[0,1]^n$.

Let $i \in [n]$ be an arbitrary index, let $\boldsymbol{x} \in \{0,1\}^n$, and consider that η maps position i to position $j \in [n]$. We now make a case distinction with respect to whether η takes the complement at position j or not.

Case $\boldsymbol{\eta(x)_j = x_i}$:

$$\begin{aligned} \alpha(\boldsymbol{x})_j &= \eta(\mathbf{0})_j + \big(\eta\big(\mathbf{e}^{(i)}\big)_j - \eta(\mathbf{0})_j\big) \cdot \boldsymbol{x}_i \\ &= 0 + (1 - 0) \cdot \boldsymbol{x}_i \\ &= \boldsymbol{x}_i = \eta(\boldsymbol{x})_j . \end{aligned}$$

Case $\boldsymbol{\eta(x)_j = 1 - x_i}$:

$$\begin{aligned} \alpha(\boldsymbol{x})_j &= \eta(\mathbf{0})_j + \big(\eta\big(\mathbf{e}^{(i)}\big)_j - \eta(\mathbf{0})_j\big) \cdot \boldsymbol{x}_i \\ &= 1 + (0 - 1) \cdot \boldsymbol{x}_i \\ &= 1 - \boldsymbol{x}_i = \eta(\boldsymbol{x})_j . \end{aligned}$$

Hence, for all $j \in [n]$, we have $\alpha(\boldsymbol{x})_j = \eta(\boldsymbol{x})_j$. This finishes the proof. ■

It follows from Lemma 5.1 that each Hamming automorphism α has a unique distance-preserving extension to $[0,1]^n$. Let $\widehat{\alpha}$ denote this unique extension of α to an isometric automorphism.

5.3 Unbiased EDAs

We start by defining the general concept of unbiasedness of a black-box algorithm. Such an algorithm follows the very general framework of querying bit strings from an oracle, where each query (that is, sample) can only depend on all bit strings and their fitness queried so far. The original definition of unbiased black-box complexity was introduced by Lehre and Witt [LW12]. Since then, the definition has been stated several times in different formulations [DKW11;

Doe+11a; RV11]. Usually, the definition makes use of unbiased variation operators, which are operators that sample from distributions in a way that is invariant under permutations and complementations (thus, only implicitly capturing the time point of the query).

Rowe and Vose [RV11], however, take a different approach and define unbiasedness in terms of the sequence of queries to the oracle (the *trace*). In this definition, an algorithm is unbiased if its trace is invariant under permutations and complementations (thus, only implicitly capturing the distribution used for sampling).

We use the definition of Rowe and Vose [RV11], phrased in the sense of our previously introduced concepts of permutations and complementations of $[0, 1]^n$.

▶ **Definition 5.2 (Unbiasedness).** A *black-box algorithm* A is a mapping that takes a sequence of bit strings with corresponding fitness values and returns a bit string. Since A may be randomized, a black-box algorithm A and a fitness function f together define a sequence of random variables $(X^{(i)})_{i \in \mathbf{N}}$, where $X^{(i)}$ denotes the i-th bit string computed by A:

$$\forall i \in \mathbf{N}\colon X^{(i)} = A\Big(\big(X^{(j)}, f\big(X^{(j)}\big)\big)_{j<i}\Big) .$$

For a given algorithm A and fitness function f, we denote the dependency of $X^{(i)}$ on A and f by writing, for all $i \in \mathbf{N}$, $X^{(i)}(A, f)$.

We call a black-box algorithm A *unbiased* if and only if, for all Hamming automorphisms α, all fitness functions f, and all $i \in \mathbf{N}$, it holds that

$$X^{(i)}(A, f) = \alpha\big(X^{(i)}(A, f \circ \alpha)\big) . \tag{5.3}$$

◀

Intuitively, when given an unbiased black-box algorithm A, it performs the same when optimizing f or $f \circ \alpha$; the only difference is that A using f optimizes the unperturbed hypercube, whereas A using $f \circ \alpha$ optimizes the hypercube perturbed by α. This way, if A using f samples $\boldsymbol{x}$, A using $f \circ \alpha$ samples $\alpha^{-1}(\boldsymbol{x})$ with the same probability and queries (with the same probability) the same individual: $\boldsymbol{x} = \alpha\big(\alpha^{-1}(\boldsymbol{x})\big)$ (since $f(\boldsymbol{x}) = (f \circ \alpha)\big(\alpha^{-1}(\boldsymbol{x})\big)$). Note that an n-Bernoulli-λ-EDA A is a black-box algorithm, as it samples λ bit strings every iteration, which can trivially be sequentialized. The probability vector $\boldsymbol{p}$ of A is implicitly captured in the history of search points $X^{(i)}(A, f)$, where those with indices $i \in \{\lambda t, \dots, \lambda(t+1)-1\}$ share the same distribution, which is $\boldsymbol{p}^{(t)}$. We now focus on an invariance property of the update scheme of an n-Bernoulli-λ-EDA.

▸ **Definition 5.3 ($\mathcal{H}$-Invariance).** Let A be an n-Bernoulli-λ-EDA with update scheme φ, and let $\mathcal{H}$ be a family of bijections over $[0,1]^n$ with fixed point $\mathbf{1/2}$ such that, for all $h \in \mathcal{H}$, it holds that $h^{-1} \in \mathcal{H}$. We say that A is *$\mathcal{H}$-invariant* if and only if, for all $h \in \mathcal{H}$, all fitness functions f, all possible frequency vectors $\boldsymbol{p} \in P(A)$, and all offspring populations D that can be sampled by $\boldsymbol{p}$, it holds that

$$h\big(\varphi\big(\boldsymbol{p}, \big(\boldsymbol{x}, f(\boldsymbol{x})\big)_{\boldsymbol{x}\in D}\big)\big) = \varphi\big(h(\boldsymbol{p}), \big(h(\boldsymbol{x}), f(\boldsymbol{x})\big)_{\boldsymbol{x}\in D}\big) . \qquad ◂$$

Note that the right-hand side of this equation can be written as

$$\varphi\big(h(\boldsymbol{p}), \big(\boldsymbol{x}, (f \circ h^{-1})(\boldsymbol{x})\big)_{\boldsymbol{x}\in h(D)}\big) ,$$

that is, $h(\boldsymbol{p})$ can sample $h(D)$ (since $\boldsymbol{p}$ can sample D). In the following, if $\mathcal{H}$ is the class of all permutations, we call an $\mathcal{H}$-invariant n-Bernoulli-λ-EDA A *permutation-invariant.* If $\mathcal{H}$ is the class of all complementations, we call A *complementation-invariant.* And if $\mathcal{H}$ is the class of all isometric automorphisms, we call A *automorphism-invariant.* Note that any function in any of these families has $\mathbf{1/2}$ as a fixed point. This is important in order to ensure that A has an initial distribution that is equally fair to any function $h \in \mathcal{H}$.

▸ **Lemma 5.4.** Let A be an $\mathcal{H}$-invariant n-Bernoulli-λ-EDA. Then, for all $h \in \mathcal{H}$, all time steps t, and all $\boldsymbol{p}^{(t)} \in P^{(t)}(A)$, we have

$$h\big(\boldsymbol{p}^{(t)}\big) \in P^{(t)}(A) . \qquad ◂$$

Proof. Let h be any function of $\mathcal{H}$. We prove this lemma by induction over t.

For the **base case,** we have $\boldsymbol{p}^{(0)} = \mathbf{1/2}$, which is equal to $h(\mathbf{1/2})$, since $\mathbf{1/2}$ is a fixed point for any $h \in \mathcal{H}$. Hence, $h\big(\boldsymbol{p}^{(0)}\big) \in P^{(0)}(A)$.

For the **inductive step,** we assume that $h\big(\boldsymbol{p}^{(t)}\big) \in P^{(t)}(A)$ holds. We now show that $h\big(\boldsymbol{p}^{(t+1)}\big) \in P^{(t+1)}(A)$ holds. Due to the definition of an n-Bernoulli-λ-EDA, we have that $\boldsymbol{p}^{(t+1)} = \varphi\big(\boldsymbol{p}^{(t)}, \big(\boldsymbol{x}, f(\boldsymbol{x})\big)_{\boldsymbol{x}\in D}\big)$. And since A is $\mathcal{H}$-invariant, it holds that

$$h\big(\varphi\big(\boldsymbol{p}^{(t)}, \big(\boldsymbol{x}, f(\boldsymbol{x})\big)_{\boldsymbol{x}\in D}\big)\big) = \varphi\big(h\big(\boldsymbol{p}^{(t)}\big), \big(h(\boldsymbol{x}), f(\boldsymbol{x})\big)_{\boldsymbol{x}\in D}\big) .$$

Thus, $h\big(\boldsymbol{p}^{(t+1)}\big) = \varphi\big(h\big(\boldsymbol{p}^{(t)}\big), \big(h(\boldsymbol{x}), f(\boldsymbol{x})\big)_{\boldsymbol{x}\in D}\big)$. The latter is in $P^{(t+1)}(A)$, since

$h\big(\boldsymbol{p}^{(t)}\big) \in P^{(t)}(A)$ – due to the induction hypothesis –, which is updated by φ and, thus, creates an element in $P^{(t+1)}(A)$ by definition. ■

Lemma 5.4 shows how heavily the update scheme of an n-Bernoulli-λ-EDA is already restricted when assuming that it is invariant. We now state our main theorem, which characterizes when an n-Bernoulli-λ-EDA is unbiased. Recall that we defined $\widehat{\alpha}$ to be the unique extension of a Hamming automorphism α after Lemma 5.1.

▶ **Theorem 5.5.** Let A be an n-Bernoulli-λ-EDA. Then the following are equivalent:

1. A is unbiased.
2. For all fitness functions f, all isometric automorphisms $\widehat{\alpha}$, and all $t \in \mathbf{N}$, it holds that
$$\boldsymbol{p}^{(t)}(A, f) = \widehat{\alpha}\big(\boldsymbol{p}^{(t)}(A, f \circ \alpha)\big)\ . \tag{5.4}$$
3. A is automorphism-invariant.
4. A is permutation- and complementation-invariant. ◀

Proof. **(1) ⇒ (2).** Let f be any fitness function, and let $\widehat{\alpha}$ be any isometric automorphism. Note that $\alpha\big(X^{(i)}(A, f \circ \alpha)\big) = \widehat{\alpha}\big(X^{(i)}(A, f \circ \alpha)\big)$, since $\widehat{\alpha}$ maps bit strings to bit strings, according to Lemma 5.1.

As mentioned before, the random variables $X^{(i)}(A, f)$ with indices $i \in \{\lambda t, \dots, \lambda(t+1)-1\}$ follow the same distribution, which is $\boldsymbol{p}^{(t)}$. This means that $\boldsymbol{p}^{(t)}(A, f)$ describes the distribution of (amongst others) $X^{(\lambda t)}(A, f)$, as A samples bit strings according to its frequency vector, due to the definition of an n-Bernoulli-λ-EDA.

Since we assume that A is unbiased, the random variable $X^{(\lambda t)}(A, f)$ is equal to $\widehat{\alpha}\big(X^{(\lambda t)}(A, f \circ \alpha)\big)$ for any t, and, thus, their distributions are equal as well. This means that equation (5.4) holds.

(1) ⇐ (2). Since equation (5.4) holds and since $\boldsymbol{p}^{(t)}(A, f)$ and $\boldsymbol{p}^{(t)}(A, f \circ \alpha)$ describe distributions over bit strings, it follows that $X^{(\lambda t)}(A, f) = \widehat{\alpha}\big(X^{(\lambda t)}(A, f \circ \alpha)\big)$ for all t.

Due to an n-Bernoulli-λ-EDA sampling λ individuals every iteration, it follows that $X^{(\lambda t)}(A, f)$ is equal to $X^{(i)}(A, f)$, where $i \in \{\lambda t + 1, \dots, \lambda(t+1) - 1\}$. This shows equation (5.3) for the remaining indices i and completes this direction.

(2) ⇒ (3). Let f be any fitness function, let $\widehat{\alpha}$ be any isometric isomorphism, let $t \in \mathbf{N}$ be any iteration, let $\boldsymbol{p}^{(t)} = \boldsymbol{p}^{(t)}(A, f)$, and let $\widetilde{\boldsymbol{p}}^{(t)} = \boldsymbol{p}^{(t)}(A, f \circ \alpha)$. Note that due to the definition of $P(A)$, we cover all $\boldsymbol{p} \in P(A)$ by choosing an arbitrary t and considering $\boldsymbol{p}^{(t)}$. We now show that A is automorphism-invariant.

Since equation (5.4) holds, we have that $\boldsymbol{p}^{(t+1)} = \widehat{\alpha}\big(\widetilde{\boldsymbol{p}}^{(t+1)}\big)$. Due to the definition of an update, this means that

$$\varphi\big(\boldsymbol{p}^{(t)}, \big(\boldsymbol{x}, f(\boldsymbol{x})\big)_{\boldsymbol{x} \in D}\big) = \widehat{\alpha}\Big(\varphi\big(\widetilde{\boldsymbol{p}}^{(t)}, \big(\boldsymbol{x}, (f \circ \alpha)(\boldsymbol{x})\big)_{\boldsymbol{x} \in \widetilde{D}}\big)\Big) , \tag{5.5}$$

where D should be sampled by $\boldsymbol{p}^{(t)}$ and $\widetilde{D}$ should be sampled by $\widetilde{\boldsymbol{p}}^{(t)}$.

Due to equation (5.4), we further have that $\boldsymbol{p}^{(t)} = \widehat{\alpha}\big(\widetilde{\boldsymbol{p}}^{(t)}\big)$, which is equivalent to $\widehat{\alpha}^{-1}\big(\boldsymbol{p}^{(t)}\big) = \widetilde{\boldsymbol{p}}^{(t)}$, since $\widehat{\alpha}$ is invertible.

Substituting this into equation (5.5) and inverting $\widehat{\alpha}$ leads to

$$\widehat{\alpha}^{-1}\Big(\varphi\big(\boldsymbol{p}^{(t)}, \big(\boldsymbol{x}, f(\boldsymbol{x})\big)_{\boldsymbol{x} \in D}\big)\Big) = \varphi\big(\widehat{\alpha}^{-1}\big(\boldsymbol{p}^{(t)}\big), \big(\boldsymbol{x}, (f \circ \alpha)(\boldsymbol{x})\big)_{\boldsymbol{x} \in \widetilde{D}}\big) .$$

Note that $\widetilde{D} = \widehat{\alpha}^{-1}(D)$, since $\widetilde{\boldsymbol{p}}^{(t)} = \widehat{\alpha}^{-1}\big(\boldsymbol{p}^{(t)}\big)$, and that $f \circ \alpha = f \circ \widehat{\alpha}$, since α and $\widehat{\alpha}$ are equal on bit strings. Thus, the above equation is equivalent to A being automorphism-invariant, as we discussed after Definition 5.3.

(3) ⇒ (2). For this direction, we use the same notation as in the previous direction. We now show that equation (5.4) holds for all t.

For the **base case,** we have $\boldsymbol{p}^{(0)} = \widetilde{\boldsymbol{p}}^{(0)} = \mathbf{1}/2$ by the initialization step of an n-Bernoulli-λ-EDA. Permutations obviously do not change this equality, since all frequencies are the same. Complementations also do not change anything, since $1 - 1/2 = 1/2$. Hence, $\boldsymbol{p}^{(0)} = \widehat{\alpha}\big(\widetilde{\boldsymbol{p}}^{(0)}\big)$, as we desire.

For the **inductive step,** we assume that equation (5.4) holds up to a $t \in \mathbf{N}$, which is equivalent to $\widehat{\alpha}^{-1}\big(\boldsymbol{p}^{(t)}\big) = \widetilde{\boldsymbol{p}}^{(t)}$. When making an update, we get

$$\begin{aligned}\widetilde{\boldsymbol{p}}^{(t+1)} &= \varphi\big(\widetilde{\boldsymbol{p}}^{(t)}, \big(\boldsymbol{x}, (f \circ \alpha)(\boldsymbol{x})\big)_{\boldsymbol{x} \in \alpha^{-1}(D)}\big)\\ &= \varphi\big(\widehat{\alpha}^{-1}\big(\boldsymbol{p}^{(t)}\big), \big(\alpha^{-1}(\boldsymbol{x}), f(\boldsymbol{x})\big)_{\boldsymbol{x} \in D}\big) ,\end{aligned}$$

where D is a random variable denoting an offspring population that can be sampled by $\boldsymbol{p}^{(t)}$, and where we used the induction hypothesis and that $(f \circ \alpha)\big(\alpha^{-1}(\boldsymbol{x})\big) = f(\boldsymbol{x})$.

Since A is automorphism-invariant, we can pull $\widehat{\alpha}^{-1}$ in front of φ. Hence, we get

$$\varphi\big(\widehat{\alpha}^{-1}(\boldsymbol{p}^{(t)}), \big(\alpha^{-1}(\boldsymbol{x}), f(\boldsymbol{x})\big)_{\boldsymbol{x}\in D}\big) = \widehat{\alpha}^{-1}\Big(\varphi\big(\boldsymbol{p}^{(t)}, \big(\boldsymbol{x}, f(\boldsymbol{x})\big)_{\boldsymbol{x}\in D}\big)\Big) ,$$

which is equivalent to $\widetilde{\boldsymbol{p}}^{(t+1)}$, as we have seen previously.

Substituting $\varphi\big(\boldsymbol{p}^{(t)}, \big(\boldsymbol{x}, f(\boldsymbol{x})\big)_{\boldsymbol{x}\in D}\big)$ with $\boldsymbol{p}^{(t+1)}$, we get

$$\widetilde{\boldsymbol{p}}^{(t+1)} = \widehat{\alpha}^{-1}\big(\boldsymbol{p}^{(t+1)}\big) \Leftrightarrow \boldsymbol{p}^{(t+1)} = \widehat{\alpha}\big(\widetilde{\boldsymbol{p}}^{(t+1)}\big) ,$$

which is what we wanted to show.

(3) ⇒ (4). This direction is trivial, since every permutation and every complementation is an isometric automorphism.

(4) ⇒ (3). Let $\widehat{\alpha}$ be any isometric automorphism, and let f be any fitness function. Let σ be a permutation and χ be a complementation such that $\widehat{\alpha} = \chi\circ\sigma$.

Since A is permutation-invariant, we get, for any $\boldsymbol{p} \in P(A)$ and any D sampled by $\boldsymbol{p}$, that

$$\begin{aligned}\sigma\Big(\varphi\big(\boldsymbol{p}, \big(\boldsymbol{x}, f(\boldsymbol{x})\big)_{\boldsymbol{x}\in D}\big)\Big) &= \varphi\big(\sigma(\boldsymbol{p}), \big(\sigma(\boldsymbol{x}), f(\boldsymbol{x})\big)_{\boldsymbol{x}\in D}\big)\\ &= \varphi\big(\sigma(\boldsymbol{p}), \big(\boldsymbol{x}, (f\circ\sigma^{-1})(\boldsymbol{x})\big)_{\boldsymbol{x}\in\sigma(D)}\big) ,\end{aligned}$$

where $\sigma(\boldsymbol{p}) \in P(A)$, due to Lemma 5.4, and $\sigma(D)$ is an offspring population that can be sampled by $\sigma(\boldsymbol{p})$. Thus, since A is also complementation-invariant and since $(f\circ\sigma^{-1})$ is also a fitness function, we get

$$\begin{aligned}\chi\Big(\varphi\big(\sigma(\boldsymbol{p}), \big(\boldsymbol{x}, (f\circ\sigma^{-1})(\boldsymbol{x})\big)_{\boldsymbol{x}\in\sigma(D)}\big)\Big) &= \varphi\big(\chi(\sigma(\boldsymbol{p})), \big(\chi(\boldsymbol{x}), (f\circ\sigma^{-1})(\boldsymbol{x})\big)_{\boldsymbol{x}\in\sigma(D)}\big)\\ &= \varphi\big(\chi(\sigma(\boldsymbol{p})), \big(\chi(\sigma(\boldsymbol{x})), f(\boldsymbol{x})\big)_{\boldsymbol{x}\in D}\big)\\ &= \varphi\big(\widehat{\alpha}(\boldsymbol{p}), \big(\alpha(\boldsymbol{x}), f(\boldsymbol{x})\big)_{\boldsymbol{x}\in D}\big) .\end{aligned}$$

We now combine both cases and get

$$\begin{aligned}\widehat{\alpha}\Big(\varphi\big(\boldsymbol{p}, \big(\boldsymbol{x}, f(\boldsymbol{x})\big)_{\boldsymbol{x}\in D}\big)\Big) &= \chi\Big(\sigma\Big(\varphi\big(\boldsymbol{p}, \big(\boldsymbol{x}, f(\boldsymbol{x})\big)_{\boldsymbol{x}\in D}\big)\Big)\Big)\\ &= \varphi\big(\widehat{\alpha}(\boldsymbol{p}), \big(\alpha(\boldsymbol{x}), f(\boldsymbol{x})\big)_{\boldsymbol{x}\in D}\big) ,\end{aligned}$$

which means that A is automorphism-invariant.

Overall, we showed all equivalences that we stated. ■

Theorem 5.5 only considers n-Bernoulli-λ-EDAs without a margin. However, we would like to note that imposing a margin via equation (4.2) onto an unbiased n-Bernoulli-λ-EDA A leaves it unbiased. The reason for this is that equation (4.2) can be interpreted as a new update scheme of another n-Bernoulli-λ-EDA A' with no externally imposed margin. Since a margin b is the same for all positions and affects 0 and 1 in a symmetric fashion (it limits the frequencies to $[b, 1-b]^n$), Lemma 5.4 still applies. Hence, A' is still automorphism-invariant (since A is by assumption) and, due to Theorem 5.5, it is unbiased.

5.4 Decomposability

Theorem 5.5 characterizes the unbiasedness for all n-Bernoulli-λ-EDAs, that is, its statement is very general, and some of the requirements may be difficult to check. Common n-Bernoulli-λ-EDAs have a more specific update scheme than the one presented in the framework (Algorithm 1). Thus, we can give more precise statements on unbiasedness if we focus on meaningful subsets of update schemes.

In the following, we consider decomposable EDAs (Definition 4.1). Within this class, we look at singularly decomposable EDAs and such that we call *self-complementary* (Definition 5.8). The former (which includes all locally updating algorithms) are permutation-invariant, the latter are complementation-invariant. Hence, an algorithm having both properties is unbiased.

▶ **Theorem 5.6.** Let A be a decomposable n-Bernoulli-λ-EDA. Then the following are equivalent:

1. A is singularly decomposable.
2. A is permutation-invariant. ◀

Proof. **(1) ⇒ (2).** Let $\dot{\varphi}$ denote the update function used by A for every frequency, and let σ be any permutation. Consider that σ maps position i to position j.

Since A is singularly decomposable, we get

$$\begin{aligned}\varphi\big(\sigma(\boldsymbol{p}), \big(\sigma(\boldsymbol{x}), f(\boldsymbol{x})\big)_{\boldsymbol{x}\in D}\big)_j &= \dot{\varphi}\big(\sigma(\boldsymbol{p})_j, \big(\sigma(\boldsymbol{x})_j, f(\boldsymbol{x})\big)_{\boldsymbol{x}\in D}\big)\\ &= \dot{\varphi}\big(\boldsymbol{p}_i, \big(\boldsymbol{x}_i, f(\boldsymbol{x})\big)_{\boldsymbol{x}\in D}\big) = \sigma\Big(\varphi\big(\boldsymbol{p}, \big(\boldsymbol{x}, f(\boldsymbol{x})\big)_{\boldsymbol{x}\in D}\big)\Big)_j .\end{aligned}$$

And since i and j are arbitrary, it follows that A is permutation-invariant.

(2) ⇒ (1). We prove this direction by contraposition. Hence, assume that A is not singularly decomposable. Since A is still decomposable, there exist at least two different update functions $\dot{\varphi}_i$ and $\dot{\varphi}_j$.

Let f^* be a fitness function, $\boldsymbol{p}^* \in P(A)$ a possible frequency vector, and D^* an offspring population that can be sampled by $\boldsymbol{p}^*$ such that

$$\dot{\varphi}_i\big(\boldsymbol{p}_i^*, \big(\boldsymbol{x}_i, f^*(\boldsymbol{x})\big)_{\boldsymbol{x}\in D^*}\big) \neq \dot{\varphi}_j\big(\boldsymbol{p}_i^*, \big(\boldsymbol{x}_i, f^*(\boldsymbol{x})\big)_{\boldsymbol{x}\in D^*}\big) \ . \tag{5.6}$$

Note that only the index of $\dot{\varphi}$ changes, since the functions are different for the same input.

Let σ^* be a permutation that maps i to j. Similar to the direction before, we have

$$\begin{aligned}\sigma^*\Big(\varphi\big(\boldsymbol{p}^*, \big(\boldsymbol{x}, f^*(\boldsymbol{x})\big)_{\boldsymbol{x}\in D^*}\big)\Big)_j &= \dot{\varphi}_i\big(\boldsymbol{p}_i^*, \big(\boldsymbol{x}_i, f^*(\boldsymbol{x})\big)_{\boldsymbol{x}\in D^*}\big) \text{ and}\\ \varphi\big(\sigma^*(\boldsymbol{p}^*), \big(\sigma^*(\boldsymbol{x}), f^*(\boldsymbol{x})\big)_{\boldsymbol{x}\in D^*}\big)_j &= \dot{\varphi}_j\big(\boldsymbol{p}_i^*, \big(\boldsymbol{x}_i, f^*(\boldsymbol{x})\big)_{\boldsymbol{x}\in D^*}\big) \ .\end{aligned}$$

Using inequality (5.6), we get by combining the two above equations that

$$\sigma^*\Big(\varphi\big(\boldsymbol{p}^*, \big(\boldsymbol{x}, f^*(\boldsymbol{x})\big)_{\boldsymbol{x}\in D^*}\big)\Big)_j \neq \varphi\big(\sigma^*(\boldsymbol{p}^*), \big(\sigma^*(\boldsymbol{x}), f^*(\boldsymbol{x})\big)_{\boldsymbol{x}\in D^*}\big)_j \ ,$$

which means that A is not permutation-invariant. This concludes the proof. ■

Theorem 5.6 provides a very strong guideline for creating n-Bernoulli-λ-EDAs: if your algorithm should be decomposable and unbiased, then it has to be singularly decomposable. This statement is helpful, since it is not hard to come up with a singularly decomposable n-Bernoulli-λ-EDA. But it also tells us that there is no sense in trying any more complicated update schemes if we want a decomposable one. A question that might now be asked is whether unbiasedness implies decomposability. If so, we would know that only singularly decomposable n-Bernoulli-λ-EDAs could be unbiased. Unfortunately, there exist algorithms that are not decomposable but still unbiased, as we will argue next.

5.4.1 An Unbiased Non-decomposable EDA

The idea of the algorithm we are going to present is the following: the update scheme is closely related to the one of the λ-MMAS$_{\text{IB}}$. However, the parameter ρ is adaptive with respect to the current frequency vector. The more each frequency is away from its center 1/2, the greater ρ gets. This means that the step size is increasing the further the algorithm commits to a certain direction for its frequencies. Formally, for any $\rho \in [0,1]$ and any frequency vector $\boldsymbol{p}$, let

$$\rho_{\boldsymbol{p}} = \rho\left(\frac{1}{2} + \frac{1}{n}\sum_{k=1}^{n}\left|\boldsymbol{p}_k - \frac{1}{2}\right|\right) .$$

Consider the following update scheme:

$$\varphi\big(\boldsymbol{p}, (\boldsymbol{x}, f(\boldsymbol{x}))_{\boldsymbol{x}\in D}\big)_i = (1-\rho_{\boldsymbol{p}})\boldsymbol{p}_i + \boldsymbol{x}_i^{(1)} \cdot \rho_{\boldsymbol{p}} \ , \tag{5.7}$$

where $\boldsymbol{x}^{(1)}$ – the individual of rank 1 – is determined uniformly at random in the case of ties. Note that an n-Bernoulli-λ-EDA with such an update scheme is not decomposable, as $\rho_{\boldsymbol{p}}$ uses all frequencies.

▶ **Proposition 5.7.** Let A be the n-Bernoulli-λ-EDA with update scheme φ as given in equation (5.7). Then A is unbiased. ◀

Proof. We use Theorem 5.5 and show that A is permutation- and complementation-invariant.

We start by showing that ***A* is permutation-invariant.** For this, let σ be an arbitrary permutation that maps position i to position j. We get

$$\begin{aligned}\sigma\Big(\varphi\big(\boldsymbol{p}, (\boldsymbol{x}, f(\boldsymbol{x}))_{\boldsymbol{x}\in D}\big)\Big)_j &= (1-\rho_{\boldsymbol{p}})\boldsymbol{p}_i + \boldsymbol{x}_i^{(1)} \cdot \rho_{\boldsymbol{p}} \\ &= (1-\rho_{\sigma(\boldsymbol{p})})\sigma(\boldsymbol{p})_j + \sigma\big(\boldsymbol{x}^{(1)}\big)_j \cdot \rho_{\sigma(\boldsymbol{p})} \\ &= \varphi\big(\sigma(\boldsymbol{p}), (\sigma(\boldsymbol{x}), f(\boldsymbol{x}))_{\boldsymbol{x}\in D}\big)_j \ ,\end{aligned}$$

as we can re-order the summands in $\rho_{\boldsymbol{p}}$ without changing its value.

Note that the updates $\sigma\big(\varphi\big(\boldsymbol{p}, (\boldsymbol{x}, f(\boldsymbol{x}))_{\boldsymbol{x}\in D}\big)\big)$ and $\varphi\big(\sigma(\boldsymbol{p}), (\sigma(\boldsymbol{x}), f(\boldsymbol{x}))_{\boldsymbol{x}\in D}\big)$ are both random variables. Thus, $\boldsymbol{x}^{(1)}$ does not necessarily have to denote the same individual when looking at both cases. However, due to our assumptions on $\boldsymbol{x}^{(1)}$, we do not have to consider it changing in-between the equations, since $\boldsymbol{x}^{(1)}$ and $\sigma\big(\boldsymbol{x}^{(1)}\big)$ both have fitness $f\big(\boldsymbol{x}^{(1)}\big)$ and, thus, get chosen for an update with equal probability in both cases. This shows that A is permutation-invariant.

We now show that ***A* is complementation-invariant.** For this, let χ be an arbitrary complementation. Note that $|\chi(\boldsymbol{p})_k - 1/2| = |\boldsymbol{p}_k - 1/2|$ and, thus, $\rho_{\boldsymbol{p}} = \rho_{\chi(\boldsymbol{p})}$.

Consider that χ takes the complement at position i; the other case of χ not changing anything at position i can be shown analogously and more easily. Thus, we get

$$\begin{aligned}
\varphi\big(\chi(\boldsymbol{p}), \big(\chi(\boldsymbol{x}), f(\boldsymbol{x})\big)_{\boldsymbol{x} \in D}\big)_i &= \big(1 - \rho_{\chi(\boldsymbol{p})}\big)\chi(\boldsymbol{p})_i + \chi\big(\boldsymbol{x}^{(1)}\big)_i \cdot \rho_{\chi(\boldsymbol{p})} \\
&= \big(1 - \rho_{\boldsymbol{p}}\big)(1 - \boldsymbol{p}_i) + \big(1 - \boldsymbol{x}_i^{(1)}\big) \cdot \rho_{\boldsymbol{p}} \\
&= 1 - \rho_{\boldsymbol{p}} - \big(1 - \rho_{\boldsymbol{p}}\big)\boldsymbol{p}_i + \rho_{\boldsymbol{p}} - \boldsymbol{x}_i^{(1)} \cdot \rho_{\boldsymbol{p}} \\
&= 1 - \big((1 - \rho_{\boldsymbol{p}})\boldsymbol{p}_i + \boldsymbol{x}_i^{(1)} \cdot \rho_{\boldsymbol{p}}\big) \\
&= \chi\Big(\varphi\big(\boldsymbol{p}, \big(\boldsymbol{x}, f(\boldsymbol{x})\big)_{\boldsymbol{x} \in D}\big)\Big)_i ,
\end{aligned}$$

which means that A is complementation-invariant. This completes the proof. ■

The update scheme presented in equation (5.7) uses the entire frequency vector to update any frequency $\boldsymbol{p}_i$, but it only uses the i-th bit for each individual in the offspring population D. Thus, it may be that an operator that makes use of D in a non-bit-wise manner might make the algorithm biased. However, there exist operators that use D entirely and still result in unbiased algorithms.

One example is the operator $s(D)$, which we will define after introducing the following auxiliary function (for $i \in [n]$):

$$s(D, i) = \begin{cases} 1 & \text{if } \sum_{\boldsymbol{x} \in D} \boldsymbol{x}_i \in \{0, \lambda\} , \\ 0 & \text{otherwise.} \end{cases}$$

Note that $s(D, i)$ is 1 if and only if all individuals have the same value at index i; otherwise, it is 0. Now, let $s(D) = (1/n) \sum_{k=1}^{n} s(D, k)$. Consider the update scheme

$$\varphi\big(\boldsymbol{p}, \big(\boldsymbol{x}, f(\boldsymbol{x})\big)_{\boldsymbol{x} \in D}\big)_i = \big(1 - \rho \cdot s(D)\big)\boldsymbol{p}_i + \boldsymbol{x}_i^{(1)} \cdot \rho \cdot s(D) . \tag{5.8}$$

Analogous to Proposition 5.7, this update scheme is unbiased. Essentially, this boils down to $s(D)$ being permutation- and complementation-invariant.

Finally, one can combine the update schemes of equations (5.7) and (5.8), and the resulting n-Bernoulli-λ-EDA is still unbiased. Thus, the class of unbiased decomposable n-Bernoulli-λ-EDAs is a proper subset of the class of all unbiased n-Bernoulli-λ-EDAs. However, many theoretically analyzed n-Bernoulli-λ-EDAs – such as the ones in Section 4.3 – are singularly decomposable.

5.4.2 Unbiased Decomposable EDAs

We now look at complementation invariance for decomposable n-Bernoulli-λ-EDAs. For this, we define the following property.

▶ **Definition 5.8 (Self-Complementarity).** Let A be a singularly decomposable n-Bernoulli-λ-EDA with update scheme φ, using the update function $\dot{\varphi}$. We say that A is *self-complementary* if and only if, for all fitness functions f, all possible frequency vectors $\boldsymbol{p} \in P(A)$, all offspring populations D that can be sampled by $\boldsymbol{p}$, and all indices i, we have

$$1 - \dot{\varphi}\big(\boldsymbol{p}_i, \big(\boldsymbol{x}_i, f(\boldsymbol{x})\big)_{\boldsymbol{x} \in D}\big) = \dot{\varphi}\big(1 - \boldsymbol{p}_i, \big(1 - \boldsymbol{x}_i, f(\boldsymbol{x})\big)_{\boldsymbol{x} \in D}\big) . \qquad ◀$$

The update of a self-complementary n-Bernoulli-λ-EDA works in the following way: increasing or decreasing a frequency at index i (the probability to sample a 1) is the same as decreasing or increasing, respectively, the complement of the frequency (the probability to sample a 0) in the same manner as the frequency would if all the bits at index i were inverted. This means that the scheme for decreasing follows from the scheme of increasing or vice versa. We can now state the following theorem.

▶ **Theorem 5.9.** Let A be a singularly decomposable n-Bernoulli-λ-EDA. Then the following are equivalent:

1. A is complementation-invariant.
2. A is self-complementary. ◀

Proof. **(1) ⇒ (2).** This direction trivially follows from Definition 5.3 by applying Definition 4.1 and using the complementation such that for all i, $\chi(\boldsymbol{p})_i = 1 - \boldsymbol{p}_i$.

(2) ⇒ (1). The update scheme φ is per component equal to $\dot{\varphi}$ by Definition 4.1 for any i. Thus, we can consider an arbitrary i. For any complementation χ with $\chi(\boldsymbol{p})_i = \boldsymbol{p}_i$, the equation in Definition 5.3 trivially holds. Thus, only the case $\chi(\boldsymbol{p})_i = 1 - \boldsymbol{p}_i$ needs to be considered, which holds by the assumption of point 2. ■

For a singularly decomposable n-Bernoulli-λ-EDA, checking complementation-invariance is very straightforward. We can now give a more specific characterization of unbiasedness.

▶ **Corollary 5.10.** Let A be a decomposable n-Bernoulli-λ-EDA. Then the following are equivalent:

1. A is unbiased.

2. A is singularly decomposable and self-complementary. ◀

Proof. We show that point 2 is equivalent to A being permutation- and complementation-invariant. Theorem 5.5 then yields the equivalence to point 1.

Due to Theorem 5.6, A being singularly decomposable is equivalent to it being permutation-invariant; and due to Theorem 5.9, being self-complementary is equivalent to being complementation-invariant. This finishes the proof. ■

Examples of unbiased decomposable EDAs

We now apply Corollary 5.10 to some of the algorithms mentioned in Section 4.3 to prove their unbiasedness. We discuss the remaining algorithms in the next section.

▶ **Theorem 5.11.** The PBIL, the UMDA, and the λ-MMAS$_{\text{IB}}$ are unbiased. ◀

Proof. We only show that the PBIL is unbiased, since the UMDA and the λ-MMAS$_{\text{IB}}$ are special instances of it. Note that the PBIL is singularly decomposable. Thus, we only show that it is complementation-invariant. Then applying Corollary 5.10 finishes the proof.

In the following, let $\dot{\varphi}$ denote the update function that PBIL uses for its update scheme. Further, recall that we assume a uniform tie-breaking rule, that is, if there is a tie in fitness, the winner is determined uniformly at random. Hence, for any fitness function f and any Hamming automorphism α, if an individual $\boldsymbol{x}$ is the winner of a tie-break using f, $\alpha(\boldsymbol{x})$ will be the winner of a tie-break using $f \circ \alpha^{-1}$ with equal probability.

$$
\begin{aligned}
\dot{\varphi}\Big(1 - \boldsymbol{p}_i, \big(1 - \boldsymbol{x}_i, f(\boldsymbol{x})\big)_{\boldsymbol{x} \in D}\Big) &= (1-\rho)(1-\boldsymbol{p}_i) + \rho \frac{\sum_{k=1}^{\mu}\big(1 - \boldsymbol{x}_i^{(k)}\big)}{\mu} \\
&= 1 - \rho - (1-\rho)\boldsymbol{p}_i + \rho\left(\frac{\mu}{\mu} - \frac{\sum_{k=1}^{\mu}\boldsymbol{x}_i^{(k)}}{\mu}\right) \\
&= 1 - \left((1-\rho)\boldsymbol{p}_i + \rho\frac{\sum_{k=1}^{\mu}\boldsymbol{x}_i^{(k)}}{\mu}\right)
\end{aligned}
$$

$$= 1 - \dot{\varphi}\big(\boldsymbol{p}_i, \big(\boldsymbol{x}_i, f(\boldsymbol{x})\big)_{\boldsymbol{x} \in D}\big) .$$

Note that $1 - \boldsymbol{x}^{(k)}$ and $\boldsymbol{x}^{(k)}$ have the same rank with equal probability, due to uniform tie-breaking. Thus, the equalities shown above hold. ■

5.5 Locally Updating EDAs

Last, we look at locally updating n-Bernoulli-λ-EDAs. In Theorem 5.11, we skipped the cGA. We did so because it is locally updating and, thus, its update schemes can be expressed very concisely. Hence, we want to give a characterization that is even easier to check than the one given in Corollary 5.10. Note that all locally updating n-Bernoulli-λ-EDAs are singularly decomposable by definition and, hence, already permutation-invariant by Theorem 5.6. We focus on a subclass of algorithms that *move symmetrically* and show that this property is equivalent to being complementation-invariant and, thus, unbiased.

▶ **Definition 5.12 (Symmetrical Movement).** Let A be a locally updating n-Bernoulli-λ-EDA. We say that A *moves symmetrically at position* $i \in [n]$ if and only if, for all fitness functions f and all offspring populations D that can be sampled by a frequency vector from $P(A)$, it holds that $P_i(A) = \{1/2\}$ or that

$$\mathrm{flip}\Big(\mathrm{move}\big(\big(\boldsymbol{x}_i, f(\boldsymbol{x})\big)_{\boldsymbol{x} \in D}\big)_i\Big) = \mathrm{move}\big(\big(1 - \boldsymbol{x}_i, f(\boldsymbol{x})\big)_{\boldsymbol{x} \in D}\big)_i , \tag{5.9}$$

where flip: $\{\mathrm{up}, \mathrm{stay}, \mathrm{down}\} \longrightarrow \{\mathrm{up}, \mathrm{stay}, \mathrm{down}\}$ swaps ›up‹ and ›down‹ and maps ›stay‹ to ›stay‹.

If A moves symmetrically at all positions $i \in [n]$, we say that it *moves symmetrically.* ◀

A symmetrically moving n-Bernoulli-λ-EDA moves a frequency into the opposite direction if all the bits for an update at this position are flipped.

▶ **Theorem 5.13.** Let A be a locally updating n-Bernoulli-λ-EDA. Then the following are equivalent:

1. A is complementation-invariant.
2. A moves symmetrically. ◀

Proof. We show that being self-complementary (Definition 5.8; here denoted by (S)) is equivalent to A moving symmetrically (this theorem's point 2). The equivalence to A being complementation-invariant then follows by applying Theorem 5.9. In the following, we always consider any $i \in [n]$.

(S) ⇒ (2). We assume that $P_i(A) \supset \{1/2\}$ and show that equation (5.9) holds. For this, let $v_i = \mathrm{move}\big(\big(\boldsymbol{x}_i, f(\boldsymbol{x})\big)_{\boldsymbol{x} \in D}\big)$, and let $\overline{v}_i = \mathrm{move}\big(\big(1 - \boldsymbol{x}_i, f(\boldsymbol{x})\big)_{\boldsymbol{x} \in D}\big)$. We make a case distinction with respect to v_i.

First, assume that $v_i = \mathrm{up}$. Then, $1 - \dot{\varphi}\big(\boldsymbol{p}_i, \big(\boldsymbol{x}_i, f(\boldsymbol{x})\big)_{\boldsymbol{x} \in D}\big) = 1 - \mathrm{set}(\boldsymbol{p}_i)$ by assumption and the definition of a locally updating n-Bernoulli-λ-EDA. Further, since we assume that A is self-complementary, we have

$$1 - \dot{\varphi}\big(\boldsymbol{p}_i, \big(\boldsymbol{x}_i, f(\boldsymbol{x})\big)_{\boldsymbol{x} \in D}\big) = \dot{\varphi}\big(1 - \boldsymbol{p}_i, \big(1 - \boldsymbol{x}_i, f(\boldsymbol{x})\big)_{\boldsymbol{x} \in D}\big)$$

and, thus, $1 - \mathrm{set}(\boldsymbol{p}_i) = \dot{\varphi}\big(1 - \boldsymbol{p}_i, \big(1 - \boldsymbol{x}_i, f(\boldsymbol{x})\big)_{\boldsymbol{x} \in D}\big)$.

We now make a case distinction with respect to $\overline{v}_i$ in the update on the right-hand side. **For $\overline{v}_i = \mathrm{up}$,** we get $\dot{\varphi}\big(1 - \boldsymbol{p}_i, \big(1 - \boldsymbol{x}_i, f(\boldsymbol{x})\big)_{\boldsymbol{x} \in D}\big) = \mathrm{set}(1 - \boldsymbol{p}_i)$ and, thus, $1 - \mathrm{set}(\boldsymbol{p}_i) = \mathrm{set}(1 - \boldsymbol{p}_i)$, which is only true if $\mathrm{set}(\boldsymbol{p}_i) = \boldsymbol{p}_i$. Now, $1/2$ is not a fixed point of ›set‹ due our assumption of $P_i(A) \supset \{1/2\}$. Note that D as well as its bit-wise complement can be sampled in iteration 0, that is, when $\boldsymbol{p}^{(0)} = \boldsymbol{1/2}$, especially $\boldsymbol{p}_i^{(0)} = 1/2$. Since ›move‹ decides independent of $\boldsymbol{p}_i$ in which direction to move, $\overline{v}_i$ cannot be ›up‹, because that would contradict A being complementation-invariant in iteration 0: $\boldsymbol{p}_i^{(1)}$ would be greater than $1/2$ (since $1/2$ is not a fixed point) but so would $\widetilde{\boldsymbol{p}}_i^{(1)}$ (the update of $\boldsymbol{p}^{(0)}$ with respect to the bit-wise complement of D). However, $\widetilde{\boldsymbol{p}}_i^{(0)}$ had to be smaller than $1/2$.

For $\overline{v}_i = \mathrm{stay}$, we get $1 - \mathrm{set}(\boldsymbol{p}_i) = 1 - \boldsymbol{p}_i$, which, again, can only hold if $\mathrm{set}(\boldsymbol{p}_i) = \boldsymbol{p}_i$. We can argue analogously to before, the only difference being that $\widetilde{\boldsymbol{p}}_i^{(1)}$ would be $1/2$, whereas $\boldsymbol{p}_i^{(1)}$ would, again, be greater than $1/2$, which contradicts A being complementation-invariant.

For $\overline{v}_i = \mathrm{down}$, we get $1 - \mathrm{set}(\boldsymbol{p}_i) = 1 - \mathrm{set}\big(1 - (1 - \boldsymbol{p}_i)\big)$, which is always true. Hence, if $v_i = \mathrm{up}$, then $\overline{v}_i = \mathrm{down}$.

The other two cases with respect to v_i can be done analogously and yield that $\overline{v}_i = \mathrm{stay}$ if $v_i = \mathrm{stay}$, and that $\overline{v}_i = \mathrm{up}$ if $v_i = \mathrm{down}$ (always assuming that $P_i(A) \supset \{1/2\}$). All in all, this results in $\mathrm{flip}(v_i) = \overline{v}_i$.

(2) ⇒ (S). If $P_i(A) = \{1/2\}$, A is trivially complementation-invariant for that position, since its only frequency for this position is $1/2$, independent of any D sampled.

If $P_i(A) \supset \{1/2\}$, we show that

$$1 - \dot{\varphi}\big(\boldsymbol{p}_i, \big(\boldsymbol{x}_i, f(\boldsymbol{x})\big)_{\boldsymbol{x} \in D}\big) = \dot{\varphi}\big(1 - \boldsymbol{p}_i, \big(1 - \boldsymbol{x}_i, f(\boldsymbol{x})\big)_{\boldsymbol{x} \in D}\big) .$$

We start with the former and make, as before, a case distinction with respect to v_i. Due to A moving symmetrically, we have $\mathrm{flip}(v_i) = \overline{v}_i$.

First, assume that $v_i = \mathrm{up}$. We then get

$$\begin{aligned} 1 - \dot{\varphi}\big(\boldsymbol{p}_i, \big(\boldsymbol{x}_i, f(\boldsymbol{x})\big)_{\boldsymbol{x} \in D}\big) &= 1 - \mathrm{set}(\boldsymbol{p}_i) \\ &= 1 - \mathrm{set}\big(1 - (1 - \boldsymbol{p}_i)\big) \\ &= \dot{\varphi}\big(1 - \boldsymbol{p}_i, \big(1 - \boldsymbol{x}_i, f(\boldsymbol{x})\big)_{\boldsymbol{x} \in D}\big) , \end{aligned}$$

where the first and third equality follow from the definition of a locally updating n-Bernoulli-λ-EDA's update scheme, using that $v_i = \mathrm{up}$ (first equality) and that $\overline{v}_i = \mathrm{down}$ (third equality). The other two cases can be done analogously and yield the same result. This concludes the proof. ■

Using Theorem 5.13, we get the following Corollary.

▶ **Corollary 5.14.** Let A be a locally updating n-Bernoulli-λ-EDA. Then the following are equivalent:

1. A is unbiased.
2. A moves symmetrically. ◀

Proof. We show that point 2 is equivalent to A being unbiased. Note that we assume A to be singularly decomposable, as it is locally updating.

Due to Theorem 5.13, A moving symmetrically is equivalent to it being complementation-invariant, which is equivalent to it being self-complementary (Theorem 5.9); and due to Corollary 5.10, this is equivalent to A being unbiased. ■

For a locally updating n-Bernoulli-λ-EDA, only the ›move‹ function decides whether the algorithm is unbiased or not. The remaining update via ›set‹ is already set up to perform an unbiased update. This makes checking for unbiasedness very easy.

▶ **Theorem 5.15.** The cGA is unbiased. ◀

Proof. We make use of Corollary 5.14 and show that the cGA moves symmetrically. Let $i \in [n]$ be an arbitrary index. As before, since we assume uniform tie-breaking, we can assume for all $\boldsymbol{x} \in D$ that the ranks of $\boldsymbol{x}$ and $1 - \boldsymbol{x}$ are the same with equal probability (as they have the same fitness $f(\boldsymbol{x})$, according to equation (5.9)). Further, we use the same notation for the two ›move‹ values as in the proof of Theorem 5.13: v_i and $\overline{v}_i$.

For the cGA, $\mathrm{move}\big((\boldsymbol{x}_i, f(\boldsymbol{x}))_{\boldsymbol{x} \in D}\big) = v_i$ depends on the relation of $\boldsymbol{x}_i^{(1)}$ and $\boldsymbol{x}_i^{(2)}$. If $\boldsymbol{x}_i^{(1)} = \boldsymbol{x}_i^{(2)}$, then $1 - \boldsymbol{x}_i^{(1)} = 1 - \boldsymbol{x}_i^{(2)}$, as well. This means that $v_i = \text{stay}$ and $\overline{v}_i = \text{stay}$. Thus, $\mathrm{flip}(v_i) = \overline{v}_i$.

If $\boldsymbol{x}_i^{(1)} > \boldsymbol{x}_i^{(2)}$, then $1 - \boldsymbol{x}_i^{(1)} < 1 - \boldsymbol{x}_i^{(2)}$. This means that $v_i = \text{up}$ and $\overline{v}_i = \text{down}$. Hence, $\mathrm{flip}(v_i) = \overline{v}_i$. The case $\boldsymbol{x}_i^{(1)} < \boldsymbol{x}_i^{(2)}$ follows completely analogously. Overall, the cGA moves symmetrically and is, thus, unbiased. ■

Since we only need to check the ›move‹ function of a locally updating n-Bernoulli-λ-EDA, Theorem 5.15 can be generalized: every locally updating n-Bernoulli-λ-EDA with a ›move‹ function as in the cGA is unbiased. Thus, the framework of locally updating n-Bernoulli-λ-EDAs provides a powerful tool to easily come up with unbiased n-Bernoulli-λ-EDAs: a single (symmetrically moving) ›move‹ function can be used with arbitrary ›set‹ functions and will always result in an unbiased algorithm.

5.6 Conclusions

We analyzed the broad class of n-Bernoulli-λ-EDAs with respect to unbiasedness. That is, we proved when those algorithms behave the same, regardless of the problem encoding, which is a desirable property in true black-box optimization, as the optimization algorithm has no knowledge about the representation of the problem. We showed how this applies to certain algorithms by providing examples. In order to account for the simpler update schemes of those example algorithms, we provided conciser characterizations of certain subclasses, which can be verified more easily. Our results can especially be viewed as guidelines on how to create an update scheme when one wants to create an unbiased algorithm – or when not.

The results of this chapter allow us to generalize our run time results in the following chapters to the OneMax function class or the LeadingOnes function class when we consider one of these functions, respectively. Further, Corollary 5.14 allows us to easily prove the unbiasedness of locally updating n-Bernoulli-λ-EDAs when the ›move‹ function is that of an algorithm of which we already know that it is unbiased. In Sections 6.4.1 and 8.2.1, we use this in order to easily argue that the considered algorithms are unbiased.

For future research, the subclass of locally updating n-Bernoulli-λ-EDAs seems reasonably limited in order to derive a lower bound in the unbiased black-box complexity setting. And we conjecture that the lower bound is $\Omega\big(n \log(n)\big)$, motivated by the results on the cGA and the 2-MMAS$_{\mathrm{IB}}$ by Sudholt and Witt [SW16a] and insights from Chapter 9 like Lemma 9.7.

6 n-Bernoulli-λ-EDAs Cannot be Balanced and Stable

This chapter is based on joint work with Tobias Friedrich and Timo Kötzing [FKK16]. The results from that paper have been heavily reworked for this chapter. The definition of stable *was changed in order to better reflect the idea of the concept. Consequently, all of the results have been rephrased with respect to the new definition. Further, the results have been generalized and follow the concept of drift for martingales (described in Section 3.6), which was not available when the results were originally published. Last, in the conference version, the main result applied a negative-drift theorem incorrectly, as it assumed the drift to be negative everywhere. This has now been corrected by using an appropriate negative-drift theorem where the drift does not need to be negative everywhere.*

Parts of the introduction are from joint work with Benjamin Doerr [DK18a], which is based on a joint conference publication [DK18b].

In this chapter, we define the two properties of *balanced* and *stable* for n-Bernoulli-λ-EDAs. A balanced EDA does not change its frequency vector in expectation when there is no signal from the fitness function, and the frequencies of a stable EDA stay clsoe to $1/2$ in such a case. We prove that these concepts are mutually exclusive (Theorem 6.11), show that the algorithms mentioned in Section 4.3 are all balanced (Theorems 6.8 and 6.10 and Corollary 6.9), and we introduce a stable n-Bernoulli-λ-EDA that is able to optimize LeadingOnes with high probability in $\mathrm{O}\big(n \log(n)\big)$ iterations in expectation (Section 6.4).

6.1 Introduction

EDAs optimize a function by evolving a probabilistic model of the solution space. In an iterative fashion, an EDA uses its model to generate samples and then updates it with respect to observations made from these samples. An algorithm-specific parameter determines how strong the changes to the model in each iteration are.

In order for an EDA to succeed in optimization, it is important that the probabilistic model is changed over time in a way that better solutions are sampled more frequently. However, due to the randomness in sampling (also called *genetic drift*), the model should not be changed too drastically in a single iteration in order to prevent wrong updates from having a long-lasting impact.

Recent theoretical results for EDAs have clearly demonstrated that this trade-off between convergence speed and accumulation of erratic updates can be delicate and non-trivial to understand. Among the most relevant works, Sudholt and Witt [SW16a] and our results in Chapter 9 prove lower bounds of the expected run times of three common n-Bernoulli-λ-EDAs on the benchmark function OneMax. In simple words, these bounds show that if the parameter for updating the model is too large, the model converges too quickly and very likely to a wrong model. In consequence, it then takes a long time to find the optimum (usually by first reverting to a better fitting model). On the other hand, if the parameter is too small, then the model converges to the correct model, but it does so slowly. More formally, Sudholt and Witt [SW16a] prove a lower bound of $\Omega\big(K\sqrt{n} + n\log(n)\big)$ for the 2-MMAS$_{\mathrm{IB}}$ and the cGA, where $1/K$ is the step size of the algorithm, and in Chapter 9, we prove a lower bound of $\Omega\big(\lambda\sqrt{n} + n\log(n)\big)$ for the UMDA, where λ is the population size of the algorithm. These results show that choosing the parameter with a value of $\omega\big(\sqrt{n}\log(n)\big)$ has no benefit. Further, it has been recently shown by Lengler et al. [LSW18] that the run time of the cGA on OneMax is $\Omega\big(K^{1/3}n + n\log(n)\big)$ for $K \in O\big(\sqrt{n}/\big(\log(n)\log(\log(n))\big)\big)$. Together with the results from Sudholt and Witt [SW16a], this suggests a bimodal behavior in the run time with respect to the parameter K in the regime of $K \in \Omega\big(\log(n)\big) \cap O\big(\sqrt{n}\log(n)\big)$, showing that the run time is sensitive to the parameter choice.

In this chapter, we analyze when the parameter choice of an n-Bernoulli-λ-EDA likely results in the convergence to a non-optimal frequency vector. By *convergence* we mean that a frequency gets subconstantly close to 0 or 1, which then allows to sample bits at the respective position more consistently. We consider a setting with a constant fitness function such that there is no preference for an individual over another. That is, we analyze the behavior of an algorithm with respect to genetic drift, and convergence to any frequency vector is considered bad, as the fitness function does not indicate that 1s are better than 0s or vice versa.

In Section 6.3, we define two important properties in this setting. We say that an n-Bernoulli-λ-EDA A is *balanced* if it does not change its frequency vector in a single iteration in expectation (Definition 6.3). And we say that A is t-*stable* if its frequencies stay close to $1/2$ during t many iterations (Definition 6.4).

Being stable for a large number of iterations is a desirable property of an n-Bernoulli-λ-EDA, since it guarantees with high probability that the algorithm will not converge during that time to a wrong model when there is no input from the fitness function. The property of being balanced seems to be related to this concept, as it guarantees that the frequency vector does not change in expectation in such a scenario. Hence, it is reasonable to ask whether an n-Bernoulli-λ-EDA with both properties exists or if being balanced may even imply being stable. The relevance of this question is increased by the fact that we prove all of the commonly analyzed n-Bernoulli-λ-EDAs (see Section 4.3) to be balanced. Unfortunately, we give a general result that, informally speaking, states that a balanced n-Bernoulli-λ-EDA cannot be stable (Theorem 6.11).

In Section 6.4, we then apply the concept of stability to the fitness function LeadingOnes, which does not provide a signal for a long period of time for certain positions: the bits at the end of a bit string will only yield a preference for 1s if all prior bits in the string are 1s as well. This is very unlikely to happen during early iterations of an n-Bernoulli-λ-EDA, since all frequencies start at a constant value of $1/2$. Since all commonly analyzed n-Bernoulli-λ-EDAs are balanced and thus not stable, it is likely that their step size has to be small in order to prevent convergence to an incorrect frequency vector, resulting in a slow optimization time. Hence, we introduce the *scGA* (short for *stable cGA*), which is a variant of the cGA that is stable. This allows to choose a large step size and optimize with high probability LeadingOnes within $\mathrm{O}\big(n \log(n)\big)$ iterations in expectation (Corollary 6.16). This is a run time that only few EAs can match (see Table 10.1). However, we would like to mention that we prove in Chapter 10 that the scGA optimizes OneMax inefficiently in return (Theorem 10.9).

6.2 Preliminaries

In the following, we define the concepts of *balanced* and *stable* for n-Bernoulli-λ-EDAs. Afterward, we mention theorems that we use in order to analyze algorithms that have one of these properties.

A balanced n-Bernoulli-λ-EDA does not change its frequencies in expectation when there is no signal from the fitness function. The following definition formalizes what we mean by *no signal.*

▶ **Definition 6.1 (No Signal).** Let $f\colon \{0,1\}^n \to \mathbf{R}$ denote a fitness function, and let $S \subseteq [n]$ denote a set of position. We say that f *provides no signal at positions in S* if and only if, for all $\boldsymbol{x}, \boldsymbol{y} \in \{0,1\}^n$ such that $\boldsymbol{x}$ and $\boldsymbol{y}$ are the same at all positions not in S (and may differ at positions in S), it holds that

$$f(\boldsymbol{x}) = f(\boldsymbol{y}) \ .$$

We say that f *provides no signal* if and only if f provides no signal in $[n]$. ◀

Note that the requirement in Definition 6.1 that $\boldsymbol{x}$ and $\boldsymbol{y}$ are from $\{0,1\}^n$ is rather strict and can be relaxed such that it may also be applied to subsets of $\{0,1\}^n$, such as an offspring population of an n-Bernoulli-λ-EDA. This can be applied, for example, in the case of LeadingOnes (equation (2.2)): early in the optimization process, positions toward the end of a bit string will provide no signal for all practical purposes but will become meaningful later in the optimization.

When a fitness function does not provide a signal at some positions, it is indifferent about the respective bit values. We are interested in how an n-Bernoulli-λ-EDA behaves in such a situation.

▶ **Definition 6.2 (f-Independence).** Let f be a fitness function that provides no signal, and let A be an n-Bernoulli-λ-EDA that is optimizing f. We then say that A is *f-independent.* ◀

Definition 6.2 provides a convenient term for considering a setting where an n-Bernoulli-λ-EDA performs an update when the fitness function gives no indication of which individuals are preferable. Note that this definition (as well as the following definitions) can be also phrased with respect to a subset of indices (as done in Definition 6.1). However, for the sake of brevity, we only consider all positions at once.

We may say that an n-Bernoulli-λ-EDA A is f-independent without naming a fitness function. If we do so, we assume that we are given a function f such that A is f-independent.

We now define when an n-Bernoulli-λ-EDA is balanced.

▶ **Definition 6.3 (Balanced n-Bernoulli-λ-EDA).** Let A be an f-independent n-Bernoulli-λ-EDA. We say that A is *unbiased* if and only if, for all $t \in \mathbf{N}$, it holds that

$$\mathrm{E}\left[\boldsymbol{p}^{(t+1)} \,\middle|\, \boldsymbol{p}^{(t)}\right] = \boldsymbol{p}^{(t)} \ .$$ ◀

A balanced n-Bernoulli-λ-EDA does not change its frequency vector in expectation when performing an update, given a fitness function providing no signal. Hence, the frequencies describe martingales in such a case.

The following concept formalizes that an update of an f-independent n-Bernoulli-λ-EDA is concentrated around $1/2$, which we call *stable.*

▶ **Definition 6.4 (t-Stable n-Bernoulli-λ-EDA).** Let A be an f-independent n-Bernoulli-λ-EDA, and let $t \in \mathbf{N}$. We say that A is t*-stable* if and only if for all values $d \in \mathrm{o}(1)$ there is a constant $c > 0$ such that

$$\Pr\left[\exists i \in [n], t' \leq t \colon \boldsymbol{p}_i^{(t')} \notin [d, 1-d]\right] \leq \frac{1}{n^c} .$$

If and only if A is t-stable for all values of $t \in \mathbf{N}$ that are a polynomial in n, we say that A is *polynomially stable.* Further, if and only if A is t-stable for all $t \in \mathbf{N}$, we say that A is *stable.* ◀

The frequencies of a t-stable n-Bernoulli-λ-EDA take with high probability only constant values for t iterations, given a fitness function providing no signal. Conversely, an n-Bernoulli-λ-EDA is not t-stable when there is a time point in the interval $[0..t]$ such that a frequency gets subconstantly close to 0 or 1 with a probability greater than any polynomially low probability.

We now provide theorems that we use in our analyzes when considering balanced or stable n-Bernoulli-λ-EDAs. In order to determine whether an n-Bernoulli-λ-EDA is stable, we need to know whether a frequency can get subconstantly close to the borders (0 or 1) within a given time span. For this, we are going to use drift theory, as introduced in Chapter 3. Our main tool will be the transformation given in Theorem 3.18. As discussed in Section 3.6, we can then bound first-hitting times by considering the variance of the martingale. In order to calculate variances easily, we use the following theorem.

▶ **Theorem 6.5 ([GS01b, Chapter 3.3, Theorem 11]).** Let X and Y be two independent random variables. Then,

- for all $a \in \mathbf{R}$, it holds that $\mathrm{Var}[aX] = a^2\mathrm{Var}[X]$, and
- $\mathrm{Var}[X + Y] = \mathrm{Var}[X] + \mathrm{Var}[Y]$. ◀

Similar to the linearity of the expected value (Theorem 2.5), Theorem 6.5 allows us to calculate the variance of a random variable by decomposing it into simpler ones.

We bound the first-hitting times of martingales using the multiplicative drift theorem (Corollary 3.15) after applying Theorem 3.18, which results in the following corollary. It bounds the first-hitting time of a martingale to get within a distance of at least d to two bounds α and β, which are free to choose.

▶ **Corollary 6.6 (Martingale Upper Multiplicative Drift, Below Target).** Let $(X_t)_{t\in\mathbf{N}}$ be random variables over $[\alpha, \beta] \subset \mathbf{R}$ adapted to a filtration $(\mathcal{F}_t)_{t\in\mathbf{N}}$, let $d \in \big(0, (\beta-\alpha)/2\big]$, and let $T = \inf\{t \in \mathbf{N} \mid X_t \notin [\alpha+d, \beta-d]\}$. Furthermore, suppose that

(a) there is some value $\delta > 0$ such that, for all $t < T$, it holds that

$$\mathrm{Var}[X_{t+1} \mid \mathcal{F}_t] \geq \delta(X_t - \alpha)(\beta - X_t)\text{ , that}$$

(b) $X_0 \in [\alpha+d, \beta-d]$, and that,

(c) for all $t < T$, it holds that $\mathrm{E}[X_{t+1} \mid \mathcal{F}_t] = X_t$.

Then

$$\mathrm{E}[T \mid \mathcal{F}_0] \leq \frac{1 + \ln\left(\frac{(X_0-\alpha)(\beta-X_0)}{d(\beta-\alpha-d)}\right)}{\delta} \text{ .}$$ ◀

Proof. Let $(Y_t)_{t\in\mathbf{N}}$ denote the transformed process of X according to Theorem 3.18. We want to apply Corollary 3.15 to Y. Hence, we have to check that the conditions of the theorem are satisfied.

Since X is a martingale over $[\alpha, \beta]$, it follows, for all $t \in \mathbf{N}$, that $Y_t \geq 0$. Further, since, for all $t \in \mathbf{N}$, the random variable Y_t is a concave function with the roots α and β and since α and β are at least d apart, it holds that $X_t \notin [\alpha+d, \beta-d]$ if and only if $Y_t < \big((\alpha+d)-\alpha\big)\big(\beta-(\alpha+d)\big) = d(\beta-\alpha-d) =: y_{\min}$. Thus, T denotes the first point in time such that $Y_t < y_{\min}$. Since we can bound the drift of Y via the variances of X according to Theorem 3.18, we can apply Corollary 3.15. This concludes the proof. ■

6.3 Balanced Versus Stable

In this section, we show that the concepts of balanced and stable are mutually exclusive (Theorem 6.11). However, note that in order to be not stable, an n-Bernoulli-λ-EDA needs to be able to take frequency values that are subconstantly close to one of the borders (0 or 1). Hence, we only consider such algorithms. An n-Bernoulli-λ-EDA whose frequencies can only take constant values are of no particular interest for optimization anyway, as the probability of sampling a specific individual is always exponentially small in the problem size then, resulting in an exponential expected run time (assuming a polynomial number of optima).

Further note that adding a margin b to a balanced n-Bernoulli-λ-EDA that can reach 0 or 1 makes it become not balanced, since a frequency at a value that could reach (without loss of generality) the value 0 before now gets cut off short, leading in expectation to a greater frequency after the update. Hence, we only consider n-Bernoulli-λ-EDAs without a margin. However, we would like to mention that all of the results in this section follow analogously for an interval of frequency values where adding a margin has no impact on the update.

We start off by showing that all of the commonly analyzed n-Bernoulli-λ-EDAs are balanced.

▶ **Theorem 6.7.** The PBIL, the UMDA, the λ-MMAS$_{\text{IB}}$, and the cGA are balanced. ◀

Proof. Assume that the algorithms are f-independent. We show for every position that the frequency does not change in expectation. Let $i \in [n]$ be an index, and let $\boldsymbol{p}'_i$ be the value of $\boldsymbol{p}_i$ after an update. Let $\boldsymbol{x}^{(k)}$ denote the k-th best individual in the offspring population D of any of the n-Bernoulli-λ-EDAs listed below, as defined in the beginning of Section 4.3.

Since the algorithms are f-independent, $\boldsymbol{x}_i^{(k)}$ is 1 with probability $\boldsymbol{p}_i$ and 0 with probability $1 - \boldsymbol{p}_i$.

PBIL: Using the linearity of expectation, we get

$$\begin{aligned}
\mathrm{E}\left[\boldsymbol{p}'_i \mid \boldsymbol{p}_i\right] &= (1-\rho)\boldsymbol{p}_i + \rho \frac{\sum_{k=1}^{\mu} \mathrm{E}\left[\boldsymbol{x}_i^{(k)} \mid \boldsymbol{p}_i\right]}{\mu} \\
&= (1-\rho)\boldsymbol{p}_i + \rho \frac{\sum_{k=1}^{\mu} \boldsymbol{p}_i}{\mu} \\
&= (1-\rho)\boldsymbol{p}_i + \rho\boldsymbol{p}_i \\
&= \boldsymbol{p}_i \ .
\end{aligned}$$

UMDA/$\boldsymbol{\lambda}$-MMAS$_{\text{IB}}$: Both algorithms are balanced, since they are special cases of PBIL.

cGA: Again, using the linearity of expectation, we get

$$\begin{aligned}\mathrm{E}\big[\boldsymbol{p}_i' \,\big|\, \boldsymbol{p}_i\big] &= \Big(\boldsymbol{p}_i - \frac{1}{K}\Big)\boldsymbol{p}_i(1-\boldsymbol{p}_i) + \boldsymbol{p}_i\big(1-2\boldsymbol{p}_i(1-\boldsymbol{p}_i)\big) + \Big(\boldsymbol{p}_i + \frac{1}{K}\Big)\boldsymbol{p}_i(1-\boldsymbol{p}_i)\\ &= 2\boldsymbol{p}_i^2(1-\boldsymbol{p}_i) + \boldsymbol{p}_i - 2\boldsymbol{p}_i^2(1-\boldsymbol{p}_i)\\ &= \boldsymbol{p}_i\ .\end{aligned}$$

■

We now show that all of the above algorithms are not polynomially stable and, thus, not stable. We start with the PBIL and then discuss the other algorithms. The proof technique will be the same for all of the proofs: we use Corollary 6.6 and bound the expected first-hitting time of a frequency going below a subconstant value. Afterward, we apply Markov's inequality (Theorem 2.18) to the first-hitting time in order to argue that the algorithm is not polynomially stable.

▶ **Theorem 6.8.** For all $d \in \mathrm{o}(1)$ there is a constant $c > 0$ such that the PBIL is not $(c\mu/\rho^2)\ln(1/d)$-stable. ◀

Proof. Assume that the PBIL is f-independent and consider a position $i \in [n]$. Let $d \in \mathrm{o}(1)$, and let $T = \inf\{t \in \mathbf{N} \mid \boldsymbol{p}_i^{(t)} \notin [d, 1-d]\}$. Note that $\boldsymbol{p}_i$ is a martingale over $[0,1]$, since the PBIL is f-independent. We now bound the expected value of T by applying Corollary 6.6. In order to do so, we need to bound the conditional variance of $\boldsymbol{p}_i$.

Let $t < T$, and let $\boldsymbol{x}^{(k)}$ denote the individual of rank k from iteration t. Note that the bit values at position i follow a Bernoulli distribution with success probability $\boldsymbol{p}_i^{(t)}$. Further note that conditional on $\boldsymbol{p}_i^{(t)}$ and due to $\boldsymbol{p}_i^{(t)} \notin \{0,1\}$ because of $t < T$, the random variables $\boldsymbol{p}_i^{(t)}$ and $\boldsymbol{x}^{(k)}$ (for all $k \in [\lambda]$) are independent. Additionally, all individuals are sampled independently from one another by the definition of an n-Bernoulli-λ-EDA. Hence, using Theorem 6.5, we get

$$\begin{aligned}\mathrm{Var}\Big[\boldsymbol{p}_i^{(t+1)} \,\Big|\, \boldsymbol{p}_i^{(t)}\Big] &= \mathrm{Var}\Bigg[(1-\rho)\boldsymbol{p}_i^{(t)} + \rho\frac{\sum_{k=1}^{\mu}\boldsymbol{x}_i^{(k)}}{\mu} \,\Bigg|\, \boldsymbol{p}_i^{(t)}\Bigg]\\ &= (1-\rho)^2\overbrace{\mathrm{Var}\Big[\boldsymbol{p}_i^{(t)} \,\Big|\, \boldsymbol{p}_i^{(t)}\Big]}^{=0} + \rho^2\frac{\sum_{k=1}^{\mu}\mathrm{Var}\Big[\boldsymbol{x}_i^{(k)} \,\Big|\, \boldsymbol{p}_i^{(t)}\Big]}{\mu^2}\\ &= \frac{\rho^2}{\mu}\cdot\boldsymbol{p}_i^{(t)}\big(1-\boldsymbol{p}_i^{(t)}\big)\ .\end{aligned}$$

By now applying Corollary 6.6 with $\delta = \rho^2/\mu$ and the law of total expectation and by noting that $\boldsymbol{p}_i^{(0)} = 1/2$, we get

$$\begin{aligned}\mathrm{E}[T] &\leq \frac{1 + \ln\left(\frac{1}{4d(1-d)}\right)}{\frac{\rho^2}{\mu}} \\ &\in \mathrm{O}\left(\frac{\mu}{\rho^2}\log\left(\frac{1}{d}\right)\right) .\end{aligned}$$

Finally, by applying Markov's inequality (Theorem 2.18), we see that

$$\Pr\big[T < 2\mathrm{E}[T]\big] \geq \frac{1}{2} .$$

Thus, the probability of $\boldsymbol{p}_i$ getting subconstantly close to a border within polynomial time is at least $1/2$. This concludes the proof. ■

If $1/\rho$ and μ are bounded from above by polynomials and if d is chosen such that $1/d \leq 2^{\mathrm{poly}(n)}$, then Theorem 6.8 says that the PBIL is not polynomially stable.

Note that the t-stable property only considers the number of iterations in which frequencies only take constant values – the number of fitness function evaluations does not matter. This is why λ does not occur in the statement that the PBIL is not polynomially stable. Hence, if one is interested in the expected number of fitness function evaluations until a frequency gets subconstantly close to a border, the same method as in the proof above can be used but a factor of λ needs to be added, since an n-Bernoulli-λ-EDA samples λ individuals each iteration. Hence, in order to guarantee that a frequency of the PBIL gets subconstantly close to a border within polynomially many fitness function evaluations, λ needs to be upper-bounded by a polynomial too.

Since the UMDA and the λ-MMAS$_{\mathrm{IB}}$ are special cases of the PBIL, we get the following corollary.

▶ **Corollary 6.9.** For all $d \in o(1)$ there is a constant $c > 0$ such that the UMDA is not $c\mu\ln(1/d)$-stable and such that the λ-MMAS$_{\mathrm{IB}}$ is not $(c/\rho^2)\ln(1/d)$-stable. ◀

We now consider the cGA and show that it is not stable. Aside from a different variance than that of the PBIL, the same arguments as in the proof of Theorem 6.8 apply.

▶ **Theorem 6.10.** For all $d \in \mathrm{o}(1)$ there is a constant $c > 0$ such that the cGA is not $cK^2 \ln(1/d)$-stable. ◀

Proof. We follow the same proof strategy as in the proof of Theorem 6.8. Hence, we use the same notation.

When bounding the conditional variance of $\boldsymbol{p}_i^{(t+1)}$, we again make use of Theorem 6.5. Note that this theorem implies for a random variable X that $\mathrm{Var}[-X] = \mathrm{Var}[X]$. We get

$$\begin{aligned}\mathrm{Var}\left[\boldsymbol{p}_i^{(t+1)} \,\middle|\, \boldsymbol{p}_i^{(t)}\right] &= \mathrm{Var}\left[\boldsymbol{p}_i^{(t)} + (\boldsymbol{x}_i^{(1)} - \boldsymbol{x}_i^{(2)})\frac{1}{K} \,\middle|\, \boldsymbol{p}_i^{(t)}\right]\\ &= \overbrace{\mathrm{Var}\left[\boldsymbol{p}_i^{(t)} \,\middle|\, \boldsymbol{p}_i^{(t)}\right]}^{=0} + \frac{1}{K^2}\left(\mathrm{Var}\left[\boldsymbol{x}_i^{(1)} \,\middle|\, \boldsymbol{p}_i^{(t)}\right] + \mathrm{Var}\left[\boldsymbol{x}_i^{(2)} \,\middle|\, \boldsymbol{p}_i^{(t)}\right]\right)\\ &= \frac{2}{K^2} \cdot \boldsymbol{p}_i^{(t)}\left(1 - \boldsymbol{p}_i^{(t)}\right) .\end{aligned}$$

By applying Corollary 6.6 with $\delta = 2/K^2$, we get

$$\mathrm{E}[T] \in \mathrm{O}\left(K^2 \log\left(\frac{1}{d}\right)\right) .$$

Using Markov's inequality in the same fashion as in the proof of Theorem 6.8 concludes the proof. ■

When looking at the first-hitting time of the cGA of $\mathrm{O}\big(K^2 \log(1/d)\big)$ and at that of the λ-MMAS$_{\mathrm{IB}}$ of $\mathrm{O}\big((1/\rho)^2 \log(1/d)\big)$ of a frequency getting up to d close to a border, we see that the algorithms behave asymptotically identically when choosing $K = 1/\rho$. However, one main difference is that the cGA can actually reach a border within finite time, whereas the λ-MMAS$_{\mathrm{IB}}$ is not able to do so. Hence, it is important that we only consider getting close to a border.

The first-hitting time of the UMDA of $\mathrm{O}\big(\mu \log(1/d)\big)$ is by factor of μ smaller.[16] However, when considering the number of fitness function evaluations and

[16] When comparing μ to the algorithm-specific parameters K and $1/\rho$ of, respectively, the cGA and the λ-MMAS$_{\mathrm{IB}}$.

assuming that $\lambda \in \Theta(\mu)$, the first-hitting time is in the same order of magnitude of that of the cGA and the 2-MMAS$_{\text{IB}}$. Related to that, Lengler et al. [LSW18] discuss other similarities of the cGA and the UMDA, and the results by Sudholt and Witt [SW16a] and the results from Chapter 9 also draw a common picture for these three algorithms.

We would like to mention that using Markov's inequality in the proofs of Theorems 6.8 and 6.10 is sufficient, but stronger concentration bounds can be achieved by using the multiplicative drift theorem by Doerr and Goldberg [DG13], which also provides a tail bound. Similar calculations as in the proofs above then yield that the first-hitting times are not exceeded by a constant factor with high probability.

The proofs of Theorems 6.8 and 6.10 make use of Theorem 3.18, which can be used since the PBIL and the cGA are balanced. We now show that this idea generalizes to all n-Bernoulli-λ-EDAs without margin that can (in the limit) reach the borders.

▶ **Theorem 6.11.** Let A be a balanced n-Bernoulli-λ-EDA. If there is a position $i \in [n]$ such that there exists a $d \in \mathrm{o}(1)$ such that

$$\inf\left\{\mathrm{Var}\left[\boldsymbol{p}_i^{(t+1)} + \mathbf{1}\{\boldsymbol{p}_i^{(t)} \notin [d, 1-d]\} \,\middle|\, \boldsymbol{p}_i^{(t)}\right] \,\middle|\, t \in \mathbf{N}\right\} > 0\,, \tag{6.1}$$

then A is not stable. ◀

Proof. We follow again the same proof strategy as in the proof of Theorem 6.8. Hence, let $T = \inf\{t \in \mathbf{N} \mid \boldsymbol{p}_i^{(t)} \notin [d, 1-d]\}$. Since, for all $t < T$, we have that $\boldsymbol{p}_i^{(t)} \in [d, 1-d]$, it follows from equation (6.1) that

$$\inf\left\{\mathrm{Var}\left[\boldsymbol{p}_i^{(t+1)} \,\middle|\, \boldsymbol{p}_i^{(t)}\right] \,\middle|\, t < T\right\} =: \delta > 0\,.$$

Since $\boldsymbol{p}_i$ is a martingale, due to A being balanced, we can apply Corollary 3.19 with δ as defined above and get

$$\mathrm{E}[T] \le \frac{1}{4\delta}\,.$$

Using Markov's inequality as at the end of the proof of Theorem 6.8 concludes the proof. ■

Intuitively, Theorem 6.11 says that a balanced n-Bernoulli-λ-EDA is not stable if there is a frequency that has a positive variance while not being in distance d to the borders. In this case, the drift of the variance then pushes the frequency below d to the borders, resulting in the EDA not being stable.

Note that the random variable $\mathbf{1}\{\boldsymbol{p}_i^{(t)} \notin [d, 1-d]\}$ is not being used in the proof of Theorem 6.11. Its only purpose is to make sure that variances where $\boldsymbol{p}_i^{(t)}$ is not in $[d, 1-d]$ account for a fixed value that allows the infimum to be greater than 0. This is because we do not care about the variance if $\boldsymbol{p}_i^{(t)}$ leaves the interval $[d, 1-d]$. By adding $\mathbf{1}\{\boldsymbol{p}_i^{(t)} \notin [d, 1-d]\}$, we make sure that variances in the case of $\boldsymbol{p}_i^{(t)} \notin [d, 1-d]$ do not change the applicability of the theorem.

Further note that equation (6.1) implies that the variance cannot degenerate to 0 in the limit, with respect to the number of iterations, while the frequency is not close to the borders. This is a stronger statement than requiring that the variance is always positive, as the following example shows. Consider a deterministic update scheme that updates a frequency from $1/2$ to $1/4$ and then always to the arithmetic mean of the last two values. Then the variance will be positive for all time points while being sufficiently far away from the borders, but it will reach 0 in the limit. Equation (6.1) makes sure that no (potentially infinite) sequence of updates degenerates into such a case when the frequency is not close to the borders.[17]

We would like to mention that such pathological cases as described above do not occur in commonly analyzed EDAs and, likely, not in natural n-Bernoulli-λ-EDAs, as this would mean that a frequency can converge to other values than 0 or 1, which seems counterintuitive.

6.4 Solving LeadingOnes Efficiently

Considering positions with no signal is of particular interest when analyzing an n-Bernoulli-λ-EDA optimizing LeadingOnes. Note that when we speak of the *run time* of an algorithm, we refer to the expected number of fitness function evaluations until the optimum is sampled for the first time. Further, when

[17] Equation (6.1) is actually more restrictive than necessary, as we only need the infimum to be positive while not having left the interval $[d, 1-d]$ yet. That is, if the update scheme has degenerate cases that can only be reached *after* leaving the interval $[d, 1-d]$, Theorem 6.11 would not apply, but the same proof strategy would still yield that the respective n-Bernoulli-λ-EDA is not stable.

considering unbiased algorithms (as we do here), all results hold for the entire LeadingOnes function class.

For LeadingOnes, if the maximum number of leading 1s over all individuals in an offspring population D is j, all positions $i > j+1$ do not provide a signal (in a relaxed sense) in that iteration because the respective bits $\boldsymbol{x}_i$ of each individual in D do not contribute to the fitness.

We now look at the cGA optimizing LeadingOnes. We call positions $j \in [n]$ with $\boldsymbol{p}_j = 1$ *solved* and all other positions *unsolved.* The cGA can easily solve the leftmost *(first)* unsolved position j because an individual sampled with a 1 at position j always has a higher fitness than an individual having a 0. So $\boldsymbol{p}_j$ cannot decrease, whereas the frequencies of all the other unsolved positions can. We say that an algorithm *efficiently* optimizes LeadingOnes if the expected run time of an algorithm is in $\mathrm{o}(n^2)$, since $\Theta(n^2)$ is a common run time for evolutionary algorithms (see Table 10.1). Note that the run time of the cGA is twice the number of iterations until it first samples an optimal solution, since the cGA samples two offspring each iteration.

We first look at the expected time needed for the first unsolved position to be solved, assuming its frequency is not too low.

▶ **Lemma 6.12.** Consider the cGA optimizing LeadingOnes. Further, assume that position $i \in [n]$ becomes the first unsolved position in iteration t' and that there is a constant $c \in (0,1)$ such that $\boldsymbol{p}_i^{(t')} = c$. Then $\boldsymbol{p}_i$ reaches 1 within an expected number of $\mathrm{O}\big(K \log(K)\big)$ iterations. ◀

Proof. We are going to use Corollary 3.15 in order to bound the expected value of $T = \inf\{t - t' \in \mathbf{N} \mid \boldsymbol{p}_i^{(t)} = 1\}$. Note that we consider the iteration t' to be the first iteration, since we are only interested in the number of iterations from there on. Further, since $\boldsymbol{p}_i^{(t')} < 1$, the first-hitting time T is nonnegative. In the following, we consider a time point t with $t' \leq t < T$.

Since Corollary 3.15 requires the process to move toward 0, we look at the drift of the potential $X_t = 1 - \boldsymbol{p}_i^{(t)}$. Note that $X_t < 1/K$ is equivalent to $\boldsymbol{p}_i^{(t)} = 1$.

Because i is the first unsolved position, $\boldsymbol{p}_i^{(t)}$ cannot decrease and the probability of making an increase is $2\boldsymbol{p}_i^{(t)}\big(1 - \boldsymbol{p}_i^{(t)}\big)$, otherwise it does not move. Thus, when considering the drift of X, we get

$$\mathrm{E}[X_t - X_{t+1} \mid X_t] = 2(1 - X_t)X_t\frac{1}{K}$$

$$\geq \frac{2c}{K} X_t \ .$$

By applying Corollary 3.15 with $x_{\min} = 1/K$ and noting that $X_0 = 1 - c$, we get

$$\begin{aligned} \mathrm{E}[T \mid X_0] &\leq \frac{1 + \ln\left(\frac{K(1-c)}{2c}\right)}{\frac{2c}{K}} \\ &\in \mathrm{O}\big(K \log(K)\big) \ . \end{aligned}$$ ■

Now that we took a closer look at the cGA, we want to give a general result for optimizing LeadingOnes with an n-Bernoulli-λ-EDA with no margin by sequentially solving the positions from left to right.

▶ **Theorem 6.13.** Let A be an n-Bernoulli-λ-EDA with no margin optimizing LeadingOnes, let q be a polynomial, and let $\ell \in (0, 1/2)$. Assume that there exists a constant $\varepsilon > 0$ such that, for each position $i \in [n]$, it holds that if the frequency $\boldsymbol{p}_i$ is unsolved, it only drops below ℓ within $\mathrm{O}\big(nq(n)\big)$ iterations with a probability of at most $n^{-(\varepsilon+1)}$. Furthermore, suppose that for each position j, it holds that if j becomes the *first* unsolved position in iteration t' and $\boldsymbol{p}_j^{(t')} \geq \ell$, then $\boldsymbol{p}_j$ reaches 1 within $\mathrm{O}\big(q(n)\big)$ iterations in expectation.

Then A succeeds after an expected time of $\mathrm{O}\big(nq(n)\big)$ with a probability of at least $1 - n^{-\varepsilon}$. ◀

Proof. First, we bound the probability that a frequency drops below ℓ within $\mathrm{O}\big(nq(n)\big)$ iterations. Since each frequency only does so with probability of at most $n^{-(\varepsilon+1)}$, the probability that at least one of the n frequencies does so during the same number of iterations is at most $n^{-\varepsilon}$ by a union bound. Thus, with a probability of at least $1 - n^{-\varepsilon}$, all frequencies will be at least at ℓ for $\mathrm{O}\big(nq(n)\big)$ iterations. Conditional on this event, each of the n frequencies reaches 1 within an expected time of $\mathrm{O}\big(q(n)\big)$ once it becomes the first unsolved position. This concludes the proof. ■

Note that the expected run time stated in Theorem 6.13 is only conditional on no frequency reaching 0. If we consider unbiased n-Bernoulli-λ-EDAs, we know that frequencies can reach 0 when they can reach 1, due to Lemma 5.4. Once a frequency is at 0, the run time is infinite. Hence, the unconditional expected run time for such algorithms is infinite as well, which can be seen as a major disadvantage of this approach. In Chapter 10, we circumvent this problem by introducing a better algorithm (see Algorithm 2).

Theorem 6.13 shows us that an n-Bernoulli-λ-EDA can optimize LeadingOnes in $\mathrm{O}\big(n\log(n)\big)$ if the time needed for each frequency to reach 1 is in $\mathrm{O}\big(\log(n)\big)$ and if the frequencies of yet unsolved positions do not drop too low with high probability.

6.4.1 The Stable cGA

Because Lemma 6.12 only gives us an upper bound of $\mathrm{O}\big(K\log(K)\big)$ for a frequency of the cGA to reach 1, and because the cGA is not stable (Theorem 6.10), meaning that unsolved positions may quickly reach 0, it is unlikely that the cGA solves LeadingOnes efficiently. We hence propose to change the ›set‹ function of the cGA such that the algorithm becomes stable[18] and such that each first unsolved position's frequency reaches 1 within $\mathrm{O}\big(\log(n)\big)$ rounds. We call this variant the *stable cGA* (for short: *scGA*) with parameters $\sigma \in [0, 1]$ and $d \in (1/2, 1]$. The parameter σ denotes a bias that pushes a frequency toward $1/2$, even when there is no signal from the fitness function. The parameter d denotes a value that is sufficient for the scGA in order to set the frequency, when increasing, to 1. The ›set‹ function of the scGA is as follows:

$$\mathrm{set}(\boldsymbol{p}_i) = \begin{cases} \boldsymbol{p}_i + \frac{1}{K} + \sigma & \text{if } \boldsymbol{p}_i < 1/2\,, \\ \boldsymbol{p}_i + \frac{1}{K} & \text{if } 1/2 \le \boldsymbol{p}_i < d\,, \\ 1 & \text{else.} \end{cases}$$

Recall that ›set‹ denotes the case when the frequency should increase (see Definition 4.2), hence, the bias σ is only applied if a frequency is below $1/2$. Since the ›move‹ function of the scGA is the same as that of the cGA (Section 4.3.4), when decreasing a frequency that is above $1/2$, the bias is added as well. Similarly, if a frequency goes below $1 - d$, it is immediately set to 0. Further, this implies that the scGA is unbiased (Corollary 5.14 and Theorem 5.15).

We now go into detail about how fast the scGA solves LeadingOnes. Note that since frequencies of the scGA can reach 0, the expected run time of the

[18] Note that the time period of being stable is dependent on the algorithm's parameters. We will choose its parameters such that the algorithm is $\mathrm{O}\big(n\log(n)\big)$-stable.

algorithm is infinite. As discussed before, we thus focus on the expected run time conditional on no frequency reaching 0.

First, we consider the expected time it takes for the scGA to increase the frequency of the first unsolved position to 1.

▶ **Lemma 6.14.** Consider the scGA optimizing LeadingOnes, with d being a constant and σ being arbitrary. Further, assume that position $i \in [n]$ becomes the first unsolved position in iteration t' and that there is a constant $c \in (1-d, d)$ such that $\boldsymbol{p}_i^{(t')} = c$. Then $\boldsymbol{p}_i$ reaches 1 within an expected number of $\mathrm{O}(K)$ iterations. ◀

Proof. Since i is the first unsolved position, $\boldsymbol{p}_i$ cannot decrease and, thus, the update of the scGA is the same as of that of the cGA. Hence, we can follow the proof of Lemma 6.12, and we use the same notation. The only difference in our scenario is that when applying Corollary 3.15, we now choose $x_{\min} = 1-d$, since a value of d is sufficient in order to set $\boldsymbol{p}_i$ to 1. Note that $1 - d$ is a constant by assumption. Hence, we get $\mathrm{E}[T \mid X_0] \in \mathrm{O}(K)$. ■

We now prove that the scGA is t-stable for certain values of K, σ, and d. Recall that the scGA cannot be stable, since the frequencies can reach the borders.

▶ **Theorem 6.15.** Let $\alpha > 0$ be an arbitrary constant, and let $\beta \in \Omega(1)$ be sufficiently large. Then there is a constant $c > 0$ such that the scGA with $K = \alpha/\sigma \geq \beta \ln(n)$ and $d \in (1/2, 1]$ being constant is 2^{cK}-stable. ◀

Proof. We show that with high probability no frequency will leave the interval $(1-d, d)$ within 2^{cK} iterations. Since $d \notin \mathrm{o}(1)$, this then proves that the scGA is 2^{cK}-stable. We first show that a single frequency does not leave the interval $(1-d, d)$ with high probability and then use a union bound over all n frequencies to finish the proof.

Consider a position $i \in [n]$. We use Theorem 3.22 in order to show that $\boldsymbol{p}_i$ with $\boldsymbol{p}_i^{(0)} = 1/2$ does not reach $1-d$ within 2^{cK} iterations with high probability. Since the scGA is locally updating and since we assume that the scGA is f-independent in this scenario, the same proof strategy can be used to show that $\boldsymbol{p}_i$ does not reach d within the same time and the same probability.

Because Theorem 3.22 requires the drift to be bounded by a constant, we consider the process $(X_t)_{t\in\mathrm{N}}$ with $X_t := K\boldsymbol{p}_i^{(t)}$. Using the notation of Theorem 3.22, we choose $a = (1-d)K$ and $b = K/2$, resulting in $\ell = b - a = (d-1/2)K \in \Theta(K)$, since $d \in \Theta(1)$.

We now consider the drift of X for any time point t such that $a < X_t < b$:

$$\begin{aligned}\mathrm{E}[X_{t+1} - X_t \mid X_t] &= \left(K\left(\boldsymbol{p}_i^{(t)} - \frac{1}{K}\right) - K\boldsymbol{p}_i^{(t)}\right)\boldsymbol{p}_i^{(t)}\left(1 - \boldsymbol{p}_i^{(t)}\right)\\ &\quad + \left(K\left(\boldsymbol{p}_i^{(t)} + \frac{1}{K} + \sigma\right) - K\boldsymbol{p}_i^{(t)}\right)\boldsymbol{p}_i^{(t)}\left(1 - \boldsymbol{p}_i^{(t)}\right)\\ &= (-1 + 1 + K\sigma)\boldsymbol{p}_i^{(t)}\left(1 - \boldsymbol{p}_i^{(t)}\right)\\ &= \sigma\boldsymbol{p}_i^{(t)}\left(1 - \boldsymbol{p}_i^{(t)}\right)\\ &\geq \alpha(1 - d)d\ ,\end{aligned}$$

since $\boldsymbol{p}_i^{(t)}\left(1 - \boldsymbol{p}_i^{(t)}\right)$ is a concave function and since we assume that $a < X_t < b$. Note that the drift is a positive constant, due to $d \in \Theta(1)$ and $\alpha \in \Theta(1)$.

We now focus on condition (b) of Theorem 3.22 and only consider time points where $X_t > a$. Note that the scGA can change the value of a frequency by at most $1/K + \sigma$. Hence, it holds that

$$\begin{aligned}|X_{t+1} - X_t| &\leq 1 + K\sigma\\ &= 1 + \alpha\ .\end{aligned}$$

Hence, we only need to bound the probability of $|X_{t+1} - X_t|$ being at most $1 + \alpha$.

Let $r(\ell) = 1 + \alpha$, which is in $\mathrm{o}\big(\ell/\log(\ell)\big)$, since $\alpha \in \Theta(1)$ and $\ell \in \Theta(K) \subseteq \omega(1)$. Further, let $\delta = (1 + \alpha)^{1/(1+\alpha)} - 1$. Note that δ is a constant, due to $\alpha \in \Theta(1)$, and that $\delta > 0$. Thus, for all $j \in [0..1 + \alpha]$, it holds that

$$\begin{aligned}\Pr[|X_{t+1} - X_t| \geq j \mid X_t] &\leq 1\\ &= \frac{1 + \alpha}{1 + \alpha}\\ &\leq \frac{1 + \alpha}{(1 + \alpha)^{\frac{j}{1+\alpha}}}\\ &= \frac{r(\ell)}{(1 + \delta)^j}\ ,\end{aligned}$$

since $j/(1 + \alpha) \leq 1$ for the considered range of j. For all $j > 1 + \alpha$, the probability that $|X_{t+1} - X_t| \geq j$ is 0, as we discussed before, which is trivially upper-bounded by $r(\ell)/(1 + \delta)^j$. Hence, we can apply Theorem 3.22 in order to bound the probability that $T = \inf\{t \in \mathbf{N} \mid \boldsymbol{p}_i^{(t)} \leq 1 - d\} = \inf\{t \in \mathbf{N} \mid X_t \leq a\}$. Since $\ell/r(\ell) \in \Theta(K)$ and since $K \geq \beta \ln(n)$ for β being sufficiently large, there are positive constants c, c', and c'' such that

$$\begin{aligned}\Pr\left[T \leq 2^{cK}\right] &\leq 2^{-c'K}\\ &\leq \frac{1}{n^{c''+1}}\ .\end{aligned}$$

As we discussed at the beginning of this proof, using the same arguments above in order to show that $\boldsymbol{p}_i$ does not reach d within 2^{cK} iterations with high probability and via a union bound over all n different frequencies, the probability that at least one frequency leaves the interval $(1 - d, d)$ within 2^{cK} iterations is at most $2n^{-c''}$. This concludes the proof. ■

Note that Theorem 6.15 shows that the scGA is polynomially stable if $\sigma \in \mathrm{o}\big(1/\log(n)\big)$ (when choosing the other parameters appropriately).

We can now conclude that the scGA is able to optimize LeadingOnes in $\mathrm{O}\big(n \log(n)\big)$, as we show in the following corollary.

▶ **Corollary 6.16.** Let $\alpha > 0$ be an arbitrary constant, and let $\beta \in \Theta(1)$ be sufficiently large. Then the scGA with $K = \alpha/\sigma \geq \beta \ln(n)$ and $d \in (1/2, 1)$ being constant optimizes LeadingOnes with high probability within $\mathrm{O}\big(n \log(n)\big)$ iterations in expectation. ◀

Proof. We want to use Theorem 6.13, so we make sure to fulfill the requirements. Looking into the proof of Theorem 6.15 shows us that there are constants $c, \varepsilon > 0$ such that a frequency leaves the interval $(1 - d, d)$ within $2^{cK} \in \omega\big(n \log(n)\big)$ iterations, for β being sufficiently large, only with a probability of at most $n^{-(\varepsilon+1)}$. Further, due to Lemma 6.14 and $K \in \Theta\big(\log(n)\big)$, the scGA solves a leftmost unsolved position within $\mathrm{O}\big(\log(n)\big)$ iterations in expectation when the respective probability starts from a constant value; since the scGA is stable for at least $\omega\big(n \log(n)\big)$ in our scenario, all frequencies are at least $1 - d$, which is a constant. Applying Theorem 6.13 finishes the proof. ■

Although the scGA optimizes LeadingOnes in an expected time that is uncommon for EAs (see Table 10.1), it is not able to optimize OneMax efficiently, which we prove in Chapter 10. For OneMax, the property of being stable actually hinders the scGA to update its frequencies quickly to 1, since they are constantly pushed back to $1/2$, due to every position providing a signal with a decent probability.

6.5 Conclusions

We considered the two concepts of *balanced* and *stable,* and we proved that they are mutually exclusive for n-Bernoulli-λ-EDAs. For a frequency, this means that it either stays uncommitted when there is no signal from the fitness function and has the risk of moving randomly to one of the frequency borders (also called *genetic drift*), or that it stays concentrated around its starting value of $1/2$ but has a bias when there is no signal. Since we proved that the commonly theoretically analyzed EDAs are all balanced, this implies that they are not stable and thus have problems dealing with longer phases of no signal.

In order to compensate for this problem, we introduced a new algorithm (the scGA), which is an $\mathrm{O}\big(n\log(n)\big)$-stable version of the cGA, and we proved that it optimizes LeadingOnes within $\mathrm{O}\big(n\log(n)\big)$ iterations (Corollary 6.16). Its property of being stable helps it such that frequencies at later positions stay close to $1/2$ until the respective positions become relevant. However, we would like to mention that the artificial bias in the algorithm that pushes the frequencies to $1/2$ is bad when there is no strong signal from the fitness function, as is the case for OneMax. In fact, in Chapter 10, we prove that the scGA optimizes OneMax only in a number of iterations exponential in its parameter K (Theorem 10.9).

Overall, being not stable is not necessarily a major downside of an algorithm, since it usually is sufficient to remain stable only for a period of a certain length (as is the case for the scGA). However, the longer this period can be made without impacting the ability to easily pick up signals in the fitness function, the more the algorithm reduces the impact of genetic drift and can better optimize a fitness function. In Chapter 10, we introduce an algorithm (Algorithm 2) that can be thought of as a univariate EDA which is not an n-Bernoulli-λ-EDA with both properties (modulo a margin of $1/n$ where it cannot be balanced; and *stable* meaning for a certain polynomial period of time, dependent on the algorithm's parameters). This algorithm stores information from multiple iterations until it performs an update, which is different from n-Bernoulli-λ-EDAs, which only have access to information from the current iteration. It is an interesting question whether the approach from Chapter 10 is in general better than an n-Bernoulli-λ-EDA or whether the latter class contains some algorithms that can compete with our algorithm from Chapter 10. In other words, is the restriction of having to make a decision on how to update a frequency given only data from the current iteration a strict restriction for optimization? One step into this direction would be to prove a general lower bound for balanced n-Bernoulli-λ-EDAs on LeadingOnes.

7 Upper Bound of the cGA on Noisy OneMax

This chapter is based on joint work with Tobias Friedrich, Timo Kötzing, and Andrew M. Sutton [Fri+17]. Only the theoretical results for the cGA have been included, since it is the only EDA considered in the original paper. Further, the definition of graceful scaling *has been slightly adjusted by adding a statement about the random nature of the considered algorithm, which is absent in the journal version. Parts of the following introduction have been taken from another joint work with Tobias Friedrich, Timo Kötzing, and Andrew M. Sutton on noise [Fri+16].*

In this chapter, we analyze the cGA (without a margin) on a noisy version of OneMax, where the true fitness value is distorted by adding a Gaussian random variable to it. We introduce the concept of *graceful scaling* (Definition 7.1), which describes that an algorithm is able to optimize a class of noisy functions with high probability within polynomial time, measured in the variance of the noise. We then prove that the cGA scales gracefully on OneMax with additive Gaussian noise (Theorem 7.11).

7.1 Introduction

Evolutionary algorithms are widely used for solving real-world optimization problems in uncertain environments, and EDAs seem to perform well [SGL07]. Jin and Branke [JB05] survey a number of sources of uncertainty that randomized search heuristics must often deal with in practice:

1. noisy objective functions,
2. dynamically changing problems,
3. approximation errors in the objective function, and
4. a requirement that an optimal solution must be robust to changes in its design variables and its environmental parameters that occur after optimization is complete.

Arguably, the two most important sources of uncertainty are point 1 and point 2, namely, *stochastic* problems and *dynamic* problems (see also [Bia+09] for a recent survey). We focus on stochastic problems, that is, problems, where the objective function value of a search point follows a random distribution, and that distribution does not change over time. In these scenarios, EAs must somehow filter the fitness signal from the noise. If the noise intensity is relatively small, this poses little to no problems to selection. However, as the noise intensity grows, the signal becomes more obscured, and the picture is no longer as clear.

A first categorization of different types of noise in optimization was given by Beyer et al. [BOS02] and further developed by Beyer and Sendhoff [BS07]. The authors delineate several types of noise:

1. noise from the environment,
2. actuator imprecision (noise in the decision variables),
3. imprecision of the evaluation of system output, and
4. uncertainties regarding feasibility constraints.

We focus on uncertainty of type 3, in which the measurement of the objective function is perturbed by some additive stochastic noise term. This noise model has also been called *additive posterior noise* by Gießen and Kötzing [GK16] and has been studied in the context of combinatorial optimization in several papers [DHK12; FK13; GP96; ST12].

The rigorous analysis of the run time of EAs on noisy functions on discrete domains was initiated by Droste [Dro04]. In that paper, a noisy variant of the OneMax test function was analyzed for the simplest EA: the (1+1) EA. In essence, it was shown that the (1+1) EA can deal with small noise levels, but not medium noise levels.

In this chapter, we address the dependency of optimization time on noise intensity (measured as the variance). Specifically, we ask whether there is some kind of threshold point in the noise intensity at which the noise becomes too high for efficient optimization or whether it is possible for algorithms to be somehow robust to scaling of the noise level.

In order to formally characterize how search heuristics can exhibit robustness to noise intensity, we introduce the concept of *graceful scaling* (Definition 7.1). Intuitively, a search heuristic scales gracefully with noise if a polynomial increase of the variance can be compensated by polynomially more resources.

We consider centered Gaussian noise with variance σ^2 and use OneMax as the underlying fitness function. In the journal version that this chapter is based on [Fri+17], we proved for this setting that a constant variance can already lead to a superpolynomial run time for the simple hillclimbers Random Local Search (RLS) and (1+1) EA. Further, the (μ+1) EA using *any polynomial population size* of at least $\omega(1)$ is proven to be inefficient for a noise intensity of $\sigma^2 \in \omega\big(n^2\big)$.[19]

We investigate the cGA (Section 4.3.4), whose working principles as an EDA are in contrast to the (μ+1) EA, which is an EA. Rather than relying on an explicit population, where a good individual may be removed due to a bad fitness evaluation, the cGA maintains a probability vector that serves as a history, which is updated by at most $1/K$ in each iteration. This approach allows the cGA to smooth the noise sufficiently without having to resort to explicit noise-handling strategies. We prove that as long as K is sufficiently large, that is, $K \in \omega\big(\sigma^2\sqrt{n}\log(n)\big)$, then the cGA scales gracefully with Gaussian noise. Specifically, we prove that after $\mathrm{O}\big(K\sigma^2\sqrt{n}\log(Kn)\big)$ many iterations, the cGA will have sampled the all-1s string with high probability, as desired (Theorem 7.11). In other words, there is no threshold point in noise intensity at which the algorithm begins to perform poorly.

The proof of Theorem 7.11 gives insight into how the cGA can filter the signal out of the noise efficiently. Even under intense noise (that is, a large variance), as long as the step size $1/K$ is small enough, selection errors due to misclassification by the noisy function are not penalized greatly, and the overall effect of selection biases the stochastic process described by the frequencies toward the optimal configuration. This can be contrasted with mutation-only approaches, for which such selection errors become fatal in the sense that progress in the correct direction is no longer visible to the algorithm.

7.2 Preliminaries

We study the cGA (see Section 4.3.4) optimizing OneMax with additive Gaussian noise, parameterized by the variance. Since the cGA is unbiased (Theorem 5.15), our results also hold for the entire OneMax class. However, for the sake of clarity, we phrase all of our results choosing OneMax with the all-1s string as its optimum.

When considering the optimization of a noisy fitness function f, two evaluations of f for the same point can lead to different results, as the noise is drawn

[19] We would like to mention that the proof of the theorem that this result is based on [Fri+17, Theorem 4] seems to be incorrect, as it contains an incomplete case distinction. However, the authors believe that the proof can be fixed and that the respective theorem is true.

again with every evaluation. In this chapter, we assume that each search point is evaluated for its fitness value in each iteration anew. That is, after each iteration, the fitness evaluation is discarded, even if the search point is not. Further, we say that an algorithm succeeds in optimizing a noisy function if and only if it samples an optimum of the underlying unnoisy function.

Let F be a family of pseudo-Boolean functions $\{F_n\}_{n \in \mathbf{N}}$ where each F_n is a set of (unnoisy) functions $f\colon \{0,1\}^n \to \mathbf{R}$. Let D be a family of distributions $\{D_\theta\}_{\theta \in \mathbf{R}}$ such that, for all $D_\theta \in D$, we have $\mathrm{E}[D_\theta] = 0$. We define F with additive D-noise as the set $F[D] := \{f_n + D_\theta \mid f_n \in F_n \wedge D_\theta \in D\}$.

We now define when an algorithm is robust to additive noise.

▶ **Definition 7.1 (Graceful Scaling).** An algorithm A *scales gracefully with noise* on $F[D]$ if there is a polynomial q such that, for all $g_{n,\theta} = f_n + D_\theta \in F[D]$, there exists a parameter setting p such that $A(p)$ finds the optimum of f_n using at most $q(n, \theta)$ calls to $g_{n,\theta}$ with high probability. ◀

Let $\sigma^2 \geq 0$ be a variance, and let $Z \sim \mathrm{N}(0, \sigma^2)$, that is, Z is a normally distributed random variable with mean 0 and variance σ^2. We then denote the *noisy* OneMax *function* by $\textsc{OneMax}_{[\sigma^2]}\colon \boldsymbol{x} \mapsto \|\boldsymbol{x}\|_1 + Z$.

The following lemma gives tail bounds for Z by using estimates of the complementary error function. This will be useful in our proofs for bounding the probability that the $\textsc{OneMax}_{[\sigma^2]}$ correctly ranks two arbitrary search points.

▶ **Lemma 7.2.** Let $Z \sim \mathrm{N}(0, \sigma^2)$. For all $t > 0$, it holds that

$$\Pr[Z < -t] \leq \frac{1}{2} \mathrm{e}^{\frac{-t^2}{2\sigma^2}} . \quad ◀$$

Proof. For any $t > 0$, by the definition of $\operatorname{erfc}(t) = (2/\sqrt{\pi}) \int_t^\infty \mathrm{e}^{-t^2} \mathrm{d}t$ and the definition of a Gaussian distribution, the tail bound of Z is given by $\Pr[Z < -t] = (1/2) \operatorname{erfc}\bigl(t/(\sigma\sqrt{2})\bigr)$. Chang et al. [CCM11] bound $\operatorname{erfc}(t)$ by $\alpha \mathrm{e}^{-\beta t^2}$. Choosing $\alpha = \beta = 1$ yields the desired upper bound. ■

In order to prove our main result, we use the following drift theorem, which we phrase in a similar fashion to the theorems in Chapter 3. It is basically Corollary 3.15 but additionally provides a tail bound on how likely it is that the process has *not* reached its goal within the denoted time. We only mention the concentration bound.

▶ **Theorem 7.3 (Tail Bound For Multiplicative Drift [DG13]).** Let $(X_t)_{t \in \mathbf{N}}$ be a sequence of random variables over $\{0\} \cup [x_{\min}, x_{\max}]$, where $x_{\min} > 0$, adapted to a filtration $(\mathcal{F}_t)_{t \in \mathbf{N}}$, and let $T = \inf\{t \in \mathbf{N} \mid X_t = 0\}$. Furthermore, suppose that there exists a value $\delta > 0$ such that, for all $t < T$, it holds that

$$\mathrm{E}[X_t - X_{t+1} \mid \mathcal{F}_t] \geq \delta X_t \ .$$

Then, for all $\lambda > 0$, it holds that

$$\Pr\left[T > \frac{\lambda + \ln\left(\frac{X_0}{x_{\min}}\right)}{\delta} \;\middle|\; X_0\right] \leq \mathrm{e}^{-\lambda} \ . \qquad ◀$$

The following lemma states an exact equality for the first absolute moment of a random variable Z in terms of its characteristic function $\varphi_Z(t) = \mathrm{E}\left[\mathrm{e}^{\mathrm{i}tZ}\right]$.

▶ **Lemma 7.4 ([BE65, Special Case of Lemma 2]).** Let Z be a random variable with $\mathrm{E}[|Z|] < \infty$. Then

$$\mathrm{E}[|Z|] = \frac{1}{\pi} \int_{-\infty}^{\infty} \frac{1 - \Re\left(\varphi_Z(t)\right)}{t^2} \, \mathrm{d}t \ ,$$

where $\Re(z)$ is the real part of $z \in \mathbf{C}$. ◀

7.3 Formal Analysis

We now show that the cGA (without a margin) scales gracefully with Gaussian noise. Let $T^\star$ be the optimization time of the cGA on $\textsc{OneMax}_{[\sigma^2]}$, namely, the first time that it generates the underlying true optimal solution 1^n. Since we want to apply drift theory and thus need our target to be 0, we consider the random process $(X_t)_{t \in \mathbf{N}}$ defined as

$$X_t = n - \sum_{i=1}^{n} \boldsymbol{p}_i^{(t)} \ . \tag{7.1}$$

Further, we bound the optimization time by $T = \inf\{t \in \mathbf{N} \mid X_t = 0\}$. This is the time until the product distribution has converged to the optimal frequency vector. Clearly, $T^\star \leq T$ since the cGA produces 1^n in the T-th iteration almost surely. However, $T^\star$ and T can be infinite when there is a position $i \in [n]$ and a time point $t < T^\star$ where $\boldsymbol{p}_i^{(t)} = 0$, since the process can never subsequently generate any string $\boldsymbol{x}$ with $\boldsymbol{x}_i = 1$. In order to circumvent this, Droste [Dro06] estimates $\mathrm{E}\left[T^\star\right]$ conditioned on the event that $T^\star$ is finite, and then bounds the probability of finite $T^\star$. We follow this approach and prove that as long as K is large enough, the optimization time is polynomial with high probability.

We first define the probability that the cGA misclassifies two points under comparison due to the presence of noise.

▶ **Definition 7.5.** Let $\boldsymbol{x}, \boldsymbol{y} \in \{0,1\}^n$. Without loss of generality, suppose that $\|\boldsymbol{x}\|_1 - \|\boldsymbol{y}\|_1 = \ell \geq 0$, and let $\mathcal{E}$ be the event that $\textsc{OneMax}_{[\sigma^2]}(\boldsymbol{x}) < \textsc{OneMax}_{[\sigma^2]}(\boldsymbol{y})$. We define the *probability to misclassify* $\boldsymbol{x}$ *and* $\boldsymbol{y}$ as the function $\Delta\colon [n] \cup \{0\} \to [0,1]$ with

$$\Delta(\ell) = \begin{cases} \frac{1}{2} & \text{if } \ell = 0\,, \\ \Pr\left[\mathcal{E} \,\middle|\, \|\boldsymbol{x}\|_1 - \|\boldsymbol{y}\|_1 = \ell\right] & \text{if } \ell > 0\,. \end{cases}$$ ◀

Note that we define the probability of misclassification in the case of $\ell = 0$ to be $1/2$ because we care about which individual will be chosen in this case.

The following lemma bounds the misclassification probability in terms of the noise intensity measured by the variance.

▶ **Lemma 7.6.** For any $\ell \in [n]$, it holds that $\Delta(\ell) > \Delta(\ell+1)$. Moreover, assuming that $\sigma^2 > 0$, there exists a constant $c > 0$ such that

$$\Delta(\ell) \leq \frac{1}{2}\left(1 - \frac{c}{\sigma^2}\right).$$ ◀

Proof. Let $\boldsymbol{x}, \boldsymbol{y} \in \{0,1\}^n$ be any bit strings such that $\|\boldsymbol{x}\|_1 - \|\boldsymbol{y}\|_1 = \ell$. The event that $\textsc{OneMax}_{[\sigma^2]}$ incorrectly classifies $\boldsymbol{y}$ as superior to $\boldsymbol{x}$ is equivalent to the event $\{\textsc{OneMax}_{[\sigma^2]}(\boldsymbol{x}) < \textsc{OneMax}_{[\sigma^2]}(\boldsymbol{y})\}$. Let $Z_{\boldsymbol{x}}, Z_{\boldsymbol{y}} \sim \mathrm{N}(0, \sigma^2)$ denote the noise from the evaluation of $\boldsymbol{x}$ and $\boldsymbol{y}$, respectively. Then we get

$$\Pr\left[\textsc{OneMax}_{[\sigma^2]}(\boldsymbol{x}) < \textsc{OneMax}_{[\sigma^2]}(\boldsymbol{y})\right] = \Pr\left[\ell + (Z_{\boldsymbol{x}} - Z_{\boldsymbol{y}}) < 0\right].$$

Letting $Z^* := Z_{\boldsymbol{x}} - Z_{\boldsymbol{y}}$, we have $Z^* \sim \mathrm{N}(0, 2\sigma^2)$ and $\Delta(\ell) = \Pr[Z^* < -\ell]$. Furthermore, $\Delta(\ell+1) = \Pr[Z^* < -(\ell+1)] < \Pr[Z^* < -\ell] = \Delta(\ell)$. Finally, applying Lemma 7.2, we have $\Pr[Z^* < -\ell] \leq (1/2)\mathrm{e}^{-\ell^2/(4\sigma^2)} \leq (1/2)\mathrm{e}^{-1/(4\sigma^2)}$. The proof is then completed by applying the well-known bound $1 - z > \mathrm{e}^{-z/(1-z)}$ [Mit64, Inequality 2.68] and setting $z = 1/(4\sigma^2 + 1)$. ■

We now prove the following lemma, which we use in the proof of the upcoming Lemma 7.8.

▶ **Lemma 7.7.** Let $k \in \mathbf{N}$. Then

$$\frac{1}{\pi} \int_{-\infty}^{\infty} \frac{1 - \cos^k(t)}{t^2} \,\mathrm{d}t = \frac{2}{2^{2\left\lceil \frac{k}{2} \right\rceil}} \left\lceil \frac{k}{2} \right\rceil \binom{2\left\lceil \frac{k}{2} \right\rceil}{\left\lceil \frac{k}{2} \right\rceil} .$$ ◀

Proof. We express the k-th power of $\cos(t)$ in terms of the binomial expansion [GR07, Equations 1.320.5 and 1.320.7]:

$$\cos^k(t) = \frac{1}{2^k} \sum_{j=0}^{k} \binom{k}{j} \cos\big((2j - k)t\big) .$$

Thus, we may calculate the indefinite integral

$$\begin{aligned}
&\int \frac{1 - \cos^k(t)}{t^2} \,\mathrm{d}t \\
&= -\frac{1}{2^k} \sum_{j=0}^{k} \binom{k}{j} \int \frac{\cos((2j-k)t)}{t^2} \,\mathrm{d}t - \frac{1}{t} \\
&= -\frac{1}{2^k} \sum_{j=0}^{k} \binom{k}{j} \left(-\frac{\cos((2j-k)t)}{t} - (2j - k) \int \frac{\sin((2j-k)t)}{t} \mathrm{d}t \right) - \frac{1}{t} \\
&= \frac{\cos^k(t)}{t} - \frac{1}{2^k} \sum_{j=0}^{k} \binom{k}{j} (k - 2j) \operatorname{Si}\big((2j - k)t\big) - \frac{1}{t} ,
\end{aligned} \tag{7.2}$$

where $\operatorname{Si}(z) := \int_0^z \frac{\sin(x)}{x} \,\mathrm{d}x$ is the sine integral,[20] and where $\int \cos((2j - k)t) \cdot$

[20] Since we are eventually interested in the definite integral from negative infinity to positive infinity, the sine integral is derived by a case distinction of the intervals $(-\infty, 0)$ and $[0, \infty)$. Although Si is only defined over $[0, \infty)$, we extend it such that, for any $z \in \mathbf{R}^+$, $\operatorname{Si}(-z) := -\operatorname{Si}(z)$, since $\sin(-z) = -\sin(z)$. This saves us a case distinction in the places where we use Si.

$(1/t^2)\mathrm{d}t$ is integrated by parts once, noting that

$$\frac{\partial \cos((2j-k)t)}{\partial t} = -(2j-k) \cdot \sin((2j-k)t) \, .$$

We define the function

$$h_k(t) := -\frac{1}{2^k} \sum_{j=0}^{k} \binom{k}{j} (k-2j) \operatorname{Si}\big((2j-k)t\big) \, .$$

By the algebraic limit theorem,

$$\lim_{t \to \infty} h_k(t) = -\frac{1}{2^k} \sum_{j=0}^{k} \binom{k}{j} (k-2j) \lim_{t \to \infty} \operatorname{Si}\big((2j-k)t\big) \, .$$

The limits at infinity of the sine integral are [AS65, Equation 5.2.25]

$$\lim_{t \to \infty} \operatorname{Si}\big((2j-k)t\big) = \begin{cases} \frac{\pi}{2} & \text{if } (2j-k) > 0 \, , \\ 0 & \text{if } (2j-k) = 0 \, , \\ -\frac{\pi}{2} & \text{if } (2j-k) < 0 \, . \end{cases}$$

Therefore,

$$\begin{aligned} \lim_{t \to \infty} h_k(t) &= \frac{-\pi}{2^{k+1}} \left(\sum_{j=\lceil \frac{k}{2} \rceil}^{k} \binom{k}{j} (k-2j) - \sum_{j=0}^{\lfloor \frac{k}{2} \rfloor} \binom{k}{j} (k-2j) \right) \\ &= \frac{\pi}{2^{k+1}} \left(\sum_{j=0}^{\lfloor \frac{k}{2} \rfloor} \binom{k}{j} (k-2j) - \sum_{j=\lceil \frac{k}{2} \rceil}^{k} \binom{k}{j} (k-2j) \right) \\ &= \frac{\pi}{2^{k+1}} \left(\left\lceil \frac{k}{2} \right\rceil \binom{2\lceil \frac{k}{2} \rceil}{\lceil \frac{k}{2} \rceil} I(k) \right) , \end{aligned}$$

where

$$I(k) = \begin{cases} 1 & \text{if } k \text{ is odd,} \\ 2 & \text{if } k \text{ is even.} \end{cases}$$

The last step of the transformations above can be proven inductively by noting that $s(k) := \lceil k/2 \rceil \binom{2\lceil k/2 \rceil}{\lceil k/2 \rceil} I(k)$ follows the recursion

$$s(k) = \begin{cases} 2 \cdot s(k-1) + 2\binom{k-1}{\frac{k-1}{2}} & \text{if } k \text{ is odd,} \\ 2 \cdot s(k-1) & \text{if } k \text{ is even, and} \end{cases}$$
$$s(0) = 0\ .$$

It is possible to express $I(k) = 2^{(k+1)}/2^{2\lceil k/2 \rceil}$, and so

$$\lim_{t\to\infty} h_k(t) = \pi \left\lceil \frac{k}{2} \right\rceil \binom{2\lceil \frac{k}{2} \rceil}{\lceil \frac{k}{2} \rceil} 2^{-2\lceil \frac{k}{2} \rceil}\ . \tag{7.3}$$

A similar derivation yields

$$\lim_{t\to-\infty} h_k(t) = -\lim_{t\to\infty} h_k(t)\ . \tag{7.4}$$

In order to complete the proof, we take the indefinite integral derived in equation (7.2) and use that $\lim_{t\to\infty} \cos^k(t)/t = \lim_{t\to\infty} 1/t = 0$ to compute the definite integral as follows:

$$\begin{aligned} \int_{-\infty}^{\infty} \frac{1-\cos^k(t)}{t^2}\,\mathrm{d}t &= \lim_{t\to\infty}\left(\frac{\cos^k(t)}{t} + h_k(t) - \tfrac{1}{t}\right) - \lim_{t\to-\infty}\left(\frac{\cos^k(t)}{t} + h_k(t) - \tfrac{1}{t}\right) \\ &= \lim_{t\to\infty} h_k(t) - \lim_{t\to-\infty} h_k(t) \\ &= 2\lim_{t\to\infty} h_k(t)\ , \end{aligned}$$

where we have used equation (7.4). Finally, substituting equation (7.3) for the limit at positive infinity, we get

$$\int_{-\infty}^{\infty} \frac{1-\cos^k(t)}{t^2}\,\mathrm{d}t = \pi \cdot 2\left\lceil \frac{k}{2} \right\rceil \binom{2\lceil \frac{k}{2} \rceil}{\lceil \frac{k}{2} \rceil} 2^{-2\lceil \frac{k}{2} \rceil}\ . \qquad \blacksquare$$

We can now prove the following technical lemma that yields some properties of the sum of n independent random variables over $\{-1, 0, 1\}$. Specifically, we derive a bound on the probability that the sum is equal to 0 and a lower bound on the first absolute moment of the sum. We later use these results in proofs about the drift of the frequencies during the run of the cGA.

▶ **Lemma 7.8.** Let $a \in (0, 1)$ be a constant, and let $\boldsymbol{p} \in [a, 1]^n$ be a probability vector. Consider a random variable $Z = \sum_{i=1}^{n} Z_i$, each Z_i independent, where

$$Z_i = \begin{cases} 1 & \text{with probability } \boldsymbol{p}_i(1-\boldsymbol{p}_i)\,, \\ -1 & \text{with probability } \boldsymbol{p}_i(1-\boldsymbol{p}_i)\,, \text{ and} \\ 0 & \text{with probability } 1-2\boldsymbol{p}_i(1-\boldsymbol{p}_i)\,. \end{cases}$$

Then $\Pr[Z = 0] \geq 1/(4\sqrt{n})$, and

$$\mathrm{E}\big[|Z|\big] \geq a\sqrt{\frac{2}{n}}\left(n - \sum_{i=1}^{n} \boldsymbol{p}_i\right).$$ ◀

Proof. Let $\xi = \sum_{i=1}^{n} |Z_i|$. Then ξ is distributed as a Poisson binomial distribution with each success probability equal to $2\boldsymbol{p}_i(1-\boldsymbol{p}_i)$. Furthermore, $Z = 0$ when exactly k of the Z_i variables are non-zero for some even k, and exactly $k/2$ of these are selected to be negative, with the remaining $k/2$ positive. The probability that exactly k of the variables are non-zero is $\Pr[\xi = k]$, and the probability of selecting exactly $k/2$ to be positive is $\binom{k}{k/2}2^{-k}$ so we can write

$$\Pr[Z = 0] = \sum_{k=0}^{n} \Pr[\xi = k]\binom{k}{\frac{k}{2}}2^{-k}\,,$$

where $\binom{k}{k/2} = 0$ if k is odd. Since $\binom{k}{k/2}$ vanishes at odd i, we have

$$\Pr[Z = 0] = \sum_{k=0}^{\lfloor n/2 \rfloor} \Pr[\xi = 2k]\binom{2k}{k}2^{-2k}\,.$$

$\binom{2k}{k}$ is the k-th central binomial coefficient, for which we have the well-known bound $2^{2k}/(2\sqrt{k}) \leq \binom{2k}{k}$ [Bul15, Binomial Coefficient Inequalities (b)], hence,

$$\Pr[Z = 0] \geq \Pr[\xi = 0] + \sum_{k=1}^{\lfloor n/2 \rfloor} \Pr[\xi = 2k]\frac{1}{2\sqrt{k}}$$

$$\geq \frac{1}{2\sqrt{n}} \Pr[\xi \text{ is even}] \,, \tag{7.5}$$

since $1/(2\sqrt{n}) \leq 1/(2\sqrt{k})$.

In order to finish the proof, note that for any integer random variable X, where we denote the probability generating function by $G(z) = \mathrm{E}\big[z^X\big]$, it holds that $\Pr[X \text{ is even}] = \big(1 + G(-1)\big)/2$. Since ξ is a Poisson binomial distribution with success probability q_i for the i-th trial, we can write the probability generating function as $G(z) = \prod_{i=1}^{n}(1 - q_i + q_i z)$ (by the binomial theorem). Therefore,

$$\begin{aligned} \Pr[\xi \text{ is even}] &= \frac{1 + G(-1)}{2} \\ &= \frac{1}{2}\left(1 + \prod_{i=1}^{n}(1 - 2q_i)\right) . \end{aligned}$$

Finally, since $q_i = 2\boldsymbol{p}_i(1 - \boldsymbol{p}_i) \leq 1/2$ for all $i \in [n]$, the product in the above equation must be nonnegative. Therefore, $\Pr[\xi \text{ is even}] \geq 1/2$ and the claimed bound on $\Pr[Z = 0]$ follows from equation (7.5).

We now bound the first absolute moment of Z from below. For every $S \subseteq [n]$, let $\mathcal{E}_S$ denote the event that $|Z_i| = 1 \leftrightarrow i \in S$. We first calculate the expectation of $|Z|$ conditional on these events. Since the probabilities $\boldsymbol{p}_i$ are mutually independent, we get

$$\begin{aligned} \mathrm{E}\Big[\mathrm{e}^{\mathrm{i}tZ} \;\Big|\; \mathcal{E}_S\Big] &= \mathrm{e}^{\mathrm{i}t\mathrm{E}[Z \,\mid\, \mathcal{E}_S\,]} \\ &= \prod_{j=1}^{n} \mathrm{e}^{\mathrm{i}t\mathrm{E}[Z_j \,\mid\, \mathcal{E}_S\,]} \,. \end{aligned}$$

If $j \in S$, then $Z_j = 1$ with probability $1/2$ and $Z_j = -1$ with probability $1/2$. Thus,

$$\mathrm{e}^{\mathrm{i}t\mathrm{E}[Z_j \,\mid\, \mathcal{E}_{S \wedge j \in S}\,]} = \frac{\mathrm{e}^{\mathrm{i}t}}{2} + \frac{\mathrm{e}^{-\mathrm{i}t}}{2} \,.$$

On the other hand, if $j \notin S$, then $Z_j = 0$ and so

$$\begin{aligned} \mathrm{e}^{\mathrm{i}t\mathrm{E}[Z_j \,\mid\, \mathcal{E}_{S \wedge j \notin S}\,]} &= \mathrm{e}^0 \\ &= 1 \,. \end{aligned}$$

We collect the above terms to write

$$\begin{aligned}\mathrm{E}\left[\mathrm{e}^{\mathrm{i}tZ} \mid \mathcal{E}_S\right] &= \prod_{j=1}^{n}\left(\mathbf{1}_S(j)\left(\frac{\mathrm{e}^{\mathrm{i}t}}{2} + \frac{\mathrm{e}^{-\mathrm{i}t}}{2}\right) + 1 - \mathbf{1}_S(j)\right)\\ &= \prod_{j\in S}\cos(t) = \cos^{|S|}(t)\ .\end{aligned}$$

Thus, by Lemma 7.4,

$$\begin{aligned}\mathrm{E}[|Z| \mid \mathcal{E}_S] &= \frac{1}{\pi}\int_{-\infty}^{\infty}\frac{1-\cos^{|S|}(t)}{t^2}\,\mathrm{d}t\\ &= g(|S|)\ ,\end{aligned}$$

where $g(k) = 2\lceil k/2\rceil\binom{2\lceil k/2\rceil}{\lceil k/2\rceil}2^{-2\lceil k/2\rceil}$ by Lemma 7.7. Again applying bounds on the central binomial coefficient, we get $g(k) \geq \sqrt{\lceil k/2\rceil} \geq \sqrt{k/2}$. And since $k \leq n$, we have $g(k) \geq k/\sqrt{2n}$.

By the law of total expectation (Corollary 2.10), we get

$$\begin{aligned}\mathrm{E}[|Z|] &= \sum_{k=1}^{n} g(k) \sum_{S\subseteq[n]\colon |S|=k}\Pr[\mathcal{E}_S]\\ &\geq \frac{1}{\sqrt{2n}}\sum_{k=1}^{n} k \sum_{S\subseteq[n]\colon |S|=k}\Pr[\mathcal{E}_S]\\ &= \frac{\mathrm{E}[\xi]}{\sqrt{2n}}\ . \qquad (7.6)\end{aligned}$$

Since ξ follows a Poisson binomial distribution with the i-th success probability equal to $2\boldsymbol{p}_i(1-\boldsymbol{p}_i)$, and since every $\boldsymbol{p}_i \geq a$, we have

$$\begin{aligned}\mathrm{E}[\xi] &= \sum_{i=1}^{n} 2\boldsymbol{p}_i(1-\boldsymbol{p}_i)\\ &\geq 2a\left(n - \sum_{i=1}^{n}\boldsymbol{p}_i\right)\ .\end{aligned}$$

Substituting this inequality into inequality (7.6) completes the proof. ■

We use Lemma 7.8 in the proof of the next lemma whenever we need bounds on the expected absolute difference of the count of 1s between the offspring generated by the cGA. Specifically, the following lemma bounds the drift on X_t as defined in equation (7.1), conditional on the event that no frequency gets too small.

▶ **Lemma 7.9.** Consider the cGA optimizing $\textsc{OneMax}_{[\sigma^2]}$ and let X be the random process defined in equation (7.1). Assume that there exists a constant $a > 0$ such that, for all $i \in [n]$ and all $t < T$, we have $\boldsymbol{p}_i^{(t)} \geq a$ and that $X_t > 0$. Then

$$\mathrm{E}[X_t - X_{t+1} \mid X_t] \geq \delta X_t \ ,$$

where $1/\delta \in \mathrm{O}\big(\sigma^2 K \sqrt{n}\big)$. ◀

Proof. Let $\boldsymbol{x}$ and $\boldsymbol{y}$ be the offspring generated in iteration t, and let $Z_t = \|\boldsymbol{x}\|_1 - \|\boldsymbol{y}\|_1$. For all $i \in [n]$ defining

$$Z_{i,t} = \begin{cases} -1 & \text{if } \boldsymbol{x}_i = 0 \text{ and } \boldsymbol{y}_i = 1, \\ 0 & \text{if } \boldsymbol{x}_i = \boldsymbol{y}_i, \\ 1 & \text{if } \boldsymbol{x}_i = 1 \text{ and } \boldsymbol{y}_i = 0, \end{cases}$$

note that $Z_t = \sum_{i=1}^{n} Z_{i,t}$. Let $\mathcal{E}$ denote the event that the evaluation of $\textsc{OneMax}_{[\sigma^2]}$ incorrectly ranks $\boldsymbol{x}$ and $\boldsymbol{y}$. Without loss of generality, suppose $\|\boldsymbol{x}\|_1 \geq \|\boldsymbol{y}\|_1$. Then $\mathrm{E}\big[X_t - X_{t+1} \,\big|\, X_t; \overline{\mathcal{E}}\big] = \mathrm{E}\big[|Z_t| \,\big|\, X_t\big]/K$. On the other hand, if $\textsc{OneMax}_{[\sigma^2]}(\boldsymbol{x})$ evaluates to at most $\textsc{OneMax}_{[\sigma^2]}(\boldsymbol{y})$ during iteration t, the roles above are swapped and $\mathrm{E}[X_t - X_{t+1} \mid X_t; \mathcal{E}] = -\mathrm{E}\big[|Z_t| \,\big|\, X_t\big]/K$. By the law of total expectation, we get

$$\mathrm{E}[X_t - X_{t+1} \mid X_t] = \frac{\mathrm{E}\big[|Z_t| \,\big|\, X_t\big]}{K}\big(1 - 2\Pr[\mathcal{E}]\big) \ . \tag{7.7}$$

Note that, for any $i \in [n]$, it holds that $\Pr\big[Z_{i,t} = 1 \,\big|\, X_t\big] = \Pr\big[Z_{i,t} = -1 \,\big|\, X_t\big] = \boldsymbol{p}_i^{(t)}(1 - \boldsymbol{p}_i^{(t)})$ and $\Pr\big[Z_{i,t} = 0 \,\big|\, X_t\big]$ is the converse probability. Since we have assumed each $\boldsymbol{p}_i^{(t)} \geq a$, we can apply Lemma 7.8 to obtain

$$\mathrm{E}\big[|Z_t| \,\big|\, X_t\big] \geq a\sqrt{\frac{2}{n}}\left(n - \sum_{i=1}^{n} \boldsymbol{p}_i^{(t)}\right)$$

$$= aX_t\sqrt{\frac{2}{n}}\,. \tag{7.8}$$

Last, we substitute inequality (7.8) into equation (7.7) and use Lemma 7.6 to bound $\Pr[\mathcal{E}] = \Delta\big(\big|\|\boldsymbol{x}\|_1 - \|\boldsymbol{y}\|_1\big|\big)$ from above. This completes the proof. ■

In order to use Lemma 7.9, we require that the frequencies stay large enough during the run of the cGA. Increasing the step size K obviously translates to finer-grained frequency values, which means slower dynamics for $\boldsymbol{p}_i^{(t)}$. We point out that this scaling of the update probabilities is an application of the technique of *rescaled mutations* [Bey00] to the space of probability vectors. Provided that K is set sufficiently large, the frequencies remain above an arbitrary constant for any polynomial number of iterations with very high probability. This is captured by the following lemma.

▶ **Lemma 7.10.** Consider the cGA optimizing $\text{OneMax}_{[\sigma^2]}$ with $\sigma^2 > 0$. Let $a \in (0, 1/2)$ be an arbitrary constant, and let $T' = \inf\{t \in \mathbf{N} \mid \exists i \in [n] \colon \boldsymbol{p}_i^{(t)} \le a\}$. If $K \in \omega\big(\sigma^2\sqrt{n}\log(n)\big)$, then for every polynomial $\operatorname{poly}(n)$, for n sufficiently large, $\Pr[T' < \operatorname{poly}(n)]$ is superpolynomially small. ◀

Proof. We want to apply Corollary 3.24 and show that the probability of a frequency reaching a is superpolynomially small.

Consider a position $i \in [n]$ of the cGA. Let $(Y_t)_{t\in\mathbf{N}}$ be the stochastic process $Y_t = \big(1/2 - \boldsymbol{p}_i^{(t)}\big)K$. Note that Y is a random process on the set $[-K/2..K/2]$ with $Y_0 = 0$. Let $T = \inf\{t \in \mathbf{N} \mid Y_t > (1/2 - a)K\}$. We now show that there is a negative drift as long as $Y_t \ge -K/2 + 1$.

We first argue that there is a constant $c > 0$ such that, for all $t < T$, it holds that

$$\mathrm{E}[Y_t \mid Y_{t-1}] \le Y_{t-1} - \frac{c}{\sigma^2} \cdot \frac{\Pr[\boldsymbol{x}_i \ne \boldsymbol{y}_i]}{\sqrt{n}}\,. \tag{7.9}$$

Consider an iteration $t < T'$, and let $\boldsymbol{x}$ and $\boldsymbol{y}$ be the strings generated by the cGA in iteration that iteration. We define $\hat{\boldsymbol{x}} \coloneqq (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_{i-1}, \boldsymbol{x}_{i+1}, \ldots \boldsymbol{x}_n)$ to be the substring of $\boldsymbol{x}$ constructed by removing the i-th element and $\hat{\boldsymbol{y}}$ analogously. Since each element of $\boldsymbol{x}$ and $\boldsymbol{y}$ is constructed independently, we can regard $\hat{\boldsymbol{x}}$, $\hat{\boldsymbol{y}}$, $\boldsymbol{x}_i$, and $\boldsymbol{y}_i$ to be independent.

Define the random variable $\delta_t \coloneqq Y_t - Y_{t-1}$. Note that $\mathrm{E}[Y_t \mid Y_{t-1}] = Y_{t-1} + \mathrm{E}[\delta_t \mid Y_{t-1}]$, where $\delta_t \in \{-1, 0, 1\}$. Further, define $\hat{\ell} = \|\hat{\boldsymbol{x}}\|_1 - \|\hat{\boldsymbol{y}}\|_1$. We distinguish between the two events that $|\hat{\ell}|$ is non-zero or zero.

Case $|\hat{\ell}| > 0$. Suppose without loss of generality that $\hat{\ell} > 0$ (that is, $\|\hat{\boldsymbol{x}}\|_1 > \|\hat{\boldsymbol{y}}\|_1$). Hence, $\delta_t = 0$ if and only if $\boldsymbol{x}_i = \boldsymbol{y}_i$. Moreover, $\delta_t = -1$ only in the case that

(a) $\boldsymbol{x}_i = 1$ and $\boldsymbol{y}_i = 0$ and $\boldsymbol{x}$ is accepted (in which case $\ell = \hat{\ell} + 1$), or

(b) $\boldsymbol{x}_i = 0$ and $\boldsymbol{y}_i = 1$ and $\boldsymbol{x}$ is *not* accepted (in which case $\ell = \hat{\ell} - 1$).

Event (a) occurs only if $\textsc{OneMax}_{[\sigma^2]}$ does not misclassify $\boldsymbol{x}$ and $\boldsymbol{y}$, whereas event (b) occurs only if $\textsc{OneMax}_{[\sigma^2]}$ does misclassify $\boldsymbol{x}$ and $\boldsymbol{y}$. Thus,

$$\Pr[\delta_t = -1] = \Pr[\boldsymbol{x}_i = 1, \boldsymbol{y}_i = 0]\big(1 - \Delta(\hat{\ell} + 1)\big) + \Pr[\boldsymbol{x}_i = 0, \boldsymbol{y}_i = 1]\Delta(\hat{\ell} - 1) \ .$$

Please note that, formally speaking, the probability above should be a random variable and thus conditional on a filter (or a random variable). However, conditioning on Y_{t-1} is not enough, since this does not entail that the individuals $\boldsymbol{x}$ and $\boldsymbol{y}$ are known. Hence, we actually had to condition onto a filter where both Y_{t-1} as well as the sampled offspring are known. For the sake of brevity, we ignore this detail, treat the probability as a normal probability, and continue to condition only on Y_{t-1} for the drift. Since drift can be used with any filtration, we do not limit applicability this way.

Similarly, $\delta_t = 1$ only in the event that

(a) $\boldsymbol{x}_i = 1$ and $\boldsymbol{y}_i = 0$ but $\boldsymbol{x}$ is *not* accepted, because $\boldsymbol{x}$ and $\boldsymbol{y}$ were misclassified by $\textsc{OneMax}_{[\sigma^2]}$, or

(b) $\boldsymbol{x}_i = 0$ and $\boldsymbol{y}_i = 1$ and $\boldsymbol{x}$ is accepted because $\textsc{OneMax}_{[\sigma^2]}$ ranked $\boldsymbol{x}$ and $\boldsymbol{y}$ correctly.

Thus,

$$\Pr[\delta_t = 1] = \Pr[\boldsymbol{x}_i = 1, \boldsymbol{y}_i = 0]\Delta(\hat{\ell} + 1) + \Pr[\boldsymbol{x}_i = 0, \boldsymbol{y}_i = 1]\big(1 - \Delta(\hat{\ell} - 1)\big) \ .$$

Since $\Pr[\boldsymbol{x}_i = 1, \boldsymbol{y}_i = 0] = \Pr[\boldsymbol{x}_i = 0, \boldsymbol{y}_i = 1] = \Pr[\boldsymbol{x}_i \neq \boldsymbol{y}_i]/2$, we get

$$\begin{aligned} \mathrm{E}[\delta_t \mid Y_{t-1}; \hat{\ell} \neq 0] &= \Pr[\delta_t = 1] - \Pr[\delta_t = -1] \\ &= -\Pr[\boldsymbol{x}_i \neq \boldsymbol{y}_i]\big(\Delta(\hat{\ell} - 1) - \Delta(\hat{\ell} + 1)\big) \\ &< 0 \ , \end{aligned}$$

where we apply Lemma 7.6. We conclude that in this case,

$$\begin{aligned}\mathrm{E}[Y_t \mid Y_{t-1}; \hat{\ell} \neq 0] &= Y_{t-1} + \mathrm{E}[\delta_t \mid Y_{t-1}; \hat{\ell} \neq 0] \\ &< Y_{t-1} \ .\end{aligned}$$

Case $\hat{\ell} = 0$. In this case, if $\boldsymbol{x}_i = \boldsymbol{y}_i$, then $\boldsymbol{x} = \boldsymbol{y}$ and there is zero drift. Otherwise, if $\boldsymbol{x}_i > \boldsymbol{y}_i$, then $\|\boldsymbol{x}\|_1 - \|\boldsymbol{y}\|_1 = 1$, or if $\boldsymbol{y}_i > \boldsymbol{x}_i$, then $\|\boldsymbol{y}\|_1 - \|\boldsymbol{x}\|_1 = 1$. The drift in this case only depends on whether or not $\textsc{OneMax}_{[\sigma^2]}$ misclassifies $\boldsymbol{x}$ and $\boldsymbol{y}$. In particular,

$$\begin{aligned}\Pr[\delta_t = -1] &= \Pr[\boldsymbol{x}_i = 1, \boldsymbol{y}_i = 0](1 - \Delta(1)) + \Pr[\boldsymbol{x}_i = 0, \boldsymbol{y}_i = 1]\big(1 - \Delta(1)\big) \quad \text{and} \\ \Pr[\delta_t = 1] &= \Pr[\boldsymbol{x}_i = 1, \boldsymbol{y}_i = 0]\Delta(1) + \Pr[\boldsymbol{x}_1 = 0, \boldsymbol{y}_i = 1]\Delta(1) \ .\end{aligned}$$

By Lemma 7.6, there is a constant $c > 0$ such that

$$\begin{aligned}\mathrm{E}[\delta_t \mid Y_{t-1}; \hat{\ell} \neq 0] &= \Pr[\delta_t = 1] - \Pr[\delta_t = -1] \\ &= -\Pr[\boldsymbol{x}_i \neq \boldsymbol{y}_i]\big(1 - 2\Delta(1)\big) \\ &\leq -\frac{c}{\sigma^2}\Pr[\boldsymbol{x}_i \neq \boldsymbol{y}_i] \ .\end{aligned}$$

Hence, for this case, we get

$$\begin{aligned}\mathrm{E}[Y_t \mid Y_{t-1}; \hat{\ell} = 0] &= Y_{t-1} + \mathrm{E}[\delta_t \mid Y_{t-1}; \hat{\ell} \neq 0] \\ &\leq Y_{t-1} - \frac{c}{\sigma^2}\Pr[\boldsymbol{x}_i \neq \boldsymbol{y}_i] \ .\end{aligned}$$

Applying the law of total expectation, we see that

$$\mathrm{E}[Y_t \mid Y_{t-1}] \leq Y_t - \frac{c}{\sigma^2}\Pr[\boldsymbol{x}_i \neq \boldsymbol{y}_i]\Pr[\hat{\ell} = 0] \ .$$

It remains to bound $\Pr[\hat{\ell} = 0] = \Pr[\|\hat{\boldsymbol{x}}\|_1 = \|\hat{\boldsymbol{y}}\|_1]$. We define the random variable $Z = \sum_{j=2}^{n} Z_j$, where

$$Z_j = \begin{cases} 1 & \text{if } \boldsymbol{x}_j > \boldsymbol{y}_j \ , \\ 0 & \text{if } \boldsymbol{x}_j = \boldsymbol{y}_j \ , \text{and} \\ -1 & \text{if } \boldsymbol{x}_j < \boldsymbol{y}_j \ . \end{cases}$$

Thus, $\Pr[\|\hat{\boldsymbol{x}}\|_1 = \|\hat{\boldsymbol{y}}\|_1] = \Pr[Z = 0] \geq 1/(4\sqrt{n-1})$ by Lemma 7.8, since we consider an iteration $t < T'$ and hence all frequencies are at least a. This proves the claim in equation (7.9).

Note that in any iteration, if $\boldsymbol{x}_i = \boldsymbol{y}_i$, then $Y_t = Y_{t-1}$. Thus, for an estimate of the upper bound of T, we can ignore self-loops in the process.

More formally, let $(\hat{Y}_t)_{t\in\mathbf{N}}$ be the restriction of Y_t to iterations such that $Y_t \neq Y_{t-1}$. Similarly, let $\hat{T} = \inf\{t \in \mathbf{N} \mid \hat{Y}_t > (1/2 - a)K\}$. The random variable T stochastically dominates the random variable $\hat{T}$, since removing equal moves can only make the process hit faster, that is, $\forall t \in \mathbf{N}\colon \Pr[T > t] \geq \Pr\big[\hat{T} > t\big]$. Due to the above arguments, we see that there is a constant $c' > 0$ such that

$$\begin{aligned} \mathrm{E}\big[\hat{Y}_t \bigm| \hat{Y}_{t-1}\big] &= \mathrm{E}\big[Y_t \bigm| \hat{Y}_{t-1}; \boldsymbol{x}_i \neq \boldsymbol{y}_i\big] \\ &= \hat{Y}_t - \mathrm{E}\big[\delta_t \bigm| \hat{Y}_{t-1}; \boldsymbol{x}_i \neq \boldsymbol{y}_i\big] \\ &\leq \hat{Y}_t - \frac{c}{\sigma^2\sqrt{n}}, \end{aligned}$$

since conditioning on $\hat{Y}_{t-1}$ assures that there is a $t' < T$ such that equation (7.9) holds for iteration t'.

By Corollary 3.24 with $\varepsilon \in -\Theta\big(\sigma^{-2}/\sqrt{n}\big)$ and noting that $Y_0 = \hat{Y}_0 = 0$ and $|\hat{Y}_t - \hat{Y}_{t+1}| = 1 < \sqrt{2}$, we get for all $t \in \mathbf{N}$ that

$$\begin{aligned} \Pr[T \leq t] &\leq \Pr\big[\hat{T} \leq t\big] \\ &\leq t^2 \cdot \mathrm{e}^{-\frac{\left(\frac{1}{2}-a\right)K|\varepsilon|}{4}}. \end{aligned}$$

Since we assume that $K \in \omega\big(\sigma^2\sqrt{n}\log(n)\big)$, there exists a $d \in \omega(1)$ such that $\Pr[T \leq t] = t^2 n^{-d}$. Thus, for any polynomial t, with probability superpolynomially close to 1, Y has not yet reached a state larger than $(1/2 - a)K$, and so, for all $0 \leq t' \leq t$, we get $\boldsymbol{p}_i^{(t')} > a$. As this holds for every position i, applying a union bound retains a superpolynomially small probability that any of the n frequencies has gone below a within polynomially many steps. ■

It is now straightforward to prove that the optimization time of the cGA is polynomial in the problem size and the noise variance. This means that the cGA scales gracefully with Gaussian noise. We actually prove a stronger condition than the first-hitting time of the true optimal solution: we show that after polynomial time, the frequencies have converged to the optimal frequency distribution ($\boldsymbol{p}_i = 1$ for all $i \in [n]$). The proof is carried out by first conditioning on the event that no frequency gets too small. This event is guaranteed with high probability by Lemma 7.10. The result then follows by showing that the drift of the sum of frequencies is large enough in order to hit the optimal distribution within the claimed time bound.

▶ **Theorem 7.11.** Consider the cGA optimizing $\text{OneMax}_{[\sigma^2]}$ with variance $\sigma^2 > 0$. If $K \in \omega(\sigma^2\sqrt{n}\log(n))$, then with high probability, the cGA has converged to the optimal frequency vector after $\mathrm{O}(K\sigma^2\sqrt{n}\log(Kn))$ iterations. ◀

Proof. We consider the drift of the stochastic process $(X_t)_{t\in\mathbf{N}}$ as defined in equation (7.1). This is a stochastic process over the state space $S = \bigcup_{i=0}^{nK}\{n-i/K\}$. Hence, the bounds needed for Theorem 7.3 are $x_{\min} = 1/K$ and $x_{\max} = n$.

Let $a \in (0, 1/2)$ be a constant. We say the process has *failed* by time t if there exists some $t' \leq t$ and some $i \in [n]$ such that $\boldsymbol{p}_i^{(t')} \leq a$. Let $T = \inf\{t \in \mathbf{N} \mid X_t = 0\}$.

Assuming that the process never fails, by Lemma 7.9, the drift of X in each step is bounded by $\mathrm{E}[X_t - X_{t+1} \mid X_t] \geq \delta X_t$, where $1/\delta \in \mathrm{O}(\sigma^2 K\sqrt{n})$. By Theorem 7.3, for any $\lambda > 0$, it holds that $\Pr\left[T > \left(\ln(X_0/x_{\min}) + \lambda\right)/\delta \,\middle|\, X_0\right] \leq \mathrm{e}^{-\lambda}$. Choosing $\lambda = d\ln(n)$ for any constant $d > 0$, the probability that $T \in \Omega(K\sigma^2\sqrt{n}\log(Kn))$ is at most n^{-d}.

Let $\mathcal{E}$ be the event that the process has not failed within $\mathrm{O}(K\sigma^2\sqrt{n}\log(Kn))$ iterations. By the law of total probability (Theorem 2.1), the probability of X reaching 0 (corresponding to the optimal frequency vector) within $\mathrm{O}(K\sigma^2\sqrt{n}\log(Kn))$ steps is at least $(1 - n^{-d})\Pr[\mathcal{E}]$, where we apply Lemma 7.10 to bound the probability of $\overline{\mathcal{E}}$. This concludes the proof. ■

7.4 Conclusions

In this chapter, we examined the robustness of the cGA to additive Gaussian noise and showed that it scales gracefully (Theorem 7.11). Intuitively, the cGA uses its frequency vector to average out the noise over many iterations. This is in contrast to population-based algorithms, such as the (μ+1) EA that fail if the variance is too large [Fri+17]. The intuitive reason for their failing is that the probability of generating and accepting a worse individual becomes larger than the probability of generating and accepting a better individual, as mutation has a bias towards bit strings with about as many 0s as 1s, and, for high noise, the probability of accepting slightly worse individuals is about 1/2.

A similar technique for coping with noise to what the cGA inherently displays is *resampling*. In this strategy, when optimizing a noisy objective function, an individual is evaluated multiple times, and the average fitness is used. This has the effect of explicitly reducing the variance of the noise. However, resampling comes at the extra cost of potentially many more function evaluations. Akimoto et al. [AAT15] explicitly studied the effect of resampling on various noise models to derive the extra cost incurred by performing enough resampling to ensure that the underlying optimization algorithm sees an almost noiseless function.

Both of the above approaches – EDAs and resampling – have the problem that the parameters of the algorithm (either the step size or the number of resamples) have to be chosen with respect to the variance of the noise. Since this value is usually unknown, the approaches by itself are not very useful. Hence, in the journal version that this chapter is based on [Fri+17], we propose an algorithm scheme that starts with a variance of 1 and restarts the underlying algorithm with twice the previous variance if the prior run was deemed to have failed, due to the prior calculated expected run time. This scheme is called *noise-oblivious,* since the variance is not part of the input anymore. Experimental results in the paper show that the cGA outperforms the RLS and the $(\mu+1)$ EA (the latter two both using an appropriate amount of resamples) regardless of whether a noise-oblivious scheme is used or not.

Note that a noise-oblivious scheme still has the drawback that a bound on the expected run time of the underlying algorithm is required in order to know when to restart. Thus, this approach is not applicable in most scenarios. A better strategy would be to consider multiple runs for different choices of the variance. In each iteration, one parameter setting is chosen according to some distribution (for example, a power law distribution) and then the corresponding algorithm is advanced by a constant number of steps. This way, each parameter setting has a chance of being chosen, and no prior bound on the run time for a setting is required. However, rigorous analyzes are needed in order to assess the validity of this approach.

While all of our results considered first-hitting times, one can also look at these problems from the perspective of a fixed time budget [JZ14]. We believe that similar results can be obtained in this setting. Further, we conjecture that EAs that explicitly use recombination can also scale gracefully. A step in that direction was made by Prügel-Bennett et al. [PRS15], who proved that a generational EA using uniform crossover needs only $\mathrm{O}\big(n\log(n)^2\big)$ function evaluations to optimize OneMax with an additive noise of variance $\sigma^2 = n$.

8 Upper Bound of the MMAS-fp on Noisy Linear Functions

This chapter is based on joint work with Tobias Friedrich, Timo Kötzing, and Andrew M. Sutton [Fri+16]. We only consider posterior noise and not prior noise, in order for this chapter to be more similar to Chapter 7. Further, this chapter fixes some inaccuracies of the journal version.

In this chapter, we consider the same noise setting as in Chapter 7, and we show that the MMAS-fp (see Section 8.2.1) scales gracefully (Definition 7.1) with additive Gaussian noise on linear functions (Theorem 8.4), which is very similar to our result obtained for the cGA on OneMax (Theorem 7.11). Further, we show that the MMAS-fp also successfully optimizes linear functions when the noise is bounded within a polynomial interval with very high probability (Corollary 8.5).

8.1 Introduction

Ant colony optimization (ACO; [DS04]) is a meta-heuristic for designing randomized general-purpose optimization algorithms inspired by the foraging behavior of ant colonies, and it has been successfully applied as a heuristic technique for solving combinatorial optimization problems.

As we already discussed in Section 7.1, in real-world optimization problems, there is sometimes a large degree of uncertainty present due to the complexity of candidate solution generation, noisy measurement processes, and rapidly changing problem environments. Empirically, ACO seems particularly well-suited to uncertain problems due to its dynamic and distributed nature, and in some cases it can outperform classical state-of-the-art approaches on dynamic network routing problems [CDG08]. We focus on a version of the *Max-Min Ant System* (MMAS Stützle and Hoos [SH00]) applied to pseudo-Boolean optimization.

In order to address these practical issues, the theoretical analyses of randomized search heuristics under uncertainty has recently gained momentum. For example, a number of recent papers rigorously analyzed the performance of evolutionary algorithms in stochastic environments [BQT18; DL16; GK16]. For ant colony optimization, a series of papers considered the performance of ACO on single destination shortest paths (SDSP) problems with stochastic weights. This work was initiated by Sudholt and Thyssen [ST12] and later followed up by Doerr et al. [DHK12], who showed that by augmenting the ant system with a re-evaluation strategy on the best-so-far solution, many of the difficulties with noise discovered in [ST12] could be overcome. Feldmann and Kötzing [FK13] showed that an ant system that uses a fitness-proportional update rule (called MMAS-fp) can efficiently optimize SDSP on graphs with stochastic weights. The MMAS-fp is closer to systems that are used by practitioners [SH00] and is the ant system variant that we analyze in this chapter.

For the optimization of functions over bit strings, analyses of ACO suggest that it often performs worse than evolutionary algorithms and simple hill-climbers in a noise-free setting Kötzing et al. [Köt+11]. On the other hand, ACO can outperform evolutionary algorithms on dynamic problems [KM12; LW16]. So far, the question of how robust ACO is to noisy evaluation on pseudo-Boolean optimization remains unanswered.

Our goal is to observe the robustness of ACO to noise on a class of simple objective functions. In particular, we are interested in whether the MMAS-fp scales gracefully (Definition 7.1). In contrast to Chapter 7, we do not only consider OneMax but linear functions in general, that is, functions with a predefined sequence of weights $\boldsymbol{w}$ that map a bit string $\boldsymbol{x}$ to $\sum_{i\in[n]} \boldsymbol{x}_i\boldsymbol{w}_i$. Note that if we set all weights to 1, we recover OneMax.

The main result of this chapter is given by Theorem 8.4, where we show that the MMAS-fp scales gracefully with noise from a Gaussian distribution. Further, we extend our findings to other noise models and show that we achieve the same robustness also in the presence of other noise distributions (Corollary 8.5), where the noise is bounded by a polynomial with very high probability.

8.2 Preliminaries

The fitness functions we investigate are linear, following the notation of Droste et al. [DJW02]. That is, given a vector $\boldsymbol{w}$ of length n of *positive* weights, we consider functions $f(\boldsymbol{x}) = \sum_{i\in[n]} \boldsymbol{w}_i\boldsymbol{x}_i$. Thus, the all-1s string is the global optimum. We discuss why we assume positive weights in Section 8.2.1.

For a given weight vector $\boldsymbol{w}$, we define $w_{\min}$ to be its minimum weight, $w_{\max}$ to be its maximum weight, and $W = \sum_{i\in[n]} \boldsymbol{w}_i$ to be the sum of all weights. Further, for an individual $\boldsymbol{x} \in \{0,1\}^n$, let $\|w(\boldsymbol{x})\|_0$ denote the sum of weights $\boldsymbol{w}_i$ with $\boldsymbol{x}_i = 0$, and let $\|w(\boldsymbol{x})\|_1$ be defined analogously for weights of 1-bits. Note that, for all $\boldsymbol{x} \in \{0,1\}^n$, we have $W = \|w(\boldsymbol{x})\|_0 + \|w(\boldsymbol{x})\|_1$.

Throughout this chapter, we consider the same noisy setting as described in Section 7.2. That is, for a given deterministic fitness function f and a Gaussian random variable $D \sim \mathrm{N}(0, \sigma^2)$, we define the noisy variant as $f_{[\sigma^2]}(\boldsymbol{x}) \coloneqq f(\boldsymbol{x}) + D$. We also mainly make use of the theorems mentioned Section 7.2.

Last, we say that an event A occurs with *very high probability* if and only if, for all $p(n) \in \mathrm{O}\big(\mathrm{poly}(n)\big)$, it holds that $\Pr\big[\overline{A}\big] \le 1/p(n)$, that is, the probability of $\overline{A}$ is superpolynomially small.

8.2.1 MMAS-fp

Our algorithm of interest is the MMAS-fp, an ACO algorithm with a so-called *fitness-proportional* update rule. It is related to the λ-MMAS$_{\mathrm{IB}}$ (Section 4.3.3) and thus also an n-Bernoulli-λ-EDA (Algorithm 1). The main difference is that the MMAS-fp only samples a single individual $\boldsymbol{x}$ per iteration and then updates its frequencies with respect to the relative quality of $\boldsymbol{x}$. In order to simplify the calculations in this chapter, we assume that a frequency denotes the probability to sample a 0 (instead of a 1). For all $i \in [n]$, the update scheme of the MMAS-fp then is

$$\varphi\big(\boldsymbol{p}, \big(\boldsymbol{x}, f(\boldsymbol{x})\big)_{\boldsymbol{x} \in D}\big)_i = \begin{cases} \Big(1 - \rho\frac{f(\boldsymbol{x}^{(1)})}{W}\Big)\boldsymbol{p}_i + \rho\frac{f(\boldsymbol{x}^{(1)})}{W} & \text{if } \boldsymbol{x}_i^{(1)} = 0\,, \\ \Big(1 - \rho\frac{f(\boldsymbol{x}^{(1)})}{W}\Big)\boldsymbol{p}_i & \text{if } \boldsymbol{x}_i^{(1)} = 1\,. \end{cases}$$

The MMAS-fp is – different than the λ-MMAS$_{\mathrm{IB}}$ – *not* locally updating, since the update makes use of the fitness of its offspring $\boldsymbol{x}$, which may vary. However, the MMAS-fp is still unbiased, since the proof of Theorem 5.11 basically applies here too. Hence, switching the roles of 0s and 1s does not change the applicability of our results. Further, we consider the MMAS-fp with and without a margin (see Section 4.2.2). If we use a margin, we call it b.

Note that the update scheme of the MMAS-fp only works properly if we can guarantee that $0 \le \rho f(\boldsymbol{x})/W \le 1$. Hence, we assume that our (unnoisy) linear fitness functions have nonnegative weights in order for $\rho f(\boldsymbol{x})/W \ge 0$ to hold. Further note that we assume that the MMAS-fp has access to W. In reality, one either has to have an upper bound on W or choose ρ accordingly in order to guarantee that $\rho f(\boldsymbol{x})/W \leq 1$. However, since f is perturbed by unbounded noise in our setting, it is always possible that $\rho f(\boldsymbol{x})/W > 1$ or $\rho f(\boldsymbol{x})/W < 0$. If this happens and a frequency leaves the interval $[0, 1]$ (or, for a margin b, the interval $[b, 1 - b]$), we assume that the respective frequency is capped according to equation (4.2).

8.3 Formal Analysis

In this section, we bound the run time of the MMAS-fp for optimizing a noisy linear function $f_{[\sigma^2]}$, which we denote as f, for convenience, in the proofs but not in the statements of the theorems.

First, we bound the range of the noise with very high probability and thus the difference of two consecutive frequencies. Since the noise is unbounded, it can always happen that this bound is exceeded during an update. By bounding the noise with very high probability, we make sure that such updates happen only very rarely. We then choose ρ such that it can compensate for likely noise levels.

▶ **Lemma 8.1.** Consider the MMAS-fp (with or without a margin) with any ρ optimizing $f_{[\sigma^2]}$. Let $m \in \omega\big(\sigma\sqrt{\log(n)}\big)$. Then, for all positions $i \in [n]$ and all time points $t \in \mathbf{N}$, it holds with very high probability that

$$|\boldsymbol{p}_i^{(t)} - \boldsymbol{p}_i^{(t+1)}| \leq \rho\frac{W + m}{W} . \quad ◀$$

Proof. Let D denote the noise of a fitness evaluation. We now bound $|f(\boldsymbol{x})| = |W + D| \leq W + m$ with high very probability. By the symmetry of D around 0 and by Lemma 7.2, we get for any $r \in \mathbf{R}$ that

$$\begin{aligned}\Pr[D > r] &= \Pr[D < -r] \\ &\leq \frac{1}{2}\mathrm{e}^{-\frac{r^2}{2\sigma^2}} .\end{aligned}$$

For $r = m \in \omega\big(\sigma\sqrt{\log(n)}\big)$, this probability is superpolynomially small. Hence, we can bound $|f(\boldsymbol{x})| \leq W + m$ with very high probability.

We now make a case distinction with respect to whether $\boldsymbol{p}_i^{(t)}$ is increased or decreased. We assume no margin, since capping a frequency only makes the difference smaller.

For the case $\boldsymbol{p}_i^{(t+1)} = \left(1 - \rho\frac{f(\boldsymbol{x})}{W}\right)\boldsymbol{p}_i^{(t)}$, we get

$$\begin{aligned}
|\boldsymbol{p}_i^{(t)} - \boldsymbol{p}_i^{(t+1)}| &= \left|\boldsymbol{p}_i^{(t)} - \left(1 - \rho\frac{f(\boldsymbol{x})}{W}\right)\boldsymbol{p}_i^{(t)}\right| \\
&= \rho\frac{|f(\boldsymbol{x})\, \overbrace{\boldsymbol{p}_i^{(t)}}^{\leq 1}\, |}{W} \\
&\leq \rho\frac{|f(\boldsymbol{x})|}{W} \ .
\end{aligned}$$

For the case $\boldsymbol{p}_i^{(t+1)} = \left(1 - \rho\frac{f(\boldsymbol{x})}{W}\right)\boldsymbol{p}_i^{(t)} + \rho\frac{f(\boldsymbol{x})}{W}$, we get

$$\begin{aligned}
|\boldsymbol{p}_i^{(t)} - \boldsymbol{p}_i^{(t+1)}| &= \left|\boldsymbol{p}_i^{(t)} - \left(\left(1 - \rho\frac{f(\boldsymbol{x})}{W}\right)\boldsymbol{p}_i^{(t)} + \rho\frac{f(\boldsymbol{x})}{W}\right)\right| \\
&= \rho\frac{|f(\boldsymbol{x})\overbrace{(\boldsymbol{p}_i^{(t)} - 1)}^{\geq -1}|}{W} \\
&\leq \rho\frac{|f(\boldsymbol{x})|}{W} \ .
\end{aligned}$$

Bounding $|f(\boldsymbol{x})| \leq W + m$ with very high probability completes the proof. ■

From now on, we always condition on the event that the noise does not exceed the bounds mentioned in the proof of Lemma 8.1, which holds with very high probability in any polynomial number of steps of the MMAS-fp for all of the n frequencies via a union bound. We refer to other runs as *fails*. Due to that, all of the following theorems only hold with very high probability.

Further, we always assume from now on, for any $m \in \omega\big(\sigma\sqrt{\log(n)}\big)$, that we choose $\rho \in \mathrm{o}\big(W/(W+m)\big)$ such that $\rho f(\boldsymbol{x})/W \in \mathrm{o}(1)$ holds for all $\boldsymbol{x}$ with very high probability. Let $s := \rho(W+m)/W \in \mathrm{o}(1)$ then denote the greatest difference possible for two consecutive frequency values (with very high probability). If an update changes a frequency by at most s, we call it *normal.*

Note that the noise can also be negative and, thus, we lower-bound $\rho f(\boldsymbol{x})/W \geq -\rho(W+m)/W \geq -s$. In such a case, a decreasing update of a frequency $\boldsymbol{p}_i$ is an increase, and the resulting frequency will be greater than 1 if $\boldsymbol{p}_i > 1/(1+s) = 1-s/(1+s) > 1-cs$ for a constant $c > 0$. Conversely, an increasing update is then a decrease, and the resulting frequency will be less than 0 if $\boldsymbol{p}_i < s/(1+s) \in \mathrm{O}(s)$. Hence, a normal update can still lead to a frequency leaving the interval $[0, 1]$ (or, for a margin b, the interval $[b, 1-b]$). If this happens, the resulting frequency is set to the respective bound, which we still consider normal and *not* a failing run. The reasoning behind this consideration is that the frequency is so close to a border that the wrong update does not matter.

We aim to show that with very high probability a frequency will not reach a constant value greater than $1/2$ within polynomial time during a non-failing run of the MMAS-fp. For this, we show that there is a negative drift for every frequency toward 1.

▶ **Lemma 8.2.** Let $m \in \omega\big(\sigma\sqrt{\log(n)}\big)$, and let $d \in (1/2, 1)$ be a constant. Consider the MMAS-fp (without a margin or with a margin $b \leq 1-d$) with $\rho \in \mathrm{o}\big(W w_{\min}/\big((W+m)^2 \log(n)\big)\big)$ optimizing $f_{[\sigma^2]}$. Then, with very high probability, no frequency reaches a value of at least d in polynomial time. ◀

Proof. Note that, due to our assumptions on ρ, we have that $s \in \mathrm{o}(1)$. Further, we assume a non-failing run of the MMAS-fp. Thus, all following statements hold with very high probability.

We first show for any position $i \in [n]$ that the respective frequency $\boldsymbol{p}_i$ does not reach d within polynomial time very likely by using Corollary 3.24. Afterward, we apply a union bound (Theorem 2.17), which then concludes the proof.

Since $\boldsymbol{p}_i^{(0)} = 1/2$ and Corollary 3.24 requires the first value to be at most 0, we consider the process $(X_t)_{t \in \mathbf{N}}$ with $X_t = \boldsymbol{p}_i^{(t)} - 1/2$. We now show that X has negative drift over the interval $[-s, d-1/2)$. Thus, let $t \in \mathbf{N}$ be such that $-s \leq X_t < d - 1/2$, and let A denote the event that $\boldsymbol{x}_i$ was sampled as a 1. We get

$$
\begin{aligned}
&\mathrm{E}\Big[X_{t+1} - X_t \,\Big|\, \boldsymbol{p}_i^{(t)}\Big] \\
&\quad = \mathrm{E}\Big[\boldsymbol{p}_i^{(t+1)} - \boldsymbol{p}_i^{(t)} \,\Big|\, \boldsymbol{p}_i^{(t)}\Big] \\
&\quad = \mathrm{E}\bigg[\bigg(1 - \rho\frac{f(\boldsymbol{x})}{W}\bigg)\boldsymbol{p}_i^{(t)} \,\bigg|\, \boldsymbol{p}_i^{(t)}\bigg] + \mathrm{E}\bigg[\rho\frac{f(\boldsymbol{x})}{W} \,\bigg|\, \overline{A}; \boldsymbol{p}_i^{(t)}\bigg] \overbrace{\Pr\big[\overline{A} \,\big|\, \boldsymbol{p}_i^{(t)}\big]}^{=\boldsymbol{p}_i^{(t)}} - \boldsymbol{p}_i^{(t)} \\
&\quad = \frac{\rho}{W}\Big(\mathrm{E}\Big[f(\boldsymbol{x}) \,\Big|\, \overline{A}; \boldsymbol{p}_i^{(t)}\Big] - \mathrm{E}\Big[f(\boldsymbol{x}) \,\Big|\, \boldsymbol{p}_i^{(t)}\Big]\Big)\boldsymbol{p}_i^{(t)}
\end{aligned}
$$

$$\begin{aligned}
&= \frac{\rho}{W}\left(\mathrm{E}\left[W - \|w(\boldsymbol{x})\|_0 + D \,\middle|\, \overline{A}; \boldsymbol{p}_i^{(t)}\right] - \mathrm{E}\left[W - \|w(\boldsymbol{x})\|_0 + D \,\middle|\, \boldsymbol{p}_i^{(t)}\right]\right)\boldsymbol{p}_i^{(t)}\\
&= \frac{\rho}{W}\left(W - \left(\boldsymbol{w}_i + \sum_{\substack{j\in[n]\\ j\neq i}} \boldsymbol{w}_j\boldsymbol{p}_j^{(t)}\right) - \left(W - \left(\boldsymbol{w}_i\boldsymbol{p}_i^{(t)} + \sum_{\substack{j\in[n]\\ j\neq i}} \boldsymbol{w}_j\boldsymbol{p}_j^{(t)}\right)\right)\right)\boldsymbol{p}_i^{(t)}\\
&= \boldsymbol{p}_i^{(t)}(\boldsymbol{p}_i^{(t)} - 1)\boldsymbol{w}_i\frac{\rho}{W}\\
&\leq -\boldsymbol{p}_i^{(t)}(1 - \boldsymbol{p}_i^{(t)})w_{\min}\frac{\rho}{W}\\
&\leq -d(1-d)w_{\min}\frac{\rho}{W}\ ,
\end{aligned}$$

where the last inequality used that $-s \leq X_t < d - 1/2$ and that the concave function $\boldsymbol{p}_i^{(t)}(1 - \boldsymbol{p}_i^{(t)})$ is minimal over the interval $[1/2 - s, d]$ for $\boldsymbol{p}_i^{(t)} = d$.

Since we assume a non-failing run, we can use Lemma 8.1 in order to strictly upper-bound the maximum difference of an update by $\sqrt{2}s$. Further note that any $X_t < -s$ can be at most 0 after an update. Hence, we can apply Corollary 3.24 and see that there is a constant $c > 0$ such that

$$\begin{aligned}
\Pr[T \leq t] &\leq t^2 \exp\left(-\frac{\Big(\overbrace{d - \tfrac{1}{2}}^{>0}\Big)d(1-d)w_{\min}\frac{\rho}{W}}{4\frac{\rho^2}{W^2}(W+m)^2}\right)\\
&\leq t^2 \exp\left(-\frac{cWw_{\min}}{\rho(W+m)^2}\right).
\end{aligned}$$

Due to our assumption $\rho \in \mathrm{o}\big(Ww_{\min}/\big((W+m)^2\log(n)\big)\big)$, the exponential function in the bound above is superpolynomially small. Hence, the probability that $\boldsymbol{p}_i$ reaches values of at least d within polynomial time is superpolynomially small.

As stated at the beginning of the proof, applying a union bound over all of the n frequencies yields that no frequency reaches a value of at least d within polynomial time with very high probability. And since the probability of a run being non-failing is very high according to Lemma 8.1, the probability of that event still remains very high, which concludes the proof. ■

We now consider the sum of all frequencies, which we call p_{total}, and lower-bound its drift toward 0. Note that if we assume that the MMAS-fp has a margin, then the drift cannot be positive if the frequencies are too close to the lower border – in the extreme case that all frequencies are at the lower border, the drift is negative. However, if the margin is sufficiently small and there is at least on frequency that is sufficiently large, the drift of this frequency outweighs the (possibly negative) drift of the other frequencies that are close to the lower border.

▶ **Lemma 8.3.** Let $m \in \omega\big(\sigma\sqrt{\log(n)}\big)$. Consider the MMAS-fp optimizing $f_{[\sigma^2]}$ with no margin or a margin $b \in \mathrm{o}\big(w_{\min}/(w_{\max}n^3)\big)$ and with $\rho \in \mathrm{o}\big(Ww_{\min}/((W+m)^2\log(n))\big) \cap \mathrm{o}\big(W/((W+m)n)\big)$. If at least one frequency is at least $1/n$, then the drift of the sum of all frequencies p_{total} toward 0 is with very high probability in $\Omega(p_{\text{total}}\rho w_{\min}/W)$. ◀

Proof. We assume a non-failing run and that no frequency reaches a value of at least a constant value $d \in (1/2, 1)$. Due to Lemmas 8.1 and 8.2, all of the following statements then hold with very high probability. Further, we pessimistically assume frequencies below $1/n$ as having reached b. Note that $b \in \mathrm{o}(1/n)$ and $s = \rho(W+m)/W \in \mathrm{o}(1/n)$. Hence, if a frequency has a value of at least $1/n$, it is normally updated.

Consider an iteration t such that $k \in [0..n-1]$ frequencies are below $1/n$. We split the overall drift of the sum of all frequencies into the (positive) drift of the $n-k$ frequencies that are at least $1/n$ and into the (negative) drift of the k frequencies that are below $1/n$. Let Y denote the index set of those former $n-k$ frequencies and Z the index set of the latter k.

For the offspring $\boldsymbol{x}$ sampled by $\boldsymbol{p}^{(t)}$, let $\boldsymbol{x}^Y$ denote the bit string consisting only of those elements of $\boldsymbol{x}$ whose index is in Y, and let $\boldsymbol{x}^Z$ be defined analogously with respect to Z. Further, we denote the sum of frequencies with index in Y by p^Y_{total} and the sum of those with index in Z by p^Z_{total}. Last, let P_{total} denote $\mathrm{E}\big[\|w(\boldsymbol{x})\|_0 \,\big|\, \boldsymbol{p}^{(t)}\big] = \sum_{j\in[n]} \boldsymbol{w}_j\boldsymbol{p}^{(t)}_j$, and let P^Y_{total} and P^Z_{total} be defined analogously.

We now consider the drift of the $n-k$ frequencies with an index in Y. We mark the updated frequency vector by a prime and get

$$\begin{aligned}\mathrm{E}\Big[p^Y_{\text{total}} - \big(p^Y_{\text{total}}\big)' \,\Big|\, \boldsymbol{p}^{(t)}\Big] &= \mathrm{E}\left[p^Y_{\text{total}} - \left(p^Y_{\text{total}}\left(1-\rho\frac{f(\boldsymbol{x})}{W}\right) + \|\boldsymbol{x}^Y\|_0\rho\frac{f(\boldsymbol{x})}{W}\right) \,\middle|\, \boldsymbol{p}^{(t)}\right]\\ &= \frac{\rho}{W}\Big(p^Y_{\text{total}}\mathrm{E}\Big[f(\boldsymbol{x}) \,\Big|\, \boldsymbol{p}^{(t)}\Big] - \mathrm{E}\big[\|\boldsymbol{x}^Y\|_0 f(\boldsymbol{x}) \,\big|\, \boldsymbol{p}^{(t)}\big]\Big)\ ,\end{aligned}$$

where

$$\begin{aligned}\mathrm{E}\left[f(\boldsymbol{x})\,\middle|\,\boldsymbol{p}^{(t)}\right] &= \mathrm{E}\left[W - \|w(\boldsymbol{x})\|_0 + D \,\middle|\, \boldsymbol{p}^{(t)}\right] \\ &= W - P_{\text{total}}\ ,\end{aligned}$$

and

$$\begin{aligned}\mathrm{E}\left[\|\boldsymbol{x}^Y\|_0 f(\boldsymbol{x}) \,\middle|\, \boldsymbol{p}^{(t)}\right] &= \mathrm{E}\left[\|\boldsymbol{x}^Y\|_0 (W - \|w(\boldsymbol{x})\|_0 + D) \,\middle|\, \boldsymbol{p}^{(t)}\right] \\ &= W\mathrm{E}\left[\|\boldsymbol{x}^Y\|_0 \,\middle|\, \boldsymbol{p}^{(t)}\right] - \mathrm{E}\left[\|\boldsymbol{x}^Y\|_0 \cdot \|w(\boldsymbol{x})\|_0 \,\middle|\, \boldsymbol{p}^{(t)}\right] \\ &= W p^Y_{\text{total}} - \mathrm{E}\left[\|\boldsymbol{x}^Y\|_0 \cdot \|w(\boldsymbol{x})\|_0 \,\middle|\, \boldsymbol{p}^{(t)}\right]\ .\end{aligned}$$

We further bound, noting that the bits of $\boldsymbol{x}$ are sampled independently from another,

$$\begin{aligned}\mathrm{E}\left[\|\boldsymbol{x}^Y\|_0 \cdot \|w(\boldsymbol{x})\|_0 \,\middle|\, \boldsymbol{p}^{(t)}\right] &= \mathrm{E}\left[\sum_{a\in Y}\sum_{j\in[n]} \mathbf{1}\{\boldsymbol{x}_a = 0\}\boldsymbol{w}_j \cdot \mathbf{1}\{\boldsymbol{x}_j = 0\} \,\middle|\, \boldsymbol{p}^{(t)}\right] \\ &= \sum_{a\in Y}\sum_{j\in[n]} \boldsymbol{w}_j \mathrm{E}\left[\mathbf{1}\{\boldsymbol{x}_a = 0\} \cdot \mathbf{1}\{\boldsymbol{x}_j = 0\} \,\middle|\, \boldsymbol{p}^{(t)}\right] \\ &= \sum_{a\in Y}\sum_{\substack{j\in[n]\\ j\neq a}} \boldsymbol{w}_j \mathrm{E}\left[\mathbf{1}\{\boldsymbol{x}_a = 0\} \,\middle|\, \boldsymbol{p}^{(t)}\right]\mathrm{E}\left[\mathbf{1}\{\boldsymbol{x}_j = 0\} \,\middle|\, \boldsymbol{p}^{(t)}\right] \\ &\quad + \sum_{a\in Y} \boldsymbol{w}_a \mathrm{E}\left[\mathbf{1}\{\boldsymbol{x}_a = 0\} \cdot \mathbf{1}\{\boldsymbol{x}_a = 0\} \,\middle|\, \boldsymbol{p}^{(t)}\right] \\ &= \sum_{a\in Y}\sum_{\substack{j\in[n]\\ j\neq a}} \boldsymbol{w}_j \boldsymbol{p}^{(t)}_a \boldsymbol{p}^{(t)}_j + \sum_{a\in Y} \boldsymbol{w}_a \boldsymbol{p}^{(t)}_a \\ &= \sum_{a\in Y} \boldsymbol{p}^{(t)}_a \sum_{j\in[n]} \boldsymbol{w}_j \boldsymbol{p}^{(t)}_j - \sum_{a\in Y} \boldsymbol{w}_a \left(\boldsymbol{p}^{(t)}_a\right)^2 + \sum_{a\in Y} \boldsymbol{w}_a \boldsymbol{p}^{(t)}_a \\ &= \sum_{a\in Y} \boldsymbol{p}^{(t)}_a \sum_{j\in[n]} \boldsymbol{w}_j \boldsymbol{p}^{(t)}_j + \sum_{a\in Y} \boldsymbol{w}_a \boldsymbol{p}^{(t)}_a \left(1 - \overbrace{\boldsymbol{p}^{(t)}_a}^{\leq d}\right) \\ &\geq p^Y_{\text{total}} P_{\text{total}} + (1-d) P^Y_{\text{total}}\ ,\end{aligned}$$

where the last inequality made use of our assumption that no frequency reaches values of at least d.

Thus, we get for the positive drift

$$\begin{aligned}
&\mathrm{E}\Big[p^Y_{\text{total}} - \big(p^Y_{\text{total}}\big)' \,\Big|\, \boldsymbol{p}^{(t)}\Big] \\
&\geq \frac{\rho}{W}\Big(p^Y_{\text{total}}(W - P_{\text{total}}) - \big(Wp^Y_{\text{total}} - p^Y_{\text{total}}P_{\text{total}} - (1-d)P^Y_{\text{total}}\big)\Big) \\
&= \frac{\rho}{W}(1-d)P^Y_{\text{total}} \,.
\end{aligned}$$

We now consider the k frequencies with an index in Z. First, we bound the sum of the updated frequency vector. If a 1 is sampled at a position, the respective frequency cannot be decreased anymore, as we assume that it is at the lower border b. In the other case, the frequency is increased normally.

$$\begin{aligned}
\mathrm{E}\Big[\big(p^Z_{\text{total}}\big)' \,\Big|\, \boldsymbol{p}^{(t)}\Big] &= \mathrm{E}\left[\|\boldsymbol{x}^Z\|_1 b + \|\boldsymbol{x}^Z\|_0\left(\left(1 - \rho\frac{f(\boldsymbol{x})}{W}\right)b + \rho\frac{f(\boldsymbol{x})}{W}\right) \,\middle|\, \boldsymbol{p}^{(t)}\right] \\
&= \mathrm{E}\left[\overbrace{\|\boldsymbol{x}^Z\|_1 b + \|\boldsymbol{x}^Z\|_0 b}^{=kb} - \|\boldsymbol{x}^Z\|_0 b\rho\frac{f(\boldsymbol{x})}{W} + \|\boldsymbol{x}^Z\|_0\rho\frac{f(\boldsymbol{x})}{W} \,\middle|\, \boldsymbol{p}^{(t)}\right] \\
&= kb + \frac{\rho}{W}(1-b)\mathrm{E}\Big[\|\boldsymbol{x}^Z\|_0 f(\boldsymbol{x}) \,\Big|\, \boldsymbol{p}^{(t)}\Big] \,.
\end{aligned}$$

Thus, for the drift of those frequencies, we get

$$\begin{aligned}
\mathrm{E}\Big[p^Z_{\text{total}} - \big(p^Z_{\text{total}}\big)' \,\Big|\, \boldsymbol{p}^{(t)}\Big] &= kb - kb - \frac{\rho}{W}(1-b)\mathrm{E}\Big[\|\boldsymbol{x}^Z\|_0 f(\boldsymbol{x}) \,\Big|\, \boldsymbol{p}^{(t)}\Big] \\
&= -\frac{\rho}{W}(1-b)\mathrm{E}\Big[\|\boldsymbol{x}^Z\|_0 f(\boldsymbol{x}) \,\Big|\, \boldsymbol{p}^{(t)}\Big] \,,
\end{aligned}$$

where

$$\begin{aligned}
\mathrm{E}\Big[\|\boldsymbol{x}^Z\|_0 f(\boldsymbol{x}) \,\Big|\, \boldsymbol{p}^{(t)}\Big] &= \mathrm{E}\Big[\|\boldsymbol{x}^Z\|_0\big(W - \|w(\boldsymbol{x})\|_0 + D\big) \,\Big|\, \boldsymbol{p}^{(t)}\Big] \\
&= W\mathrm{E}\Big[\|\boldsymbol{x}^Z\|_0 \,\Big|\, \boldsymbol{p}^{(t)}\Big] - \mathrm{E}\Big[\|\boldsymbol{x}^Z\|_0 \cdot \|w(\boldsymbol{x})\|_0 \,\Big|\, \boldsymbol{p}^{(t)}\Big] \\
&= Wkb - \mathrm{E}\Big[\|\boldsymbol{x}^Z\|_0 \cdot \|w(\boldsymbol{x})\|_0 \,\Big|\, \boldsymbol{p}^{(t)}\Big] \,.
\end{aligned}$$

The calculations for $\mathrm{E}\big[\|\boldsymbol{x}^Z\|_0 \cdot \|w(\boldsymbol{x})\|_0 \,\big|\, \boldsymbol{p}^{(t)}\big]$ follow analogously to the ones before for the positive drift, and we get

$$\begin{aligned} \mathrm{E}\Big[\|\boldsymbol{x}^Z\|_0 \cdot \|w(\boldsymbol{x})\|_0 \,\Big|\, \boldsymbol{p}^{(t)}\Big] &= \sum_{a\in Z} \boldsymbol{p}_a^{(t)} \sum_{j\in[n]} \boldsymbol{w}_j \boldsymbol{p}_j^{(t)} + \sum_{a\in Z} \boldsymbol{w}_a \boldsymbol{p}_a^{(t)}\Big(1-\boldsymbol{p}_a^{(t)}\Big) \\ &\geq kbP_{\text{total}} + (1-d)P_{\text{total}}^Z \ . \end{aligned}$$

Hence, we get the negative drift

$$\begin{aligned} \mathrm{E}\Big[p_{\text{total}}^Z - \Big(p_{\text{total}}^Z\Big)' \,\Big|\, \boldsymbol{p}^{(t)}\Big] &\geq -\frac{\rho}{W}(1-b)\Big(Wkb - kbP_{\text{total}} - (1-d)P_{\text{total}}^Z\Big) \\ &\geq -\frac{\rho}{W}(1-b)Wkb \ . \end{aligned}$$

Overall, taking together the positive and the negative drift part, we get

$$\begin{aligned} \mathrm{E}\Big[p_{\text{total}} - (p_{\text{total}})' \,\Big|\, \boldsymbol{p}^{(t)}\Big] &\geq \frac{\rho}{W}(1-d)P_{\text{total}}^Y - \frac{\rho}{W}(1-b)Wkb \\ &= \frac{\rho}{W}\big((1-d)P_{\text{total}}^Y - (1-b)Wkb\big) \ , \end{aligned}$$

which we want to be positive.

Recall that we only consider $k \in [0..n-1]$. For the drift to be positive, it suffices to show that $P_{\text{total}}^Y \in \omega(Wkb)$. Using the definition of P_{total}^Y and that all those frequencies are at least $1/n$, we want to have that

$$\overbrace{\sum_{a\in Y} \boldsymbol{w}_a \boldsymbol{p}_a}^{\geq w_{\min}\frac{1}{n}} \in \omega\Bigg(\overbrace{k}^{\leq n-1} \ b \overbrace{\sum_{j\in[n]} \boldsymbol{w}_j}^{\leq n w_{\max}}\Bigg) \ .$$

Hence, we check that the following

$$\frac{w_{\min}}{n} \in \omega\big(n^2 w_{\max} b\big) \ \Leftrightarrow 1 \in \omega\bigg(n^3 \frac{w_{\max}}{w_{\min}} b\bigg) \ ,$$

which is true, because we assume $b \in o\big(w_{\min}/(w_{\max}n^3)\big)$. So we can bound the drift by $\Omega\big(P_{\text{total}}^Y \rho/W\big)$.

Since we want the drift to be expressed in terms of p_{total}, for an $\varepsilon > 0$, we need $P_{\text{total}}^Y \rho/W \geq \varepsilon p_{\text{total}}$ to hold. We bound $P_{\text{total}}^Y \geq w_{\min} p_{\text{total}}^Y$ and $p_{\text{total}} = p_{\text{total}}^Y + k\ell <$

$p^Y_{\text{total}} + k/n$ and get $w_{\min} p^Y_{\text{total}} \rho / W \geq \varepsilon(p^Y_{\text{total}} + k/n)$. This leaves us with

$$\begin{aligned} \varepsilon &\leq \frac{w_{\min} p^Y_{\text{total}} \rho}{W(p^Y_{\text{total}} + \frac{k}{n})} \\ &= \frac{w_{\min} \rho}{W(1 + \frac{k}{p^Y_{\text{total}} n})} \\ &\in \mathrm{O}\Big(\frac{w_{\min} \rho}{W}\Big) . \end{aligned}$$

Overall, we finally get a drift of $\Omega(p_{\text{total}} w_{\min} \rho / W)$. ■

Since there is a strong drift for the frequencies toward 0, due to Lemma 8.3, we can now bound the run time of the MMAS-fp with very high probability.

▶ **Theorem 8.4.** Let $m \in \omega\big(\sigma\sqrt{\log(n)}\big)$. Consider the MMAS-fp optimizing $f_{[\sigma^2]}$ with no margin or a margin $b \in \mathrm{o}\big(w_{\min}/(w_{\max} n^3)\big)$ and with $\rho \in \mathrm{o}\big(W w_{\min}/((W + m)^2 \log(n))\big) \cap \mathrm{o}(W/((W + m)n))$. Then, for any $\lambda \in \omega\big(\log(n)\big)$, the algorithm finds the optimum after $\mathrm{O}\big(W\lambda \log(n)/(\rho w_{\min})\big)$ iterations with very high probability. ◀

Proof. As before, we assume a non-failing run and that no frequency reaches values of at least a constant in $(1/2, 1)$. Due to Lemmas 8.1 and 8.2, all of the following statements then hold with very high probability.

Let T denote the first point in time such that the MMAS-fp samples the optimum, that is, the all-1s string. In order to bound the probability that the MMAS-fp takes more than $\mathrm{O}\big(W\lambda/(\rho w_{\min})\big)$ iterations, we proceed as follows: we assume that the MMAS-fp can only sample the optimum if the sum of all frequencies p_{total} is sufficiently small. If the optimum is not sampled then, we reset the algorithm. Let T' denote the first point in time that this process samples the optimum. Note that T' stochastically dominates T, since the normal MMAS-fp has a chance of sampling the optimum in any iteration and since its frequencies are at most that of the augmented process. Thus, we bound the probability that T' exceeds $\mathrm{O}\big(W\lambda \log(n)/(\rho w_{\min})\big)$.

We first consider a single (possibly restarted) run of the MMAS-fp. Let T^* denote the first point in time such that p_{total} drops below 1, and let $\varepsilon > 0$ be a constant. Due to the multiplicative of $p_{\text{total}} \cdot \varepsilon \rho w_{\min}/W$ from Lemma 8.3, we can apply Theorem 7.3, assuming that p_{total} is 0 once it is less than 1 and noting that p_{total} is $n/2$ at the beginning, and we get

$$\Pr\left[T^* > \frac{1 + \ln\left(\frac{n}{2}\right)}{\frac{\varepsilon \rho w_{\min}}{W}}\right] \leq \mathrm{e}^{-1} . \tag{8.1}$$

Now, the augmented process has one chance to sample the optimum. We know that $p_{\text{total}} < 1$, but this value does not allow us to uniquely determine the value of each frequency individually. However, note that the probability to sample the optimum is then minimized by assuming that as many frequencies as possible are as high as possible, since the probability of sampling the optimum is a Schur-concave function over $[0, 1]^n$ [BP83]. Thus, we pessimistically assume that a constant number of frequencies have constant values and that the other frequencies are at the lower border b or at 0. This results in a constant probability of $c > 0$ of sampling the optimum.

Taking this probability together with the probability from equation (8.1), we see that the probability that a single run does *not* sample the optimum within $\mathrm{O}\big(W \log(n)/(\rho w_{\min})\big)$ iterations is at most $(1 - \mathrm{e}^{-1})(1 - c)$, which is at most a constant $c' < 1$.

Note that the number of restarts of our augmented process follows a geometric distribution with a success probability of $1 - c'$, where *success* means that the optimum was sampled. Thus, the probability that the optimum was not sampled after $\lambda \in \mathbf{N}$ restarts, is at most $(c')^{\lambda+1}$. Choosing $\lambda \in \omega\big(\log(n)\big)$ yields a superpolynomially low probability. Since every restart accounts for $\mathrm{O}\big(W \log(n)/(\rho w_{\min})\big)$ iterations, the augmented process finds the optimum after $\mathrm{O}\big(W\lambda \log(n)/(\rho w_{\min})\big)$ iterations with very high probability.

Overall, due to $T' \geq T$, conditional on a non-failing run and the frequencies not reaching constant values greater than $1/2$, the MMAS-fp finds the optimum within $\mathrm{O}\big(W\lambda \log(n)/(\rho w_{\min})\big)$ iterations with very high probability. Using the law of total probability (Theorem 2.1) then yields that the unconditional probability of the MMAS-fp sampling the optimum after the mentioned number of iterations is still very high. This concludes the proof. ■

Theorem 8.4 shows that the MMAS-fp scales gracefully (see Definition 7.1) with Gaussian noise, since, assuming a polynomial variance, we can choose a polynomial ρ such that the run time of MMAS-fp is polynomial with very high probability. However, note that this only holds if $1/w_{\min}$ and W are both upper-bounded by a polynomial. Otherwise, the step size is too small in order to result in a polynomial run time.

8.3.1 Non-Gaussian Noise

Note that our considerations only discussed noise in Lemma 8.1. All of the other theorems analyzed the MMAS-fp as if it optimized a non-noisy fitness function. Since Lemma 8.1 only makes use of the noise bounds of the Gaussian distribution and no other properties (except its mean being 0), we can generalize our results to a greater class of noise distributions. For any random variable D, we now consider optimization of the function $f_D(x) \coloneqq f(x) + D$. This leads to the following corollary.

▶ **Corollary 8.5.** Consider the MMAS-fp optimizing f_D with no margin or a margin $b \in \mathrm{o}\big(w_{\min}/(w_{\max}n^3)\big)$, where D is a random variable with $\mathrm{E}[D] = 0$ such that, for some function $t^*(n) > 0$ and all polynomials $p(n)$, it holds that

$$\Pr\big[|D| > t^*(n)\big] \in \mathrm{o}\left(\frac{1}{p(n)}\right). \tag{8.2}$$

If $\rho = \mathrm{o}\big(Ww_{\min}/\big((W + t^*(n))^2 \log(n)\big)\big)$, then, for any $\lambda \in \omega\big(\log(n)\big)$, the algorithm finds the optimum after $\mathrm{O}\big(W\lambda \log(n)/(\rho w_{\min})\big)$ iterations with very high probability. ◀

Proof. We follow the same proof outline used to prove Theorem 8.4. The only difference is that we use $t^*(n)$ instead of m, which is possible, since equation (8.2) guarantees that a run of the MMAS-fp is non-failing with very high probability (similar to Lemma 8.1). Further, due to $\mathrm{E}[D] = 0$, the noise is ignored in the drift calculations. This concludes the proof. ■

Note that Corollary 8.5 does not tell us whether the MMAS-fp scales gracefully with noise D, as we do not know what the variance of the noise is. However, if $t^*(n)$ can be expressed as a polynomial in the variance of the noise, then the MMAS-fp scales gracefully with D.

8.4 Conclusions

We proved that the MMAS-fp optimizing linear pseudo-Boolean functions scales gracefully with Gaussian noise (Theorem 8.4). This means that the run time depends polynomially (in fact, only linearly) on the variance of the noise. This is similar to our results in Chapter 7, where we proved that the cGA scales gracefully with Gaussion noise on OneMax. However, in contrast to the cGA, our results for the MMAS-fp hold for all linear functions where the weights are still polynomial. Further, the noise model can be generalized to distributions that can be bounded with very high probability (Corollary 8.5).

The main difference in the analyses of the cGA and the MMAS-fp is that the former samples two individuals whereas the latter only samples one. Since, for the cGA, the components of both offspring need to be compared (and the result may be incorrect due to the noise), the analysis is more involved, whereas for the MMAS-fp, we were able to bound the impact of the noise (Lemma 8.1) and then basically analyze an unnoisy fitness function.

The same drawbacks of results on graceful scaling that we discussed in Section 7.4 apply here as well. Namely, the variance of the noise has to be known in advance for the optimization to succeed. However, the same approach that we proposed for the cGA in order to circumvent this can also be used for the MMAS-fp. The only problem that remains is that the MMAS-fp still needs to have a bound on the sum of all weights W of the fitness function in order to perform a normal update. Unfortunately, this is an inherent problem of the algorithm and not introduced by the noisy setting. Assuming that the MMAS-fp is only chosen when it seems appropriate, that is, the user has such a bound on the sum of weights, this poses no issue any longer.

Since we now have proven that two n-Bernoulli-λ-EDAs scale gracefully with noise, an interesting question is if all of the algorithms mentioned in Section 4.3 show this behavior and if this result holds for (univariate) EDAs in general.

9 Lower Bound of the UMDA on OneMax

This chapter is based on joint work with Carsten Witt [KW18a]. It eliminates some minor inaccuracies from the journal version.

In this chapter, we prove that the UMDA (with a margin of $1/n$) needs in expectation at least $\Omega\big(\mu\sqrt{n} + n\log(n)\big)$ fitness function evaluations to optimize OneMax (Theorems 9.6 and 9.20). In conjunction with some recently and independently proven upper bounds by Lehre and Nguyen [LN17] and Witt [Wit17], this results in a $\Theta\big(n\log(n)\big)$ run time for the UMDA on OneMax for many parameter combinations of μ and λ.

9.1 Introduction

The arguably most important research area in the theory of evolutionary computation is run time analysis, giving insights into how and why different algorithms behave in a certain way on specific functions. While the formal analysis of EAs started in the 1990s [Müh92], the first run time analysis of EDAs was conducted far later by Droste [Dro06], who analyzed the cGA on linear functions. Papers considering other EDAs, like, for example, the λ-MMAS$_{\text{IB}}$ by Neumann et al. [NSW10] followed.

Recently, more run time results for EDAs were published [DK18b; DL15; FKK16; LN17; LN18; LSW18; SW16a; Wit17; Wit18]. Most of these works consider either the cGA [LSW18; SW16a; Wit18] or the UMDA [DL15; LN17; Wit17], mostly on the function OneMax. While the papers on the cGA interestingly all contain some lower bounds, the same is not true for the UMDA. This chapter provides the first lower-bound run time analysis for the UMDA.

We follow the ideas of Sudholt and Witt [SW16a] and derive a lower bound of $\Omega\big(n\log(n)\big)$ for the UMDA on OneMax, which is a typical lower bound for many evolutionary algorithms on this function. The algorithm has already been analyzed some years ago for several artificially designed example functions [Che+07; Che+09a; Che+09b; Che+10]. However, none of these papers consider the standard benchmark function for theory: the OneMax function. In fact, the run time analysis of the UMDA on OneMax has turned out to be rather challenging; the first such result, showing an upper bound of $\mathrm{O}\big(n \log(n) \log(\log(n))\big)$ on its expected run time for certain settings of μ and λ, was not published until 2015 [DL15]. Specific lower bounds for the UMDA were to date missing. The previous best result $\Omega\big(n/\log(n)\big)$ on the expected run time followed from general black-box complexity theory [DJW06] and did not shed light on the working principles of the UMDA.

Recently, two matching upper bounds of $\mathrm{O}\big(n \log(n)\big)$ of the UMDA on OneMax have been proven independently from one another [LN17; Wit17] for certain cases of μ and λ. Our results match almost all of these cases, providing a tight run time bound of $\Theta\big(n \log(n)\big)$.

As mentioned above, the concepts of the proofs in this chapter are based on prior work from Sudholt and Witt [SW16a]. However, analyzing the UMDA is much more difficult than analyzing the cGA or the 2-MMAS$_{\mathrm{IB}}$, since the update of the latter algorithms is bounded by an algorithm-specific parameter and the algorithms only have up to three distinct successor states for each frequency. The UMDA, on the other hand, can change each of its frequencies to any value x/μ with a certain probability, where $x \in [0..\mu]$, due to the nature of its update rule. This makes analyzing the UMDA far more involved, because every single update has to be bounded probabilistically. Further, the simple update rules for the cGA and the 2-MMAS$_{\mathrm{IB}}$ allow for a distinction into two cases that determines whether a frequency will increase or decrease; a fact that was heavily exploited in the analyses by Sudholt and Witt [SW16a]. For the UMDA, no such simple case distinction can be made.

9.2 Preliminaries

We consider the UMDA (see Section 4.3.2) with a margin of $1/n$ optimizing OneMax. Since the UMDA is unbiased (Theorem 5.11), all of our results generalize to the OneMax class. When choosing the μ fittest individuals among the λ offspring (so-called *selection*), we break ties uniformly at random.

We are interested in a lower bound of the UMDA's expected number of *function evaluations* on OneMax until the optimum is sampled. Note that this is at least the expected number of iterations until the optimum is sampled (minus one) times λ, as we do not necessarily have to evaluate all λ offspring in the last iteration.

In all of our calculations except Section 9.4, we always assume, for some constant $\beta > 0$, that $\lambda = (1 + \beta)\mu$.

9.2.1 Selecting Individuals

In order to optimize a function efficiently, the UMDA needs to evolve its frequencies toward the right direction, making it more likely to sample an optimum. In the setting of OneMax, this means that each frequency should be increased (toward a value of $1 - 1/n$). This is where selection comes into play.

By selecting μ best individuals every iteration with respect to their fitness, we hope that many of them have correctly set bits at each position, such that the respective frequencies increase. However, even in the simple case of OneMax, where a 1 is always better than a 0, there is a flaw in the update process that prevents UMDA from optimizing OneMax too fast. To see why this flaw occurs, consider an arbitrary position $j \in [n]$ in the following.

When selecting individuals for an update to p_j, the UMDA considers the fitness of each *entire* individual. That is, although each frequency is independently updated from the others, selection is done with respect to *all* positions at once. Thus, when looking at position j, it can happen that we have many 0s, because the individuals chosen for the update may have many 1s in their remaining positions, which can lead to a decrease of p_j.

Since having a 1 at a position is always better than a 0 when considering OneMax, the selection is biased, pushing for more 1s at each position. However, this bias is not necessarily too large: consider that for each individual each bit but bit j has already been sampled. When looking at the selection with respect to only $n - 1$ bits in each individual, some individuals may already be so good that they are determined to be chosen for selection, whereas others may be so bad that they definitely cannot be chosen for selection, regardless of the outcome of bit j.

Consider the fitness of all individuals sampled during one iteration of the UMDA with respect to $n - 1$ bits, that is, all bits but bit j. We call each of these n different fitness values (from 0 to $n - 1$) a *level.* Assume that the individuals are sorted decreasingly by their level; each individual having a unique index. Let w^+ be the level of the individual with rank μ, and let w^- be the level of the individual with rank $\mu + 1$. Since bit j has not been considered so far, its value can potentially increase each individual's level by 1. Now assume that $w^+ = w^- + 1$. Then individuals from level w^- can end up with the same fitness as individuals from level w^+, once bit j has been sampled. Thus, individuals from level w^+ were still *prone to selection.* This means that the outcome of bis j can influence whether the individual is being selected or not.

Among the μ individuals chosen during selection, we distinguish between two different types: 1*st-class* and 2*nd-class* individuals. 1st-class individuals are those which are chosen during selection no matter which value bit j has. The remaining of the μ selected individuals are the 2nd-class individuals; they had to compete with other individuals for selection. Therefore, their bit value j is biased toward 1 compared to 1st-class individuals. Note that 2nd-class individuals can only exist if $w^+ \leq w^- + 1$, since, in this case, individuals from level w^- can still be as good as individuals from level w^+ after sampling bit j.

For any iteration t, let X_t denote the number of 1s at position j of the μ selected individuals of the UMDA, and let C^* denote the number of 2nd-class individuals in iteration $t + 1$. Note that the number of 1s of 1st-class individuals during iteration $t+1$ follows a binomial distribution with success probability X_t/μ. Since we have $\mu - C^*$ 1st-class individuals, the distribution of the number of 1s of these follows $\mathrm{Bin}(\mu - C^*, X_t/\mu)$. Note that the actual frequency in iteration $t + 1$ might be set to either $1/n$ or $1 - 1/n$ if the number of 1s in the μ selected individuals is too close to 0 or μ, respectively. We will be able to ignore this fact in our forthcoming analyses, since all considerations are stopped when a frequency drops below $1/n$ or exceeds $1 - 1/n$.

9.2.2 The Number of 2nd-Class Individuals

Consider a bit position $j \in [n]$. In this section, we again speak of levels as defined in the previous section. Those definitions as well as the following ones are also depicted in Figure 9.1. Level $n - 1$ is the topmost, and level 0 is the bottommost. For all $i \in [0..n - 1]$, let C_i denote the cardinality of level i, that is, the number of individuals in level i during an arbitrary iteration of the UMDA, and let $C_{\geq i} = \sum_{a=i}^{n-1} C_a$.

Let M denote the index of the first level from the top such that the number of sampled individuals is greater than μ when including the following level, that is,

$$M = \max\{i \mid C_{\geq i-1} > \mu\} .$$

Note that M can never be 0, and only if $M = n-1$, then C_M can be greater than μ. Note further that C_M can be 0.

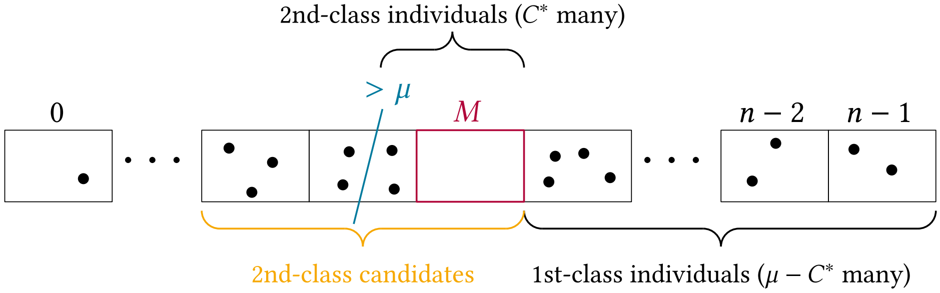

2nd-class individuals (C^* many)
$> \mu$
0
M
$n-2$
$n-1$
2nd-class candidates
1st-class individuals ($\mu - C^*$ many)

Figure 9.1: An exemplary visualization of the different definitions we need. The boxes depict all of the n levels, the numbers above show their respective fitness, and the black dots symbolize individuals in these levels. The blue line cutting through level $M-1$ marks the point where more than μ individuals have been sampled when starting from the top. In that level, not all individuals are going to be selected. Further, the individuals from the level below can be selected (as their fitness can still increase by one when sampling the last bit), and individuals from the level above can be not selected. Hence, the individuals in those levels are 2nd-class candidates. The individuals in higher levels will always be selected, thus they are 1st-class individuals. Out of the 2nd-class candidates, those individuals that are chosen during selection are the 2nd-class individuals (in this example, those would be two individuals, that is, $C^* = 2$). Last, M depicts the cut level, that is, the topmost level such that the number of sampled individuals is greater than μ when including the next (lower) level.

Due to the definition of M, if $M \neq n-1$, level $M-1$ contains the individual of rank $\mu+1$, as described in the previous section. Thus, levels M, $M-1$, and $M-2$ contain all of the individuals that are prone to selection (if such exist at all). Hence, individuals in levels at least $M+1$ are definitely 1st-class individuals. 2nd-class individuals, if any, have to come from level M, $M-1$, or $M-2$. We call the individuals from these three levels *2nd-class candidates.* Note that the actual number of 2nd-class individuals is bounded from above by $\mu - C_{\geq M+1} = \mu - C_{\geq M} + C_M$, since exactly μ individuals are selected.

Since the 2nd-class individuals are the only ones that are prone to selection and thus the only ones that actively help in progressing a single frequency toward $1-1/n$, it is of utmost importance to understand the distribution of $C^* := \mu - C_{\geq M+1}$, that is, the biased impact to an update as explained in Section 9.2.1. Moreover, we will also need a bound on the number of 2nd-class candidates.

Before we get to analyzing the 2nd-class individuals, we introduce several auxiliary statements. We start with a very useful lemma on conditional binomially distributed random variables.

▶ **Lemma 9.1.** Let $x, y \geq 0$, and let X be a binomially distributed random variable with arbitrary parameters such that $\Pr[X \geq x] > 0$. Then

$$\Pr[X \geq x + y \mid X \geq x] \leq \Pr[X \geq y] \ . \qquad ◀$$

Proof. Let n and p be the parameters of the underlying binomial distribution of X. Given $x \geq 0$, we define the random variable $Y_x := X - x$. Conditional on $X \geq x$, we have $Y_x \sim \mathrm{Bin}(k, p)$ for $k \in [0..n - x]$ and therefore $Y_x \preceq X$ (see Section 2.2.2). Hence, $\Pr[X \geq x + y \mid X \geq x] = \Pr[Y_x \geq y \mid X \geq x] \leq \Pr[X \geq y]$. ■

Moreover, we are going to use a corollary that is based on Lemma 8 from Sudholt and Witt [SW16a], the proof of which can be seen in an extended version [SW16b, Lemma 9]. Also, the idea behind the corollary is given by Sudholt and Witt [SW16b] but not presented as an independent statement.

▶ **Lemma 9.2 ([SW16b, Lemma 9]).** Let S be the sum of $m \in \mathbf{N}^+$ independent Poisson trials where trial $i \in [m]$ has success probability $p_i \in [1/6, 5/6]$. Then, for all $s \in [0..m]$, it holds that $\Pr[S = s] \in \mathrm{O}\big(1/\sqrt{m}\big)$. ◀

▶ **Corollary 9.3.** Let X be the sum of $n \in \mathbf{N}^+$ independent Poisson trials where trial $i \in [n]$ has success probability p_i. Further, let $\Theta(n)$ many p_i-s be within $[1/6, 5/6]$. Then, for all $x \in [0..n]$, it holds that $\Pr[X = x] \in \mathrm{O}\big(1/\sqrt{n}\big)$. ◀

Proof. Let $m \in \Theta(n)$ denote the number of p_i-s that are within $[1/6, 5/6]$. When sampling X, assume without loss of generality that the first m trials are the ones with $p_i \in [1/6, 5/6]$. Let S denote the sum of these trials, and let Y denote the sum of the remaining $n - m$ trials. Since the trials are independent, we get

$$\Pr[X = x] = \sum_{s=0}^{x} \Pr[S = s]\Pr[Y = x - s] \ .$$

We can upper-bound $\Pr[S = s] \in \mathrm{O}\big(1/\sqrt{m}\big) = \mathrm{O}\big(1/\sqrt{n}\big)$ by using Lemma 9.2 and $m \in \Theta(n)$. Thus, there exists a constant $c > 0$ such that

$$\Pr[X = x] \leq \frac{c}{\sqrt{n}} \sum_{s=0}^{x} \Pr[Y = x - s] \ .$$

Bounding the sum by 1 concludes the proof. ■

Corollary 9.3 lets us easily get upper bounds for the probability that a sampled individual has a certain (and arbitrary) fitness (with respect to either all n positions or all positions but j). In order to apply it, we have to make sure that $\Theta(n)$ frequencies are still within $[1/6, 5/6]$. Thus, we assume from now on that this assumption holds. In Section 9.3.2, we go into detail and prove under which circumstances this assumption holds.

Note that all statements from now on regarding a specific position j hold regardless of the bits at any other of the $\Theta(n)$ positions that do not stay within $[1/6, 5/6]$. This means that the statements are even true if the bits at those other positions are chosen by an adversary.

We are now ready to analyze C^* and the number of 2nd-class candidates.

▶ **Lemma 9.4.** Consider the UMDA with $\lambda = (1 + \beta)\mu$ optimizing OneMax. Let $\varepsilon > 0$ be a constant such that $\varepsilon < 1 - 1/(1 + \beta)$, and let $\widetilde{Z}$ be a random variable that takes values in $[\lambda]$ only with probability at most $2\mathrm{e}^{-(\varepsilon^2/(3+3\varepsilon))\mu}$ and is 0 otherwise. If there are $\Theta(n)$ frequencies in $[1/6, 5/6]$, then the distribution of C^* is stochastically dominated by $\mathrm{Bin}\big(\lambda, \mathrm{O}\big(1/\sqrt{n}\big)\big) + \widetilde{Z}$, and the distribution of $C_M + C_{M-1} + C_{M-2}$ is stochastically dominated by $1 + \mathrm{Bin}\big(\lambda, \mathrm{O}\big(1/\sqrt{n}\big)\big) + \widetilde{Z}$. ◀

Proof. We carefully investigate and then reformulate the stochastic process generating the λ individuals (before selection), restricted to $n - 1$ bits. Each individual is sampled via a vector of probabilities $\boldsymbol{p}' = (\boldsymbol{p}'_1, \ldots, \boldsymbol{p}'_{n-1})$ obtained from the frequencies of the UMDA by leaving one entry out. By counting its number of 1s, each of the λ individuals then falls into some level $i \in [0..n - 1]$, with some probability q_i depending on the vector $\boldsymbol{p}'$. Since the individuals are created independently, the number of individuals in level i is binomially distributed with parameters λ and q_i.

Next, we take an alternative view on the process of putting individuals into levels, using the principle of deferred decisions. We imagine that the process first samples all individuals in level 0 (through λ trials, all of which either hit the level or not), then (using the trials which did not hit level 0) all individuals in level 1, and so on, up to level $n - 1$.

The number of individuals C_0 in level 0 is still binomially distributed with parameters λ and q_0. However, after all individuals in level 0 have been sampled, the distribution changes. We have $\lambda - C_0$ trials left, each of which can hit one of the levels 1 to $n - 1$. In particular, such a trial will hit level 1 with probability

$q_1/(1-q_0)$, by the definition of conditional probability, since level 0 is excluded. This holds independently for all of the $\lambda - C_0$ trials so that C_1 follows a binomial distribution with parameters $\lambda - C_0$ and $q_1/(1-q_0)$. Inductively, also all C_i for $i > 1$ are binomially distributed; for example, C_{n-1} is distributed with parameters $\lambda - C_{n-2} - \cdots - C_0$ and 1. Note that this model of the sampling process can also be applied for any other permutation of the levels; we will make use of this fact.

Analyzing the number of 2nd-class individuals. We first focus on $C^* = \mu - C_{\geq M+1}$ and will later use bounds on its distribution to analyze $C_M + C_{M-1} + C_{M-2}$. Formally, by applying the law of total probability (Theorem 2.1), the distribution of C^* looks as follows for $k \in [0..\lambda]$:

$$\Pr[C^* \geq k] = \sum_{i=1}^{n-1} \Pr[M = i] \cdot \Pr[\mu - C_{\geq i+1} \geq k \mid M = i] \,. \tag{9.1}$$

We bound the terms of the sum differently with respect to the index i. First, we look into a particular value i^* such that $\Pr[M \geq i^*]$ is exponentially unlikely, and then make a case distinction via i^*.

Let X be the number of 1s in a single individual sampled (without conditioning on certain levels being hit). Let $c_1 > 0$ be a constant, and choose i^* such that

$$\Pr[X \geq i^* - 1] \leq \frac{1}{(1+\varepsilon)(1+\beta)} \qquad \text{and}$$
$$\Pr[X \geq i^* - 1] \geq \frac{1}{(1+\varepsilon)(1+\beta)} - \frac{c_1}{\sqrt{n}} \,.$$

Such an i^* must exist, since every level is hit with probability $\mathrm{O}\big(1/\sqrt{n}\big)$ when sampling an individual, according to Corollary 9.3. Clearly, we also have $i^* \leq n-1$.

A crucial observation is that there is a constant $c_2 > 0$ such that $\Pr[M \geq i^*] \leq \mathrm{e}^{-c_2\mu}$, since the expected number of individuals sampled with at least $i^* - 1$ 1s is at most $\lambda/\big((1+\varepsilon)(1+\beta)\big) = \mu/(1+\varepsilon)$, and the probability of sampling at least $(1+\varepsilon)\cdot\mu/(1+\varepsilon) = \mu$ is at most $\mathrm{e}^{-\varepsilon^2\cdot\mu/(3(1+\varepsilon))} = \mathrm{e}^{-c_2\mu}$ by Chernoff bounds (Theorem 2.19). Note that we have considered level $i^* - 1$ because $C_{\geq i^*-1} < \mu$ implies $M < i^*$.

In equation (9.1), considering the partial sum for all $i \geq i^*$, we therefore immediately estimate , as just discussed,

$$\sum_{i=i^*}^{n-1} \Pr[M=i] \cdot \Pr[\mu - C_{\geq i+1} \geq k \mid M=i] \leq \Pr[M \geq i^*]$$
$$\leq \mathrm{e}^{-c_2 \mu} .$$

For the terms with $i < i^*$ (in particular, the case $i = n-1$ is excluded), we take a view on the final expression in equation (9.1) and focus on the conditional probability $\Pr[\mu - C_{\geq i+1} \geq k \mid M = i]$, in which we want to reformulate the underlying event appropriately. Here we note that, defining $C_{\leq i} = \sum_{j=0}^{i} C_j$, the event

$$\{\mu - C_{\geq i+1} \geq k\} \cap \{M = i\}$$

is equivalent to

$$\{C_{\leq i} \geq \lambda - \mu + k\} \cap \{M = i\} ,$$

and, using the definition of M, this is also equivalent to

$$\{C_{\leq i} \geq \lambda - \mu + k\} \cap \{C_{\leq i-2} < \lambda - \mu\} \cap \{C_{\leq i-1} \geq \lambda - \mu\} .$$

We now take the above-mentioned view on the stochastic process and assume that levels 0 to $i-2$ have been sampled, and a number of experiments in a binomial distribution is carried out to determine the individuals from level $i-1$. Hence, given some $C_{\leq i-2} =: a < \lambda - \mu$, our event is equivalent to the event

$$E^* := \left\{C_i + C_{i-1} \geq (\lambda - \mu - a) + k\right\} \cap \{C_{i-1} \geq \lambda - \mu - a\} .$$

Recall from our model above that C_{i-1} follows a binomial distribution with $\lambda - a$ trials and with a certain success probability s. Similarly, C_i follows a binomial distribution with parameters $\lambda - a - C_{i-1}$ and s'. As we are interested in a cumulative distribution, we may pessimistically upper-bound the total number of trials for C_{i-1} by λ. Regarding s, note that it denotes the probability to sample an individual with $i-1$ 1s, given that it cannot have less than $i-1$ 1s. Note further that $\Pr[X \geq i^* - 1]$, where X again denotes the level of the individual sampled in a trial, is a lower bound for all probabilities $\Pr[X \geq i-1]$, since $i < i^*$. To upper-bound s, we use Corollary 9.3, which tells us that the unconditional probability to hit a level is in $\mathrm{O}(1/\sqrt{n})$, regardless of which level is hit. However, we have to condition on the event that certain levels (namely 0 to $i-2$, where

$i < i^*$) cannot be hit anymore. We pessimistically exclude even some more levels than possible, more precisely, we exclude the levels from 0 up to $i^* - 2$. This means that we condition on $\Pr[X \geq i^* - 1]$. By the definition of conditional probability, the probability of $O\bigl(1/\sqrt{n}\bigr)$ from Corollary 9.3 thus gets increased by a factor of $1/\Pr[X \geq i^* - 1]$, which is constant. Hence, C_{i-1} is stochastically dominated by a binomial distribution with parameters λ and $O\bigl(1/\sqrt{n}\bigr)$.

Similarly, assuming that also level $i - 1$ has been sampled, C_i is dominated by a binomial distribution with parameters $\lambda - C_{i-1}$ and $O\bigl(1/\sqrt{n}\bigr)$.

In order to finally bound $\Pr[E^*]$ from above, which involves a condition on the outcome of C_{i-1}, we apply Lemma 9.1, where we let $X := C_{i-1}$ and $x = \lambda - \mu - a$ as well as $y = k$. Since we have bounded C_{i-1} (without the condition on $C_{i-1} \geq x$) by a binomial distribution with success probability $O\bigl(1/\sqrt{n}\bigr)$, we get from the lemma that $\Pr[C_{i-1} - x \geq k \mid C_{i-1} \geq x] \leq \Pr\bigl[\mathrm{Bin}\left(\lambda, O\bigl(1/\sqrt{n}\bigr)\right) \geq k\bigr]$. Note that the right-hand side is a bound independent of $C_0, \ldots, C_{i-1}$. With respect to C_i, we do not consider an additional condition on its outcome but use the result $\Pr[C_i \geq k] \leq \Pr\bigl[\mathrm{Bin}(\lambda - C_{i-1}, O\bigl(1/\sqrt{n}\bigr)) \geq k\bigr]$ derived in the last paragraph directly. Hence, both $C_{i-1} - x$, conditional on $C_{i-1} \geq x$, and C_i have been bounded by binomial distributions with a success probability in $O\bigl(1/\sqrt{n}\bigr)$. In E^*, we are confronted with the sum of these two random variables and study the distribution of the sum. Together, we get

$$\Pr[E^*] \leq \Pr\bigl[\mathrm{Bin}(\lambda, O\bigl(1/\sqrt{n}\bigr)) \geq k\bigr] ,$$

since we consider at most λ trials. Pulling this term in front of the sum over i for the terms $i < i^*$ in equation (9.1) and estimating this sum with 1 (since we sum over mutually disjoint events) leaves us with an additional term of $\Pr\bigl[\mathrm{Bin}(\lambda, O\bigl(1/\sqrt{n}\bigr)) \geq k\bigr]$ for $\Pr[\mu - C_{\geq M+1} \geq k]$. This proves the lemma's statement on the distribution of C^*.

Analyzing the number of 2nd-class candidates. We are left with analyzing $C^{**} := C_M + C_{M-1} + C_{M-2}$. We handle the very unlikely case $M = n - 1$, whose probability is upper-bounded by $\Pr[M \geq i^*]$, separately and cover it by adding the random variable $\widetilde{Z}$ to our result. By a symmetrical argument to the above, for some index i^{**} such that there is a constant $c_3 > 0$ such that

$$\Pr[X < i^{**}] = 1 - \frac{1}{(1-\varepsilon)(1+\beta)} + \frac{c_3}{\sqrt{n}} ,$$

we obtain that $M \leq i^{**}$ also happens with probability at most $\mathrm{e}^{-\varepsilon^2 \cdot \mu/(2(1-\varepsilon))} \leq \mathrm{e}^{-\varepsilon^2 \cdot \mu/(3+3\varepsilon))}$, for $\varepsilon < 1 - 1/(1+\beta)$. This unlikely case is also included in $\widetilde{Z}$. From now on, we assume $i^{**} < M < n-1$. We note that by definition of M, we then have $C_{\geq M} \leq \mu$ and $C_{\geq M-1} \geq \mu + 1$. Hence, $C_{M-1} \geq 1$ such that we have to investigate the distribution of C^{**} conditional on $\{C_{M-1} \geq 1 + (\mu - C_{\geq M})\}$.

We take the same view on the stochastic process as above but imagine now that the levels are sampled in the order from $n-1$ down to 0. Conditional on that the levels $n-1$ to $M+1$ have been sampled, the levels M, $M-1$, and $M-2$ are still hit with probability $\mathrm{O}\big(1/\sqrt{n}\big)$ each, since $\Pr[X < i^{**}]$ is a constant. Therefore, the distribution of C_M is bounded according to

$$\Pr[C_M \geq k] \leq \Pr\Big[\mathrm{Bin}(\lambda - C_{\geq M+1}, \mathrm{O}\big(1/\sqrt{n}\big)) \geq k\Big] .$$

In order to analyze C_{M-1}, we recall that we have to condition on the event $\{C_{M-1} \geq 1 + (\mu - C_{\geq M})\}$. Hence, we can use Lemma 9.1 similarly as above and get that

$$\begin{aligned}\Pr[C_{M-1} \geq 1 + (\mu - C_{\geq M}) + k \mid C_{M-1} \geq 1 + (\mu - C_{\geq M})]&\\ \leq \Pr\Big[\mathrm{Bin}(\lambda - C_{\geq M}, \mathrm{O}\big(1/\sqrt{n}\big)) \geq k\Big] .&\end{aligned}$$

Note that the right-hand side of the inequality is independent of C^{*}.

Applying the same argumentation once more for level $M-2$ (where no conditions on the size exist), we get

$$\Pr[C_{M-2} \geq k] \leq \Pr\Big[\mathrm{Bin}(\lambda - C_{\geq M-1}, \mathrm{O}\big(1/\sqrt{n}\big)) \geq k\Big] .$$

Using our stochastic bound on C^{*} from above, we altogether obtain that C^{**} is stochastically dominated by the sum of 1, three binomially distributed random variables with a total number of λ trials and success probability $\mathrm{O}\big(1/\sqrt{n}\big)$ each, and the variable $\widetilde{Z}$. ■

Now that we understand how C^{*} is distributed, we can look at the distribution of both the 1st- and 2nd-class individuals. We even can take a finer-grained view on the number of 1s contributed by them.

▶ **Lemma 9.5.** Consider the UMDA optimizing OneMax. Consider further that $\Theta(n)$ frequencies are within $[1/6, 5/6]$ and that we are in iteration t. Let $j \in [n]$

be any position, and let X_{t-1} denote the number of 1s at position j in iteration $t-1$.

The distribution $Z_{1,t}$ of the number of 1s of 1st-class individuals is stochastically dominated by $\mathrm{Bin}(\mu, X_{t-1}/\mu)$, and the distribution $Z_{2,t}$ of the number of 1s of 2nd-class individuals is stochastically dominated by C^*, where C^* is distributed as seen in Lemma 9.4. In particular, there are constants $c_1, c_2 > 0$ such that

$$\mathrm{E}\big[Z_{2,t}\big] \leq \frac{c_2\mu}{\sqrt{n}} + \mathrm{e}^{-c_1\mu} \qquad \text{and}$$
$$\mathrm{E}\big[Z_{2,t} \,\big|\, X_{t-1}\big] \leq c_2\left(\frac{X_{t-1}}{\mu} + \frac{X_{t-1}}{\sqrt{n}}\right) + \mathrm{e}^{-c_1\mu} \,. \qquad ◀$$

Proof. The distribution of $Z_{1,t}$ has already been described in Section 9.2.1 as $\mathrm{Bin}(\mu - C^*, X_{t-1}/\mu)$, which is dominated by $\mathrm{Bin}(\mu, X_{t-1}/\mu)$. We also know that the number of 2nd-class individuals is bounded from above by C^*, and their number of 1s is trivially bounded by this cardinality too. The first statement on the expected value of $Z_{2,t}$ follows by taking the expected value of the binomial distribution and noting that there is a constant $c_3 > 0$ such that $\mathrm{E}\big[\widetilde{Z}\big] \leq \lambda\mathrm{e}^{-c_1\mu} = \mathrm{e}^{-c_3\mu}$, since $\lambda \in \Theta(\mu)$.

In order to show the second statement on the expected value of $Z_{2,t}$, we recall our definition of 2nd-class candidates from above. These candidates have not been subject to selection yet. Each of these candidates samples a 1 at position j independently of the others with probability X_{t-1}/μ, thus the expected total number of 1s in 2nd-class candidates is the expected number of candidates multiplied with X_{t-1}/μ, by Wald's equation (Theorem 2.21). By Lemma 9.4, there is an expected number of at most $1 + c_2\mu/\sqrt{n} + \mathrm{e}^{-c_3\mu}$ of candidates, using again $\lambda \in \Theta(\mu)$. Since the 2nd-class individuals are only selected from the candidates, the claim on the expected value of $Z_{2,t}$ follows. ■

9.3 Lower Bound on OneMax

In the following, we derive a lower bound on the UMDA's run time on OneMax. First, we state the main theorem.

▶ **Theorem 9.6.** For some constant $\beta > 0$, let $\lambda = (1+\beta)\mu$. Then the expected optimization time of the UMDA on OneMax is $\Omega\big(\mu\sqrt{n} + n\log(n)\big)$. ◀

In order to prove the theorem, we distinguish between different cases for λ: small, medium, and large. We cover the lemmas that we use to prove the different cases in different sections. The first and the last case are fairly easy to prove, hence we discuss them first, leaving the second case of medium λ – the most difficult one – to be discussed last.

In each of the following sections, we introduce the basic idea behind each of the proofs.

9.3.1 Small Population Sizes

In this section, for some constant $c_1 > 0$, we consider a population size of $\lambda \leq (1 - c_1)\log_2(n)$. If the population size is that small, the probability that a frequency reaches $1/n$ is rather high, and thus the probability to sample the optimum will be quite small.

If enough frequencies drop to $1/n$, we can bound the expected number of fitness evaluations until we sample the optimum by $\Omega\big(n\log(n)\big)$. The following lemma and its proof closely follow Sudholt and Witt [SW16b, Lemma 13].

▶ **Lemma 9.7.** Assume that $\Omega(n^{c_1})$ frequencies, $c_1 > 0$ being a constant, are at $1/n$. Then the UMDA needs with overwhelming probability and in expectation $\Omega\big(n\log(n)\big)$ function evaluations to optimize any function, where the all-1s string is the unique global optimum. ◀

Proof. We look at $\big(c_2 n\ln(n)\big)/(2\lambda)$ iterations, where $c_2 < c_1$ is a positive constant, and show that it is very unlikely to sample the all-1s string during that time. Note that this translates to $\Omega\big(n\log(n)\big)$ function evaluations until the optimum is sampled, as the UMDA samples λ offspring every iteration.

Consider a single position with frequency at $1/n$. The probability that this position never samples a 1 during our time of $\big(c_2 n\ln(n)\big)/(2\lambda)$ iterations is at least

$$\begin{aligned}\left(\left(1-\frac{1}{n}\right)^{\lambda}\right)^{\frac{c_2 n\ln(n)}{2\lambda}} &= \left(1-\frac{1}{n}\right)^{\frac{c_2 n\ln(n)}{2}}\\ &\geq \frac{1}{2}\mathrm{e}^{-\frac{c_2}{2}\ln(n)}\\ &\geq n^{-c_2}\end{aligned}$$

if n is large enough, where we used Theorem 2.22.

Given at least $c_3 n^{c_1}$ frequencies at $1/n$, with $c_3 > 0$ being a constant, the probability that all of these positions sample at least one 1 during $\big(c_2 n \ln(n)\big)/(2\lambda)$ iterations is at most, using Theorem 2.22,

$$(1 - n^{-c_2})^{c_3 n^{c_1}} \leq \mathrm{e}^{-c_3 n^{c_1 - c_2}} ,$$

which is exponentially small in n, since $c_1 > c_2$, due to our assumptions. Hence, with overwhelming probability, UMDA will need at least $\Omega\big(n \log(n)\big)$ function evaluations to find the optimum. ■

Note that since the UMDA is unbiased (Theorem 5.11), Lemma 9.7 holds for any function with a unique global optimum, if $\Omega(n^{c_1})$ frequencies are at the wrong borders.

We can now prove our lower bound for small population sizes.

▶ **Theorem 9.8.** Let $\lambda \leq (1-c_1)\log_2(n)$ for some arbitrarily small constant $c_1 > 0$. Then the UMDA needs with overwhelming probability and in expectation $\Omega\big(n \log(n)\big)$ function evaluations to optimize any function with a unique global optimum. ◀

Proof. Since the UMDA is unbiased, we assume without loss of generality that the global optimum is the all-1s string. We consider an arbitrary position $i \in [n]$ and study the first iteration of the UMDA.

The probability that all λ bits at position i are sampled as 0 equals $2^{-\lambda} \geq n^{-(1-c_1)}$. In this case, the frequency of the position is set to $1/n$. The expected number of such positions is n^{c_1}, and by Chernoff bounds (Theorem 2.19), with overwhelming probability, $\Omega(n^{c_1})$ such positions exist (noting that c_1 is a positive constant by assumption).

Applying Lemma 9.7 yields the result, since we already have $\Omega(n^{c_1})$ frequencies at $1/n$ after a single iteration of the UMDA with overwhelming probability. ■

9.3.2 Large Population Sizes

We now show that a population size of $\lambda \in \Omega\big(\sqrt{n}\log(n)\big)$ leads to a run time of $\Omega\big(n \log(n)\big)$. In order to prove this, we first show that it is unlikely that too many frequencies leave the interval $[1/6, 5/6]$ quickly in this scenario. Thus, it is also unlikely to sample the optimum.

We start by proving that a single frequency does not leave $[1/6, 5/6]$ too quickly if $\mu \in \omega(1)$. We make use of Corollary 9.3 and the lemmas following from it, all of which make use of the lemmas we prove here themselves. At the end of this section, we discuss why this seemingly contradictory approach is feasible.

▶ **Lemma 9.9.** Consider an arbitrary frequency of the UMDA with $\lambda \in \omega(1)$ optimizing OneMax. For a sufficiently small constant γ, during the first at least $\gamma \cdot \min\{\mu, \sqrt{n}\}$ iterations, this frequency will not leave $[1/6, 5/6]$ with a probability of at least a constant greater than 0. ◀

Proof. We consider the expected change of an arbitrary position's frequency p_t over time t. Let X_t, again, denote the number of 1s of the μ selected individuals. Note that $p_{t+1} = X_t/\mu$.

Due to Lemma 9.5, we know that X_t is the sum of two random variables $Z_{1,t}$ and $Z_{2,t}$, where $Z_{1,t} \preceq \mathrm{Bin}(\mu, X_{t-1}/\mu)$ corresponds to the number of 1s due to the 1st-class individuals, and $Z_{2,t} \preceq \mathrm{Bin}\left(\lambda, \mathrm{O}\left(1/\sqrt{n}\right)\right) + \widetilde{Z}_t$ corresponds to the 2nd-class individuals' number of 1s, pessimistically assuming that each 2nd-class individual contributes a 1.

First, we upper-bound the probability of p_t reaching $5/6$ during $\gamma \cdot \min\{\mu, \sqrt{n}\}$ iterations. Then we do the same for reaching $1/6$. Taking the converse probability of a union bound over both cases then yields the result.

The probability of reaching 5/6. Since $Z_{1,t}$ is dominated by a martingale which we want to account for in the process, we analyze $\phi_{t+1} := (X_t/\mu)^2$, with $\phi_0 = (1/2)^2$. The original process of p_t reaching $5/6$ translates into the new process p_t^2 reaching $(5/6)^2$.

We bound the expected change during one step:

$$\begin{aligned}
\mathrm{E}[\phi_{t+1} - \phi_t \mid X_{t-1}] &= \frac{1}{\mu^2}\left(\mathrm{E}\left[X_t^2 \mid X_{t-1}\right] - X_{t-1}^2\right)\\
&= \frac{1}{\mu^2}\Big(\mathrm{E}\left[(Z_{1,t} + Z_{2,t})^2 \mid X_{t-1}\right] - X_{t-1}^2\Big)\\
&= \frac{1}{\mu^2}\left(\mathrm{E}\left[Z_{1,t}^2 \mid X_{t-1}\right] + \mathrm{E}\left[Z_{2,t}^2 \mid X_{t-1}\right]\right.\\
&\qquad \left. + 2\mathrm{E}\left[Z_{1,t} \cdot Z_{2,t} \mid X_{t-1}\right] - X_{t-1}^2\right) .
\end{aligned}$$

As discussed before, we look at the dominating distributions of $Z_{1,t}$ and

$Z_{2,t}$. Further, note that $Z_{1,t}$ and $Z_{2,t}$ are not independent but their dominating distributions are.

We calculate the different terms separately. Using that $Z_{1,t} \preceq \mathrm{Bin}(\mu, X_{t-1}/\mu)$ and that, for any two random variables Y and Y', it holds that $\mathrm{E}\big[Y^2 \,\big|\, Y'\big] = \mathrm{Var}[Y \mid Y'] + \mathrm{E}[Y \mid Y']^2$ (similar to what is discussed in Definition 2.6 and using Theorem 2.12), we get

$$\begin{aligned}\mathrm{E}\big[Z_{1,t}^2 \,\big|\, X_{t-1}\big] &\leq \mu\frac{X_{t-1}}{\mu}\left(1 - \frac{X_{t-1}}{\mu}\right) + \left(\mu\frac{X_{t-1}}{\mu}\right)^2 \\ &\leq X_{t-1} + X_{t-1}^2 \,.\end{aligned}$$

For $Z_{2,t}$, let $Z_t^* \sim \mathrm{Bin}\left(\lambda, \mathrm{O}\big(1/\sqrt{n}\big)\right)$, and let $c_1 > 0$ be a constant. Recall that $\widetilde{Z}$ is a random variable that takes values in $[\lambda]$ with probability $\mathrm{e}^{-c_1\mu}$ and is 0 otherwise. Similar to before and noting that $\mu\mathrm{e}^{-c_1\mu} \leq \mathrm{e}^{-c_2\mu}$, for a constant $c_2 > 0$, we see that there is a constant $c_3 > 0$ such that

$$\begin{aligned}\mathrm{E}\big[Z_{2,t}^2 \,\big|\, X_{t-1}\big] &\leq \mathrm{E}\big[(Z_t^*)^2 \,\big|\, X_{t-1}\big] + \mathrm{E}\big[(\widetilde{Z}_t)^2 \,\big|\, X_{t-1}\big] + 2\mathrm{E}\big[Z_t^* \,\big|\, X_{t-1}\big]\mathrm{E}\big[\widetilde{Z}_t \,\big|\, X_{t-1}\big] \\ &\leq \frac{c_3\mu}{\sqrt{n}} + \frac{c_3\mu^2}{n} + \mathrm{e}^{-c_2\mu} + \frac{c_3\mu}{\sqrt{n}}\mathrm{e}^{-c_2\mu} \\ &\leq \max\left\{\frac{c_3\mu}{\sqrt{n}}, \frac{c_3\mu^2}{n}, \mathrm{e}^{-c_2\mu}\right\},\end{aligned}$$

because the term $c_3\mu/(\sqrt{n}\mathrm{e}^{-c_2\mu})$ is always dominated by another term. Note that $c_3\mu/\sqrt{n}$ dominates if $\mu \in \mathrm{o}\big(\sqrt{n}\big)$ and $\mu \geq c_4 \ln(n)$ for a sufficiently large constant $c_4 > 0$. For $\mu \in \Omega\big(\sqrt{n}\big)$, the term $c_3\mu^2/n$ dominates. In the remaining cases (when μ is at most logarithmic), the term $\mathrm{e}^{-c_2\mu}$ dominates.

By Lemma 9.5, $X_{t-1} \leq \mu$ and $\mu \in \Omega\big(\sqrt{n}\log(n)\big)$, we get

$$\mathrm{E}\big[Z_{2,t} \,\big|\, X_{t-1}\big] \leq \max\left\{\frac{c_3\mu}{\sqrt{n}}, \mathrm{e}^{-c_2\mu}\right\},$$

where the term $\mu\mathrm{e}^{-c_1\mu}$ only dominates if $\mu \leq c_5 \ln(n)$ for a sufficiently small constant $c_5 > 0$.

Using our prior calculations and independence of the dominating distributions, we can bound

$$2\mathrm{E}\big[Z_{1,t} \cdot Z_{2,t} \,\big|\, X_{t-1}\big] \leq X_{t-1} \cdot \max\left\{\frac{c_3\mu}{\sqrt{n}}, \mathrm{e}^{-c_2\mu}\right\} .$$

Thus, we get

$$\begin{aligned}
\mathrm{E}[\phi_{t+1} - \phi_t \mid X_{t-1}] &\leq \frac{1}{\mu^2}\Bigg(X_{t-1} + X_{t-1}^2 + \max\left\{\frac{c_3\mu}{\sqrt{n}}, \frac{c_3\mu^2}{n}, \mathrm{e}^{-c_2\mu}\right\} \\
&\qquad + X_{t-1} \cdot \max\left\{\frac{c_3\mu}{\sqrt{n}}, \mathrm{e}^{-c_2\mu}\right\} - X_{t-1}^2\Bigg) \\
&\leq \frac{1}{\mu^2}\Bigg(\max\left\{\frac{c_3\mu}{\sqrt{n}}, \frac{c_3\mu^2}{n}, \mathrm{e}^{-c_2\mu}\right\} \\
&\qquad + X_{t-1}\left(1 + \max\left\{\frac{c_3\mu}{\sqrt{n}}, \mathrm{e}^{-c_2\mu}\right\}\right)\Bigg) \\
&\overset{X_{t-1} \leq \mu}{\leq} \frac{2}{\mu^2}\mu\left(1 + \max\left\{\frac{c_3\mu}{\sqrt{n}}, \mathrm{e}^{-c_2\mu}\right\}\right) \\
&\in \Theta\left(\max\left\{\frac{1}{\mu}, \frac{1}{\sqrt{n}}\right\}\right) .
\end{aligned}$$

We now look at how far ϕ gets within T iterations and want to bound the probability of it going above $(5/6)^2$. Note that

$$\phi_T = \phi_0 + \sum_{t=0}^{T-1}(\phi_{t+1} - \phi_t) .$$

Due to our bounds and Theorem 2.12, we get, for a sufficiently large constant ζ,

$$\begin{aligned}
\mathrm{E}[\phi_T] &= \left(\frac{1}{2}\right)^2 + \sum_{t=0}^{T-1}\mathrm{E}\big[\mathrm{E}[\phi_{t+1} - \phi_t \mid X_{t-1}]\big] \\
&\leq \frac{1}{4} + \zeta T \cdot \max\left\{\frac{1}{\mu}, \frac{1}{\sqrt{n}}\right\} .
\end{aligned}$$

Using Markov's inequality (Theorem 2.18) gives us, for $k > 1$,

$$\Pr\left[\phi_T \geq k\left(\frac{1}{4} + \zeta T \cdot \max\left\{\frac{1}{\mu}, \frac{1}{\sqrt{n}}\right\}\right)\right] \leq \Pr\left[\phi_T \geq k\mathrm{E}[\phi_T]\right] \\ \leq \frac{1}{k} .$$

We want that $(5/6)^2 \geq k\left(1/4 + \zeta T \cdot \max\{1/\mu, 1/\sqrt{n}\}\right)$ holds, since then $\Pr\left[\phi_T \geq (5/6)^2\right]$ is upper-bounded by $\Pr\left[P_T \geq k(1/4 + \zeta T \cdot \max\{1/\mu, 1/\sqrt{n}\})\right] \leq 1/k$, which we want to be less than $1/2$ in order to apply a meaningful union bound over both cases at the end of this proof. Hence, assume $k > 2$. We get

$$\left(\frac{5}{6}\right)^2 \geq k\left(\frac{1}{4} + \zeta T \cdot \max\left\{\frac{1}{\mu}, \frac{1}{\sqrt{n}}\right\}\right) \Leftrightarrow T \leq \left(\frac{25}{36k} - \frac{1}{4}\right)\frac{\min\{\mu, \sqrt{n}\}}{\zeta} ,$$

which is positive as long as $k < 25/9$. Thus, we can bound $k \in (2, 25/9)$.

Therefore, if $T \leq \gamma \cdot \min\{\mu, \sqrt{n}\}$, for a sufficiently small constant $\gamma > 0$, then the probability of an arbitrary frequency exceeding $5/6$ is at most a constant less than $1/2$ (for $k \in (2, 25/9)$).

The probability of reaching $1/6$. We now analyze how likely it is that p_t hits $1/6$ in a similar amount of time. For this case, we define a slightly different potential $\phi'_{t+1} := (1 - X_t/\mu)^2 = 1 - 2X_t/\mu + (X_t/\mu)^2$, that is, we mirror the process at $1/2$ and then use the same potential as before.

Looking at the expected difference during one step, we see that

$$\begin{aligned}\mathrm{E}\left[\phi'_{t+1} - \phi'_t \mid X_{t-1}\right] &= 1 - 2\frac{\mathrm{E}[X_t \mid X_{t-1}]}{\mu} + \left(\frac{\mathrm{E}[X_t \mid X_{t-1}]}{\mu}\right)^2 \\ &\quad - 1 + 2\frac{X_{t-1}}{\mu} - \left(\frac{X_{t-1}}{\mu}\right)^2 \\ &= \frac{2}{\mu}(X_{t-1} - \mathrm{E}[X_t \mid X_{t-1}]) + \mathrm{E}[\phi_{t+1} - \phi_t \mid X_{t-1}] ,\end{aligned}$$

where we only have to determine the expected value of $X_{t-1} - \mathrm{E}[X_t \mid X_{t-1}]$, because we already analyzed $\mathrm{E}[\phi_{t+1} - \phi_t \mid X_{t-1}]$ before.

Considering just the 1st-class individuals, it holds that $\mathrm{E}[X_t \mid X_{t-1}] = X_{t-1}$, because we then have a martingale. But due to the elitist selection of the UMDA, actually $\mathrm{E}[X_t \mid X_{t-1}] \geq X_{t-1}$ holds, because of the bias of the 2nd-class individuals, which prefer 1s over 0s. Thus, $\mathrm{E}[X_{t-1} - X_t \mid X_{t-1}] \leq 0$, and we get

$$\mathrm{E}\big[\phi'_{t+1} - \phi'_t \,\big|\, X_{t-1}\big] \leq \mathrm{E}[\phi_{t+1} - \phi_t \mid X_{t-1}]\,,$$

which we already analyzed.

Hence, we can argue analogously as before and get, again, a probability of at most a constant less than $1/2$ to reach $1/6$ during at most $\gamma \cdot \min\{\mu, \sqrt{n}\}$ iterations. Taking a union bound over both cases concludes the proof. ■

We now expand the case from a single frequency to all frequencies.

▶ **Lemma 9.10.** For a sufficiently small constant γ, during the first at least $\gamma \cdot \min\{\mu, \sqrt{n}\}$ iterations of the UMDA optimizing OneMax, it holds that $\Theta(n)$ frequencies stay in the interval $[1/6, 5/6]$ with at least constant probability. ◀

Proof. We look at $T \leq \gamma \cdot \min\{\mu, \sqrt{n}\}$ iterations. Thus, the probability for a single frequency to leave $[1/6, 5/6]$ is at most a constant $c < 1$, according to Lemma 9.9. In expectation, there are at most cn frequencies outside of $[1/6, 5/6]$, and due to Markov's inequality (Theorem 2.18), for a constant $\delta > 0$ with $(1+\delta)c < 1$, the probability that there are at least $(1+\delta)cn$ such frequencies is at most $1/(1+\delta)$. This means that with at least constant probability, at least $\big(1 - c(1+\delta)\big)n \in \Theta(n)$ frequencies are still within $[1/6, 5/6]$. ■

Note that the proof of Lemma 9.9 relies on Corollary 9.3, and the proof of Corollary 9.3 also relies on Lemma 9.9. Formally, this cyclic dependency can be solved by proving both propositions in conjunction via induction over the number of iterations up to $\gamma \cdot \min\{\mu, \sqrt{n}\}$, for a sufficiently small constant γ. For the base case, all frequencies are at $1/2 \in [1/6, 5/6]$, and both propositions hold. For the inductive step, assuming that $t < \gamma \cdot \min\{\mu, \sqrt{n}\}$, we already now that both propositions hold up to iteration t. Thus, the requirements for the proofs of Corollary 9.3 and Lemma 9.9 are fulfilled, and the proofs themselves pass.

We now prove an easy lower bound.

▶ **Corollary 9.11.** Consider the UMDA with $\mu \in \Omega\big(\sqrt{n}\log(n)\big)$ optimizing OneMax. Its run time is then in $\Omega\big(n\log(n)\big)$ in expectation and with at least constant probability. ◀

Proof. Since we assume $\mu \in \Omega\big(\sqrt{n}\log(n)\big)$, Lemma 9.10 yields that within at most $\gamma \cdot \min\{\mu, \sqrt{n}\} = \gamma\sqrt{n}$ iterations, γ sufficiently small, at least $c_1 n$ frequencies, with $c_1 > 0$ being a constant, are at most $5/6$ with probability $\Omega(1)$. Hence, assuming this to happen, the probability to sample the optimum is at most $(5/6)^{c_1 n} \leq \mathrm{e}^{-c_2 n}$, where $c_2 > 0$ is a constant. Thus, the expected run time is at least $\gamma\sqrt{n}\lambda \in \Omega\bigl(n\log(n)\bigr)$. ■

9.3.3 Medium Population Sizes

In this section, we consider the remaining population sizes of $\mu \in \mathrm{O}\bigl(\sqrt{n}\log(n)\bigr) \cap \Omega\bigl(\log(n)\bigr)$, where we recall that $\lambda = (1+\beta)\mu$. Basically, we lower-bound the probability that a single frequency hits $1/n$. In order to do so, we analyze the one-step change of the number of 1s at the frequency's position and approximate it via a normal distribution. For this, we use a general form of the central limit theorem (CLT), along with a bound on the approximation error.

▶ **Lemma 9.12 (CLT With Berry-Esseen Inequality [Fel71, Chapter XVI.5, Theorem 2]).** Let $m \in \mathbf{N}^+$, and let $(X_i)_{i\in[m]}$ be a sequence of independent random variables, where, for each $i \in [m]$, the random variable X_i has finite expected value μ_i, variance σ_i^2, and third central moment. Further, define

$$s_m^2 := \sum_{i=1}^{m} \sigma_i^2 \quad \text{and} \quad C_m := \frac{1}{s_m}\sum_{i=1}^{m}(X_i - \mu_i)\,.$$

Then the distribution of C_m can be approximated by the standard normal distribution, and for the approximation error, it holds for all $x \in \mathbf{R}$ that

$$\bigl|\Pr[C_m \leq x] - \Phi(x)\bigr| \leq C \cdot \frac{\sum_{i=1}^{m} \mathrm{E}\bigl[|X_i - \mu_i|^3\bigr]}{s_m^3}\,,$$

where $C > 0$ is a constant and $\Phi(x)$ denotes the cumulative distribution function of the standard normal distribution. ◀

Note that the approximation error in Lemma 9.12 is vanishing if

$$\lim_{m\to\infty} \frac{1}{s_m^3}\sum_{i=1}^{m} \mathrm{E}\bigl[|X_i - \mu_i|^3\bigr] = 0 \tag{9.2}$$

holds. In this case, C_m converges in distribution to the standard normal distribution [Bil95, Theorem 27.3]. Equation (9.2) is called the *Lyapunov condition.*

In order to make use of Lemma 9.12, we need to study the stochastic process on the X_t values (which, again, denotes the number of 1s of an arbitrary position) and determine the accumulated expectations and variances of every single one-step change. Using the notation from Lemma 9.5, we note that the X_t value in expectation changes very little from one step to the next. However, considerable variances are responsible for changes of the X_t value, and it turns out that the variances are heavily dependent on the current state. We get $\mathrm{Var}\big[Z_{1,t} \bigm| X_{t-1}\big] \le X_{t-1}(1 - X_{t-1}/\mu)$, that is, if $X_{t-1} \le \mu/2$, then the 1st-class individuals are responsible for a typical deviation of $\sqrt{X_{t-1}}$. This dependency of $\mathrm{Var}\big[Z_{1,t} \bigm| X_{t-1}\big]$ on X_{t-1} makes a direct application of Lemma 9.12 difficult.

In order to make Lemma 9.12 applicable, we define a potential function that transforms X_{t-1} such that the expected difference between two points in time is still close to 0, but the variance is independent of the state. This potential function is inspired by the approach used by Sudholt and Witt [SW16a] in order to analyze two very simple EDAs. Since the standard deviation of $Z_{1,t}$ is $\Theta\big(\sqrt{X_{t-1}}\big)$, we work with a potential function whose slope at point X_{t-1} is $\Theta\big(1/\sqrt{X_{t-1}}\big)$, so that the dependency of the variance on the state cancels out.

We proceed with the formal definition. Let g denote the potential function, defined over $[0..\mu]$. Our definition is simpler than the one from Sudholt and Witt [SW16a], as we do not need g to be centrally symmetric around $\mu/2$. We define

$$g(x) \coloneqq \sqrt{\mu} \sum_{j=x}^{\mu-1} \frac{1}{\sqrt{j+1}} \,.$$

Since we are interested in the one-step change in potential for a point in time $t \in \mathbf{N}$, let $\Delta_t \coloneqq g(X_{t+1}) - g(X_t)$.

We will often use the following bounds on the change of potential. For $x, y \in [0..\mu]$ with $y < x$, we get

$$g(y) - g(x) = \sqrt{\mu} \sum_{j=y}^{x-1} \frac{1}{\sqrt{j+1}} \le \sqrt{\mu} \frac{x-y}{\sqrt{y+1}} \,, \text{ and} \tag{9.3}$$

$$g(y) - g(x) = \sqrt{\mu} \sum_{j=y}^{x-1} \frac{1}{\sqrt{j+1}} \ge \sqrt{\mu} \frac{x-y}{\sqrt{x+1}} \,. \tag{9.4}$$

Bounding the Expected Change in Potential

We start by bounding the expected value of Δ_t and see that also the transformed process moves very little in expectation (however, its variance will be large, as shown in the following subsection). Because of the Lyapunov condition (equation (9.2)), which we address in Section 9.3.3, we do so in both directions.

▶ **Lemma 9.13.** Let $\mu \in \mathrm{O}\big(\sqrt{n}\log(n)\big)$. Then, for all $t \in \mathbf{N}$ and all $X_t \in [\mu - 1]$, there are constants $c_1, c_2 > 0$ such that

$$\mathrm{E}[\Delta_t \mid X_t] \geq -\left(\mathrm{e}^{-c_1\mu} + c_2\left(\frac{X_t}{\mu} + \frac{X_t}{\sqrt{n}}\right)\right)\sqrt{\frac{\mu}{X_t + 1}} \text{ and}$$
$$\mathrm{E}[\Delta_t \mid X_t] \leq 120\sqrt{\frac{\mu}{X_t}}\ . \qquad ◀$$

Proof. Consider an iteration $t \in \mathbf{N}$. We abbreviate $X_t = x$.

The lower bound. First, we derive the lower bound. We have $\mathrm{E}[\Delta_t \mid x] = \mathrm{E}[g(X_{t+1}) \mid x] - g(x)$. Due to Lemma 9.5, we get that there are constants $c_1, c_2 > 0$ such that

$$\mathrm{E}[X_{t+1} \mid x] \leq x + \mathrm{e}^{-c_1\mu} + c_2\left(\frac{x}{\mu} + \frac{x}{\sqrt{n}}\right).$$

Since g is convex, we get by Jensen's inequality (Theorem 2.20) that

$$\begin{aligned}\mathrm{E}[g(X_{t+1}) \mid x] - g(x) &\geq g\big(\mathrm{E}[X_{t+1} \mid x]\big) - g(x)\\ &\geq g\left(x + \mathrm{e}^{-c_1\mu} + c_2\left(\frac{x}{\mu} + \frac{x}{\sqrt{n}}\right)\right) - g(x)\ .\end{aligned}$$

Applying equation (9.3) gives us the desired result of

$$g\left(x + \mathrm{e}^{-c_1\mu} + c_2\left(\frac{x}{\mu} + \frac{x}{\sqrt{n}}\right)\right) - g(x) \geq -\left(\mathrm{e}^{-c_1\mu} + c_2\left(\frac{x}{\mu} + \frac{x}{\sqrt{n}}\right)\right)\sqrt{\frac{\mu}{x+1}}\ .$$

The upper bound. We show the upper bound by ignoring 2nd-class individuals, since they are biased toward increasing x and, therefore, decreasing Δ_t. Hence, we now assume that X_{t+1} follows a binomial distribution with parameters μ and x/μ, that is, $\mathrm{E}[X_{t+1} - x \mid x] = 0$. In a delicate analysis, we estimate how much $\mathrm{E}[\Delta_t \mid x]$ is shifted away from 0 due to the non-linearity of the potential function. We use the inequalities

$$\begin{aligned} g(i) &\le g(x) + \frac{\sqrt{\mu}(x-i)}{\sqrt{i+1}} \quad \text{for } i < x, \text{ and} \\ g(i) &\le g(x) + \frac{\sqrt{\mu}(x-i)}{\sqrt{i+1}} \quad \text{for } i > x, \end{aligned}$$

which are just rearrangements of equations (9.3) and (9.4), noting that $x - i$ is negative in the second inequality. We estimate

$$\begin{aligned} \mathrm{E}[\Delta_t \mid x] &= \sum_{i=0}^{\mu} \big(g(i) - g(x)\big)\Pr[X_{t+1} = i \mid x] \\ &\le \sum_{i=0}^{x-1} \left(g(x) + \frac{\sqrt{\mu}(x-i)}{\sqrt{i+1}} - g(x)\right)\Pr[X_{t+1} = i \mid x] \\ &\quad + \sum_{i=x+1}^{\mu} \left(g(x) + \frac{\sqrt{\mu}(x-i)}{\sqrt{i+1}} - g(x)\right)\Pr[X_{t+1} = i \mid x] \\ &= \sum_{i=0}^{\infty} \left(\frac{\sqrt{\mu}(x-i)}{\sqrt{i+1}}\Pr[X_{t+1} = i \mid x]\right) . \end{aligned}$$

We now split the set of possible outcomes of i into intervals of length $\sqrt{x}$. More precisely, for $k \in \mathbf{N}$, we define $I_k := \{\lceil x - (k+1)\sqrt{x}\rceil, \dots, \lfloor x - k\sqrt{x}\rfloor\}$. The points in these intervals are all less than or equal to x. In order to cover the outcomes above x when considering some $i \in I_k$, we consider the points i and $2x - i$ together, exploiting that they are mirrors of each other of distance $x - i$ to x, more formally $x - i = -(x - (2x - i))$. Plugging in i and $2x - i$ for $i \in I_k$ and summing over all $k \ge 0$, we obtain

$$\begin{aligned} \mathrm{E}[\Delta_t \mid x] \le \sum_{k=0}^{\infty} \sum_{i \in I_k} &\left(\frac{\sqrt{\mu}(x-i)}{\sqrt{i+1}}\Pr[X_{t+1} = i \mid x]\right. \\ &\qquad \left. + \frac{\sqrt{\mu}(i-x)}{\sqrt{2x-i+1}}\Pr[X_{t+1} = 2x - i \mid x]\right) \\ \le \sum_{k=0}^{\infty} \sum_{i \in I_k} &\left(\frac{\sqrt{\mu}(x-i)}{\sqrt{x-(k+1)\sqrt{x}+1}}\Pr[X_{t+1} = i \mid x]\right. \end{aligned}$$

$$-\frac{\sqrt{\mu}(x-i)}{\sqrt{x+(k+1)\sqrt{x}+1}}\Pr[X_{t+1}=2x-i \mid x]\Bigg),$$

where the last inequality used that the choice $i = x-(k+1)\sqrt{x}$ maximizes both the positive and the negative term in the inner sum.

We take special care of intervals where $x-(k+1)\sqrt{x} \leq x/2$ (that is, $k \geq \sqrt{x}/2-1$) and handle them directly. The maximum increase in potential is observed when $X_{t+1}=0$ and equals

$$\begin{aligned}\sqrt{\mu}\sum_{j=0}^{x-1}\frac{1}{\sqrt{j+1}} &\leq \sqrt{\mu}\left(1+\int_1^x \frac{1}{\sqrt{z}}\,\mathrm{d}z\right)\\ &= \sqrt{\mu}(1+2\sqrt{x}-2\sqrt{1})\\ &\leq \sqrt{4\mu x}\,,\end{aligned}$$

where the first inequality approximates the sum by an integral by noting that $1/\sqrt{z}$ is monotonically decreasing in z [Cor+09, Inequality (A.12)].

By Chernoff bounds (Theorem 2.19), the probability of $X_{t+1} \leq x/2$ is at most $\mathrm{e}^{-x/24}$. Hence, the intervals of index at least $k_{\max} := \sqrt{x}/2-1$ contribute only a term of $S^* := \sqrt{4\mu x}\mathrm{e}^{-x/24} \leq 100\sqrt{\mu/x}$ to $\mathrm{E}[\Delta_t \mid x]$.[21]

For smaller k, we argue more precisely. Note that, for $a \geq b > 0$, it holds that $a-b \leq (a^2-b^2)/2b$. Now, since

$$\begin{aligned}\frac{\sqrt{x+(k+1)\sqrt{x}+1}}{\sqrt{x-(k+1)\sqrt{x}+1}} &= 1+\frac{\sqrt{x+(k+1)\sqrt{x}+1}-\sqrt{x-(k+1)\sqrt{x}+1}}{\sqrt{x-(k+1)\sqrt{x}+1}}\\ &\leq 1+\frac{\frac{2(k+1)\sqrt{x}}{2\sqrt{x-(k+1)\sqrt{x}+1}}}{\sqrt{x-(k+1)\sqrt{x}+1}}\\ &= 1+\frac{(k+1)\sqrt{x}}{x-(k+1)\sqrt{x}+1}\,,\end{aligned}$$

we have

$$\mathrm{E}[\Delta_t \mid x]$$

[21] The inequality $2x\mathrm{e}^{-x/24} \leq 100/\sqrt{x}$ for $x \geq 1$ can be checked using elementary calculus.

$$\leq \sum_{k=0}^{k_{\max}} \sum_{i \in I_k} \left(\frac{\sqrt{\mu}(x-i)}{\sqrt{x+(k+1)\sqrt{x}+1}} \left(1 + \frac{(k+1)\sqrt{x}}{x-(k+1)\sqrt{x}+1}\right) \Pr[X_{t+1} = i \mid x] \right.$$
$$\left. - \frac{\sqrt{\mu}(x-i)}{\sqrt{x+(k+1)\sqrt{x}+1}} \Pr[X_{t+1} = 2x - i \mid x] \right) + S^* \; . \tag{9.5}$$

We now look more closely into the inner sum and work with the abbreviation

$$E_k^* := \sum_{i \in I_k} \Big((x-i) \cdot \Pr[X_{t+1} = i] - (x-i)\Pr[X_{t+1} = 2x - i] \Big) \; .$$

Coming back to inequality (9.5), this enables us to estimate the inner sum for arbitrary k by

$$\begin{aligned}
&\sum_{i \in I_k} \left(\frac{\sqrt{\mu}(x-i)}{\sqrt{x+(k+1)\sqrt{x}+1}} \left(1 + \frac{(k+1)\sqrt{x}}{x-(k+1)\sqrt{x}+1}\right) \Pr[X_{t+1} = i \mid x] \right. \\
&\qquad \left. - \frac{\sqrt{\mu}(x-i)}{\sqrt{x+(k+1)\sqrt{x}+1}} \Pr[X_{t+1} = 2x - i \mid x] \right) \\
&= E_k^* \cdot \frac{\sqrt{\mu}}{\sqrt{x+(k+1)\sqrt{x}+1}} \\
&\quad + \sum_{i \in I_k} \frac{\sqrt{\mu}(x-i)}{\sqrt{x+(k+1)\sqrt{x}+1}} \cdot \frac{(k+1)\sqrt{x}}{x-(k+1)\sqrt{x}+1} \Pr[X_{t+1} = i \mid x] \\
&\leq \frac{E_k^* \sqrt{\mu}}{\sqrt{x+(k+1)\sqrt{x}+1}} + \sum_{i \in I_k} \frac{\sqrt{\mu}(x-i)(k+1)}{x-(k+1)\sqrt{x}+1} \Pr[X_{t+1} = i \mid x] \; ,
\end{aligned}$$

where the last inequality estimated $\sqrt{x}/\sqrt{x+(k+1)\sqrt{x}+1} \leq 1$. Since $k \leq k_{\max}$, that is, $(k+1)\sqrt{x} \leq x/2$, the last bound is easily bounded from above by

$$\frac{E_k^* \sqrt{\mu}}{\sqrt{x+(k+1)\sqrt{x}+1}} + \sum_{i \in I_k} \frac{\sqrt{\mu}(x-i)(k+1)}{\frac{x}{2}} \Pr[X_{t+1} = i \mid x] \; .$$

We proceed by bounding the sum over I_k, noting that we have, by Chernoff bounds (Theorem 2.19),

$$\begin{aligned}\Pr[X_{t+1} \in I_k \mid x] &\leq \Pr\left[X_{t+1} \leq x - k\sqrt{x} \mid x\right] \\ &\leq \mathrm{e}^{-\frac{k^2}{3}} .\end{aligned}$$

Hence, since $x - i \leq (k+1)\sqrt{x}$ for $i \in I_k$, we get

$$\begin{aligned}\sum_{i \in I_k} \frac{\sqrt{\mu}(x-i)(k+1)}{\frac{x}{2}} &\leq \frac{2\sqrt{\mu}}{x} \sum_{i \in I_k} (k+1)^2 \sqrt{x} \Pr[X_{t+1} = i \mid x] \\ &\leq \frac{2\sqrt{\mu}(k+1)^2}{\sqrt{x}} \sum_{i \in I_k} \Pr[X_{t+1} = i \mid x] \\ &\leq \frac{2\sqrt{\mu}(k+1)^2 \mathrm{e}^{-\frac{k^2}{3}}}{\sqrt{x}} .\end{aligned}$$

Altogether, we have obtained from inequality (9.5) the simpler inequality

$$\mathrm{E}[\Delta_t \mid x] \leq \sum_{k=0}^{k_{\max}} \left(\frac{E_k^* \sqrt{\mu}}{\sqrt{x + (k+1)\sqrt{x} + 1}} + \frac{2\sqrt{\mu}(k+1)^2 \mathrm{e}^{-\frac{k^2}{3}}}{\sqrt{x}} \right) + S^* , \tag{9.6}$$

which we will bound further. The idea is to exploit that

$$\sum_{k \geq 0} E_k^* = 0 , \tag{9.7}$$

which is a consequence of $\mathrm{E}[X_{t+1} \mid x] = x$, since

$$\begin{aligned}0 &= \mathrm{E}[X_{t+1} \mid x] - x \\ &= \sum_{k \geq 0} \sum_{i \in I_k} \Big((i - x) \cdot \Pr[X_{t+1} = i \mid x] + ((2x - i) - x) \Pr[X_{t+1} = 2x - i \mid x] \Big) \\ &= \sum_{k \geq 0} E_k^* .\end{aligned}$$

Using similar calculations as above, we manipulate the sum

$$\sum_{k\geq 0} \frac{E_k^* \sqrt{\mu}}{\sqrt{x + (k+1)\sqrt{x} + 1}}$$

from the upper bound of inequality (9.6) and recognize that, using again $a - b \leq (a^2 - b^2)/2b$ for $a \geq b > 0$, it is equal to

$$\begin{aligned}
&\sum_{k\geq 0} \frac{E_k^* \sqrt{\mu}}{\sqrt{x + \sqrt{x} + 1}} \cdot \left(1 + \frac{\sqrt{x + \sqrt{x} + 1} - \sqrt{x + (k+1)\sqrt{x} + 1}}{\sqrt{x + (k+1)\sqrt{x} + 1}}\right) \\
&\leq \sum_{\substack{k\geq 0 \\ E_k^* < 0}} \frac{E_k^* \sqrt{\mu}}{\sqrt{x + \sqrt{x} + 1}} \left(1 - \frac{k\sqrt{x}}{2\sqrt{x + (k+1)\sqrt{x} + 1}\sqrt{x + \sqrt{x} + 1}}\right) \\
&\quad + \sum_{\substack{k\geq 0 \\ E_k^* \geq 0}} \frac{E_k^* \sqrt{\mu}}{\sqrt{x + \sqrt{x} + 1}} \cdot 1 \,.
\end{aligned}$$

Similarly as above, we get, using Chernoff bounds (Theorem 2.19),

$$\begin{aligned}
E_k^* &\geq \sum_{i = x + k\sqrt{x}}^{x + (k+1)\sqrt{x}} (x - i) \Pr[X_{t+1} = i] \\
&\geq -2(k+1) \mathrm{e}^{-\frac{k^2}{3}} \sqrt{x} \,.
\end{aligned}$$

Combining this with equation (9.7), we arrive at the inequality

$$\begin{aligned}
&\sum_{k\geq 0} \frac{E_k^* \sqrt{\mu}}{\sqrt{x + (k+1)\sqrt{x} + 1}} \\
&\leq \sum_{\substack{k\geq 0 \\ E_k^* < 0}} \frac{E_k^* \sqrt{\mu}}{\sqrt{x + \sqrt{x} + 1}} \left(1 - \frac{k\sqrt{x}}{2\sqrt{x + (k+1)\sqrt{x} + 1}\sqrt{x + \sqrt{x} + 1}}\right) \\
&\leq \sum_{k\geq 0} \frac{2(k+1)\mathrm{e}^{-\frac{k^2}{3}} \sqrt{x}\sqrt{\mu}}{\sqrt{x + \sqrt{x} + 1}} \cdot \frac{k\sqrt{x}}{2\sqrt{x + (k+1)\sqrt{x} + 1}\sqrt{x + \sqrt{x} + 1}} \,,
\end{aligned}$$

which is at most

$$\sum_{k\geq 0} \frac{k(k+1)\mathrm{e}^{-k^2/3}\sqrt{\mu}}{\sqrt{x}} .$$

Substituting this into inequality (9.6), we finally obtain

$$\begin{aligned} \mathrm{E}[\Delta_t \mid x] &\leq \sum_{k\geq 0}\left(\frac{2(k+1)^2\mathrm{e}^{-\frac{k^2}{3}}\sqrt{\mu}}{\sqrt{x}} + \frac{k(k+1)\mathrm{e}^{-\frac{k^2}{3}}\sqrt{\mu}}{\sqrt{x}}\right) + S^* \\ &\leq 20\frac{\sqrt{\mu}}{\sqrt{x}} + \frac{100\sqrt{\mu}}{\sqrt{x}} \\ &= 120\frac{\sqrt{\mu}}{\sqrt{x}} , \end{aligned}$$

where the bound $\sum_{k=0}^{\infty}(2(k+1)^2 + k(k+1))\mathrm{e}^{-k^2/3} \leq 20$ was obtained numerically. This finally proves the upper bound on $\mathrm{E}[\Delta_t \mid x]$. ■

Lower Bound on the Variance of the Potential Change

Before we analyze the variance of Δ_t, we introduce a lemma that we are going to use.

▶ **Lemma 9.14 ([OW15, Lemma 6]).** Let $X \sim \mathrm{Bin}(\mu, r/\mu)$ with $r \in [1, \mu-1]$, let $\ell = \min\{r, \mu - r\}$, and let $\zeta > 0$ be an arbitrary constant. Then

$$\Pr\left[X \geq \mathrm{E}[X] + \zeta\sqrt{\ell}\right] \in \Omega(1) .$$

Note that if $r \leq \mu/2$, we get

$$\Pr\left[X \geq \mathrm{E}[X] + \zeta\sqrt{\mathrm{E}[X]}\right] \in \Omega(1) .$$ ◀

Oliveto and Witt [OW15] only state the lemma for $\zeta = 1$. However, introducing the constant factor does not change the lemmas's proof at all.

With Lemma 9.14 in place, we now lower-bound the variance of Δ_t. Note that the following lemma only applies up to $X_t \leq (5/6)\mu$, which will be guaranteed in its application.

▶ **Lemma 9.15.** Let $\mu \in \omega(1)$ and $\mu \in \mathrm{O}\big(\sqrt{n}\log(n)\big)$. Then, for all $t \in \mathbf{N}$ and $X_t \in [(5/6)]$, it holds that

$$\mathrm{Var}[\Delta_t \mid X_t] \in \Omega(\mu)\ .$$ ◀

Proof. Consider an iteration $t \in \mathbf{N}$. Again, we abbreviate $X_t = x$. We lower-bound $\mathrm{E}[\Delta_t \mid x]$ from Lemma 9.13 by $E^* := -\big(1 + c_2(x/\sqrt{n} + 1)\big) \cdot \sqrt{\mu/(x+1)}$, where we pessimistically estimated $\mathrm{e}^{-c_1\mu} \leq 1$ and $x/\mu \leq 1$ because $x \leq \mu$. We estimate

$$\begin{aligned}\mathrm{Var}[\Delta_t \mid x] &= \mathrm{E}\Big[\big(\Delta_t - \mathrm{E}[\Delta_t \mid x]\big)^2 \,\Big|\, x\Big]\\ &\geq \mathrm{E}\Big[\big(\Delta_t - \mathrm{E}[\Delta_t \mid x]\big)^2 \cdot \mathbf{1}\{\Delta_t < E^*\} \,\Big|\, x\Big]\\ &\geq \mathrm{E}\big[(\Delta_t - E^*)^2 \cdot \mathbf{1}\{\Delta_t < E^*\} \,\big|\, x\big]\ .\end{aligned}$$

Note that we can ignore 2nd-class individuals, as they would only increase X_{t+1} even further, leading to a greater difference of Δ_t and E^*.

We derive a sufficient condition for $\Delta_t < E^*$. For this, we introduce the constant ζ and claim that $g(x + \zeta\sqrt{x}) \leq g(x) + E^*$ if ζ is sufficiently large. This claim is equivalent to

$$g(x) - g(x + \zeta\sqrt{x}) \geq -E^*\ . \tag{9.8}$$

We lower-bound the left-hand side as follows, assuming that ζ is sufficiently large and using equation (9.4):

$$\begin{aligned}g(x) - g(x + \zeta\sqrt{x}) &\geq \sqrt{\mu} \cdot \frac{\zeta\sqrt{x}}{\sqrt{x + \zeta\sqrt{x} + 1}}\\ &\geq \sqrt{\mu} \cdot \frac{\zeta\sqrt{x}}{\sqrt{2\zeta x}}\\ &= \sqrt{\frac{\mu\zeta}{2}}\ ,\end{aligned}$$

and we want this to be at least $-E^*$.

The inequality $\sqrt{\mu\zeta/2} \geq -E^*$ is equivalent to

$$\sqrt{\frac{\zeta}{2}} \cdot \sqrt{x+1} - 1 \geq c_2\left(\frac{x}{\sqrt{n}} + 1\right) .$$

We prove this inequality by lower-bounding the left-hand side, assuming that ζ is sufficiently large, and get

$$\sqrt{\frac{\zeta}{2}} \cdot \sqrt{x+1} - 1 \geq \frac{\sqrt{\zeta x}}{2} .$$

It is now evident that $\sqrt{\zeta x}/2 \geq c_2(x/\sqrt{n} + 1) \Leftrightarrow \sqrt{\zeta}/2 \geq c_2(\sqrt{x/n} + 1/\sqrt{x})$ holds (for $x \neq 0$) if ζ is sufficiently large, that is, if $\zeta \geq (4c_2)^2$, because $x \leq \mu$ and we assume $\mu \in \mathrm{O}\big(\sqrt{n}\log(n)\big)$, thus, $\sqrt{x/n} + 1/\sqrt{x} \leq 1 + 1$. For $x = 0$, the inequality trivially holds.

Using inequality (9.8), which we just derived, we get

$$\begin{aligned} \Delta_t < E^* &\Leftrightarrow g(X_{t+1}) - g(x) < E^* \\ &\Leftrightarrow g(X_{t+1}) < g(x) + E^* \\ &\Leftarrow g(X_{t+1}) < g(x + \zeta\sqrt{x}) \\ &\Leftrightarrow X_{t+1} > x + \zeta\sqrt{x} , \end{aligned}$$

where we used the definition of g and that it is a decreasing function.

We proceed by estimating $\mathrm{E}\big[(\Delta_t - E^*)^2 \cdot \mathbf{1}\{\Delta_t < E^*\} \bigm| x\big]$. First, we see that, assuming $X_{t+1} > x + \zeta\sqrt{x}$ and using inequality (9.8) as well as equation (9.4), we have

$$\begin{aligned} \Delta_t - E^* &= g(X_{t+1}) - \big(g(x) + E^*\big) \\ &\leq g(X_{t+1}) - g(x + \zeta\sqrt{x}) \\ &= -\sqrt{\mu} \sum_{j=x+\zeta\sqrt{x}}^{X_{t+1}-1} \frac{1}{\sqrt{j+1}} . \end{aligned}$$

Note that we derive an upper bound of $\Delta_t - E^*$ because we only consider $\Delta_t < E^*$, that is, $\Delta_t - E^* < 0$. Thus, its square gets minimized for an upper bound.

Since $X_{t+1} > x + \zeta\sqrt{x}$ implies $\Delta_t < E^*$, we get, using Jensen's inequality

(Theorem 2.20), that

$$
\begin{aligned}
&\mathrm{E}\left[(\Delta_t - E^*)^2 \cdot \mathbf{1}\{\Delta_t < E^*\} \,\middle|\, x\right] \\
&\geq \mathrm{E}\left[(\Delta_t - E^*)^2 \cdot \mathbf{1}\{X_{t+1} > x + \zeta\sqrt{x}\} \,\middle|\, x\right] \\
&\geq \mathrm{E}\left[\left(g(X_{t+1}) - g(x + \zeta\sqrt{x})\right) \cdot \mathbf{1}\{X_{t+1} > x + \zeta\sqrt{x}\} \,\middle|\, x\right]^2 \\
&= \left(\sum_{i=0}^{\mu}(-\sqrt{\mu}) \sum_{j=x+\zeta\sqrt{x}}^{i-1} \frac{1}{\sqrt{j+1}} \cdot \mathbf{1}\{i > x + \zeta\sqrt{x}\}\Pr[X_{t+1} = i \mid x]\right)^2 \\
&= \mu\left(\sum_{i=x+\zeta\sqrt{x}+1}^{\mu} \sum_{j=x+\zeta\sqrt{x}}^{i-1} \frac{1}{\sqrt{j+1}} \Pr[X_{t+1} = i \mid x]\right)^2 .
\end{aligned}
$$

We now derive a lower bound for the inner sum. Using equation (9.4), we get

$$
\sum_{j=x+\zeta\sqrt{x}}^{i-1} \frac{1}{\sqrt{j+1}} \geq \frac{i - x - \zeta\sqrt{x}}{\sqrt{i}} .
$$

Substituting this back into the expectation gives us

$$
\begin{aligned}
&\mu\left(\sum_{i=x+\zeta\sqrt{x}+1}^{\mu} \sum_{j=x+\zeta\sqrt{x}}^{i-1} \frac{1}{\sqrt{j+1}} \Pr[X_{t+1} = i \mid x]\right)^2 \\
&\geq \mu\left(\sum_{i=x+\zeta\sqrt{x}+1}^{\mu} \frac{i - x - \zeta\sqrt{x}}{\sqrt{i}} \Pr[X_{t+1} = i \mid x]\right)^2 \\
&\geq \mu\left(\sum_{i=x+2\zeta\sqrt{x}+1}^{\mu} \frac{i - x - \zeta\sqrt{x}}{\sqrt{i}} \Pr[X_{t+1} = i \mid x]\right)^2 ,
\end{aligned}
$$

where we narrowed the range for i. In this new range, the term $(i - x - \zeta\sqrt{x})/\sqrt{i}$ is monotonically increasing with respect to i and hence minimal for $i = x + 2\zeta\sqrt{x} + 1$ in the range $[x + 2\zeta\sqrt{x} + 1..\mu]$. We lower-bound this term by

$$
\frac{x + 2\zeta\sqrt{x} + 1 - x - \zeta\sqrt{x}}{\sqrt{x + 2\zeta\sqrt{x} + 1}} = \frac{\zeta\sqrt{x} + 1}{\sqrt{x + 2\zeta\sqrt{x} + 1}}
$$

$$\begin{aligned} &\geq \frac{\zeta\sqrt{x}+1}{\sqrt{3\zeta x}} \\ &= \sqrt{\frac{\zeta}{3}} + \frac{1}{\sqrt{3\zeta x}} \\ &\in \Omega(1)\,. \end{aligned}$$

Hence, we finally have

$$\begin{aligned} \mathrm{Var}[\Delta_t \mid x] &\in \Omega\left(\mu\left(\sum_{i=x+2\zeta\sqrt{x}+1}^{\mu} \Pr[X_{t+1} = i \mid x]\right)^2\right) \\ &\subseteq \Omega\left(\mu\Pr\left[X_{t+1} \geq x + 2\zeta\sqrt{x} + 1 \,\middle|\, x\right]^2\right) \\ &\subseteq \Omega(\mu)\,, \end{aligned}$$

where the last inequality used Lemma 9.14 to lower-bound the probability. The lemma can be used immediately for $x \leq \mu/2$. Otherwise, we still have $x \leq (5/6)\mu$ by assumption. Then Lemma 9.14 bounds $\Pr\left[X_{t+1} \geq x + \zeta\sqrt{\mu - x} \,\middle|\, x\right]$, which only changes everything by a constant factor, since $\sqrt{\mu - x} \in \Omega\left(\sqrt{x}\right)$ holds due to $\sqrt{x}/\sqrt{\mu - x} \leq \sqrt{(5\mu/6)/(\mu/6)} \in \mathrm{O}(1)$. This concludes the proof. ■

Establishing the Lyapunov Condition

In order to apply Lemma 9.12, we are left with bounding the third central moments.

▶ **Lemma 9.16.** If $\mu \in \omega(1) \cap \mathrm{O}\left(\sqrt{n}\log(n)\right)$, then, for all $t \in \mathbf{N}$, it holds that

$$\mathrm{E}\left[|\Delta_t - \mathrm{E}[\Delta_t \mid X_t]|^3 \,\middle|\, X_t\right] \in \mathrm{O}\left(\mu^{\frac{3}{2}}\right)\,. \quad ◀$$

Proof. Consider an iteration $t \in \mathbf{N}$. We bound $\mathrm{E}\left[|\Delta_t - \mathrm{E}[\Delta_t \mid X_t]|^3 \,\middle|\, X_t\right]$ by

$$\mathrm{E}\left[\left(|\Delta_t| + |\mathrm{E}[\Delta_t \mid X_t]|\right)^3 \,\middle|\, X_t\right]\,,$$

aiming at reusing the bounds on $\mathrm{E}[\Delta_t \mid X_t]$ we know from Lemma 9.13.

In order to treat the binomial expression raised to the third power, we use the simple bound, for $a, b \geq 0$, that

$$\begin{aligned}(a+b)^3 &= a^3 + 3ab^2 + 3a^2b + b^3 \\ &\leq 4a^3 + 4b^3 \ .\end{aligned}$$

Thus,

$$\mathrm{E}\big[|\varDelta_t - \mathrm{E}[\varDelta_t \mid X_t]|^3 \bigm| X_t\big] \leq 4\mathrm{E}\big[|\varDelta_t|^3 \bigm| X_t\big] + 4|\mathrm{E}[\varDelta_t \mid X_t]|^3 \ ,$$

and due to Lemma 9.13, there is a constant $\zeta > 0$ such that, for all $X_t \in [\mu - 1]$ and $\mu \in \mathrm{O}\big(\sqrt{n}\log(n)\big)$, we have

$$-\zeta\sqrt{\mu} \leq \mathrm{E}[\varDelta_t \mid X_t] \leq \zeta\sqrt{\mu} \ .$$

Hence, our main task left is to bound $\mathrm{E}\big[|\varDelta_t|^3 \bigm| X_t\big]$. We claim that

$$\mathrm{E}\big[|\varDelta_t|^3 \bigm| X_t\big] \in \mathrm{O}\Big(\mu^{\frac{3}{2}}\Big) \ .$$

In order to show this, we analyze the distribution of $g(X_{t+1}) - g(X_t)$ conditional on X_t. We recall from Lemma 9.5 that X_{t+1} (that is, the new value before applying the potential function) is given by the sum of two distributions, both of which are binomial or almost binomial; more precisely, $X_{t+1} = Z_{1,t+1} + Z(C^*)$, where $Z_{1,t+1}$ is the number of 1s sampled through 1st-class individuals in iteration $t+1$, C^* is the number of 2nd-class individuals, and $Z(C^*)$ is the number of 1s sampled by them. We note, using Lemmas 9.4 and 9.5, that $Z(C^*) \prec C^* \prec \mathrm{Bin}(\lambda, c_1/\sqrt{n}) + \widetilde{Z}$, for some constant $c_2 > 0$, and $\widetilde{Z}$ takes some value in $[\lambda]$ only with probability of at most $\mathrm{e}^{-c_2\mu}$, for some constant $c_2 > 0$. Moreover, $Z_{1,t+1} \sim \mathrm{Bin}(\mu - C^*, X_t/\mu)$.

In order to overestimate $|\varDelta_t| = |g(X_{t+1}) - g(X_t)|$, we observe that

$$\begin{aligned}|g(X_{t+1}) - g(X_t)| = {}&\big|g\big(Z_{1,t+1} + Z(C^*)\big) - g(X_t)\big| \cdot \mathbf{1}\{Z_{1,t+1} + Z(C^*) < X_t\} \\ &+ \big|g\big(Z_{1,t+1} + Z(C^*)\big) - g(X_t)\big| \cdot \mathbf{1}\{Z_{1,t+1} + Z(C^*) \geq X_t\} \ .\end{aligned}$$

Hence, in order to bound $|\varDelta_t|$, it is enough to take the maximum of the two values

- $\varPsi_1 := \Big|g\Big(\mathrm{Bin}\left(\mu, \frac{X_t}{\mu}\right)\Big) - g(X_t)\Big|$ and
- $\varPsi_2 := \Big|g\Big(\mathrm{Bin}\big(\mu, \frac{X_t}{\mu}\big) + \mathrm{Bin}\big(\lambda, \frac{c}{\sqrt{n}}\big) + \widetilde{Z}\Big) - g(X_t)\Big|$

and analyze it. The first expression covers the case that $Z_{1,t+1} + Z(C^*) < X_t$. Then, we transform C^* random variables whose success probability is greater than X_t/μ (since 2nd-class individuals are biased toward 1s) into variables with success probability exactly X_t/μ, which increases the probability of $Z_{1,t+1}+Z(C^*)$ being less than X_t. On the other hand, if $Z_{1,t+1} + Z(C^*) \geq X_t$, we get an even larger value by including C^* additional experiments.

Case Ψ_1. We claim that $\mathrm{E}\big[|\Psi_1|^3 \bigm| X_t\big] \in \mathrm{O}\big(\mu^{3/2}\big)$. In order to show this, we proceed similarly as in bounding the expected value of Δ_t in Lemma 9.13 and define intervals of length $\sqrt{x}$, where $x := X_t$. More precisely, for $k \in \mathbf{Z}$, we define $I_k := \{\lceil x - (k+1)\sqrt{x}\rceil, \dots, \lfloor x - k\sqrt{x}\rfloor\}$, that is, also negative indices are allowed, leading to intervals lying above x. By applying equation (9.3), we get

$$\begin{aligned}\mathrm{E}\big[|\Psi_1|^3 \bigm| x\big] &\leq \sum_{k=0}^{\infty} \sum_{i \in I_k \cup I_{-k}} \left(\frac{\sqrt{\mu}(|i-x|)}{x-(k+1)\sqrt{x}}\right)^3 \Pr[|\Psi_1| = |i| \mid x] \\ &\leq \sum_{k=0}^{\infty} \left(\frac{\sqrt{\mu}(k+1)\sqrt{x}}{x-(k+1)\sqrt{x}}\right)^3 \Pr\big[|\Psi_1| \geq k\sqrt{x} \bigm| x\big] ,\end{aligned}$$

Note that for $k \leq \sqrt{x}$, we have by Chernoff bounds (Theorem 2.19) that

$$\begin{aligned}\Pr[X_{t+1} \in I_k \mid x] \leq \Pr\big[X_{t+1} \leq x - k\sqrt{x} \bigm| x\big] &\leq \mathrm{e}^{-\frac{k^2}{3}} \text{ and} \\ \Pr[X_{t+1} \in I_{-k} \mid x] \leq \Pr\big[X_{t+1} \geq x + k\sqrt{x} \bigm| x\big] &\leq \mathrm{e}^{-\frac{k^2}{4}} .\end{aligned}$$

Moreover, $\Pr[X_{t+1} \leq x/2 \mid x] \leq \mathrm{e}^{-x/24}$. Using the standard form of Chernoff bounds [MU05, Theorem 4.4], we additionally bound, for $j \geq 1$, the probability $\Pr[X_{t+1} \geq (1+j/2)x \mid x] \leq \big(\mathrm{e}^{j/2}/(1+j/2)^{1+j/2}\big)^x \leq \mathrm{e}^{-jx/10}$.

Using these different estimates while distinguishing between $k \leq \sqrt{x}/2 - 1$ and $k \geq \sqrt{x}/2$, we get for $x \geq 1$ that there are constants $c_3, c_4 > 0$ such that

$$\begin{aligned}\mathrm{E}\big[|\Psi_1|^3 \bigm| x\big] \leq \sum_{k=0}^{\frac{\sqrt{x}}{2}-1} &\left(\frac{\sqrt{\mu}(k+1)\sqrt{x}}{\frac{x}{2}}\right)^3 2\mathrm{e}^{-\frac{k^2}{4}} + \big(g(0)-g(x)\big)^3 \Pr\Big[X_{t+1} \leq \frac{x}{2} \Bigm| x\Big] \\ &+ \sum_{j=1}^{\infty} \left(g(x) - g\left(x\left(1+\frac{j}{2}\right)\right)\right)^3 \Pr\Big[X_{t+1} \geq x + j\frac{x}{2} \Bigm| x\Big]\end{aligned}$$

$$\begin{aligned} &\leq c_3\mu^{\frac{3}{2}} + (x\sqrt{\mu})^3 \mathrm{e}^{-\frac{x}{24}} + \sum_{j=1}^{\infty} \left(j\frac{x}{2}\sqrt{\mu}\right)^3 \mathrm{e}^{-j\frac{x}{10}} \\ &\leq c_4\mu^{\frac{3}{2}} \,, \end{aligned}$$

where we use the trivial bound $g(x) - g(y) \leq \sqrt{\mu}|x - y|$ and pessimistically assume $X_{t+1} = 0$ in the case $X_{t+1} \leq x/2$.

Case Ψ_2. With respect to Ψ_2, we observe that, for $c_5 > 0$ being a constant,

$$\Psi_2 < \left|g\left(\mathrm{Bin}\left(\mu, \frac{x}{\mu}\right)\right) - g(x)\right| + c_5\mu\Pr\left[\widetilde{Z} \neq 0 \,\middle|\, x\right] + \left(g(0) - g\left(\mathrm{Bin}\left(\lambda, \frac{c_1}{\sqrt{n}}\right)\right)\right)$$

by using $g(x+a+b)-g(x) = \big(g(x+a)-g(x)\big) + \big(g(x+a+b)-g(x+a)\big)$, for arbitrary $a, b \in \mathbf{R}$, and pessimistically estimating the contribution of $Z(C^*)$ to occur at point 0, where the potential function is steepest. Moreover, we pessimistically assume that the event $\widetilde{Z} \neq 0$ leads to the maximum possible change in potential, which is $g(0) - g(\mu) \in \mathrm{O}(\mu)$. Hence,

$$\begin{aligned} \mathrm{E}\left[|\Psi_2|^3 \,\middle|\, x\right] \leq 4\mathrm{E}&\left[\left|g\left(\mathrm{Bin}\left(\mu, \frac{x}{\mu}\right)\right) - g(x)\right|^3 \,\middle|\, x\right] \\ &+ 4\mathrm{E}\left[\left(g(0) - g\left(\mathrm{Bin}\left(\lambda, \frac{c_1}{\sqrt{n}}\right)\right)\right)^3 \,\middle|\, x\right] + (c_5\mu)^3 \cdot \Pr\left[\widetilde{Z} \neq 0 \,\middle|\, x\right] . \end{aligned} \tag{9.9}$$

We recall that $\Pr\left[\widetilde{Z} \neq 0 \,\middle|\, x\right] \leq \mathrm{e}^{-c_2\mu}$, thus

$$\begin{aligned} (c_5\mu)^3 \cdot \Pr\left[\widetilde{Z} \neq 0 \,\middle|\, x\right] &= (c_5\mu)^3 \cdot \mathrm{e}^{-c_2\mu} \\ &\in \mathrm{o}(1) \\ &\subseteq \mathrm{O}\left(\mu^{3/2}\right) \end{aligned}$$

for $\mu = \omega(1)$. Hence, the last term from inequality (9.9) has already been bounded as desired, and we only have to show bounds on the first two terms.

We recognize that the first term of inequality (9.9) is $\mathrm{O}\big(\mu^{3/2}\big)$ since, up to constant factors, it is the same as $\mathrm{E}\big[|\Psi_1|^3 \mid X_t\big]$. Hence, we are left with the claim

$$\mathrm{E}\left[\left(g(0) - g\left(\mathrm{Bin}\left(\lambda, \frac{c_1}{\sqrt{n}}\right)\right)\right)^3 \,\middle|\, x\right] \in \mathrm{O}\left(\mu^{\frac{3}{2}}\right) .$$

In order to show this, we let $Z \sim \mathrm{Bin}(\lambda, c_1/\sqrt{n})$ and consider different definitions of the intervals I_k, for $k \geq 0$, that Z can fall into. The definition of the intervals distinguishes two cases.

Case 1: $\lambda \geq \sqrt{n}/(2\mathrm{e}^2 c_1)$. As the greatest difference of two neighboring values of $-g$ is at most $\sqrt{\mu}$, it suffices to prove the stronger claim

$$\sqrt{\mu} \cdot \mathrm{E}\left[\mathrm{Bin}\left(\lambda, \frac{c_1}{\sqrt{n}}\right)^3 \,\middle|\, x\right] \in \mathrm{O}\left(\mu^{\frac{3}{2}}\right) .$$

We define $I_0 := [0, 2\mathrm{e}^2 c_1\lambda/\sqrt{n}]$ and $I_k := [(1+k)\mathrm{e}^2 c_1 \lambda/\sqrt{n}, (2+k)\mathrm{e}^2 c_1\lambda/\sqrt{n}]$ for $k \geq 1$. Then (similar to the analysis of $\mathrm{E}[|\Psi_1|^3 \mid x]$), we get

$$\mathrm{E}\left[\mathrm{Bin}\left(\lambda, \frac{c_1}{\sqrt{n}}\right)^3 \,\middle|\, x\right] \leq \left(\frac{2\mathrm{e}^2 c_1 \lambda}{\sqrt{n}}\right)^3 + \sum_{k=1}^{\infty} \left(\frac{(2+k)\mathrm{e}^2 c_1\lambda}{\sqrt{n}}\right)^3 \Pr[Z \in I_k \mid x] .$$

Using the Chernoff bound [MU05, Theorem 4.4] that $\Pr[Y \geq s] \leq 2^{-s}$ for $s \geq 2\mathrm{e}^2\mathrm{E}[Y]$, we get $\Pr[Z \in I_k] \leq \mathrm{e}^{-(2+k)\mathrm{e}^2\lambda/\sqrt{n}} \leq \mathrm{e}^{-c_6 k/2}$ by our assumption on λ, where $c_6 > 0$ is a constant. We see that

$$\begin{aligned}\mathrm{E}\left[\mathrm{Bin}\left(\lambda, \frac{c_1}{\sqrt{n}}\right)^3 \,\middle|\, x\right] &\leq \frac{c_6\lambda^3}{n^{\frac{3}{2}}} + \frac{c_6\lambda^3}{n^{\frac{3}{2}}} \sum_{k=1}^{\infty} (2+k)^3 \mathrm{e}^{-c_6 \frac{k}{2}} \\ &\in \mathrm{O}\left(\frac{\lambda^3}{n^{\frac{3}{2}}}\right) \\ &= \mathrm{O}\left(\frac{\mu^3}{n^{\frac{3}{2}}}\right) ,\end{aligned}$$

hence $\sqrt{\mu} \cdot \mathrm{E}\big[\mathrm{Bin}(\lambda, c_1/\sqrt{n})^3 \mid x\big] \in \mathrm{O}\big(\mu^{7/2}/n^{3/2}\big)$. Since $\mu \in \mathrm{O}\big(\sqrt{n}\log(n)\big)$ by assumption of the lemma, the bound is at most $\mathrm{O}\big(n^{1/4}\log^{7/2}(n)\big)$, and this is clearly $\mathrm{O}\big(\mu^{3/2}\big)$, since $\mu \in \Omega\big(\sqrt{n}\big)$ in this case.

Case 2: $\lambda < \sqrt{n}/(2\mathrm{e}^2 c_1)$. We define $I_k := [k, k+1]$ for $k \geq 0$ and note that $\mathrm{E}[Z] \in \mathrm{O}(1)$ since $\mu \in \mathrm{O}(\lambda) = \mathrm{O}\big(\sqrt{n}\big)$. Hence, by Chernoff bounds for $k > \mathrm{E}[Z]$,

$\Pr[Z \geq k \mid x] = \mathrm{e}^{-\alpha k}$ for some constant $\alpha > 0$. Similar to the other case, we get

$$\begin{aligned} \mathrm{E}\Big[\big(g(0) - g(Z)\big)^3 \,\Big|\, x\Big] &\leq (\sqrt{\mu})^3 \cdot \mathrm{E}[Z]^3 + \sqrt{\mu} \sum_{k > \mathrm{E}[Z]}^{\infty} k^3 2^{-\alpha k} \\ &\in \mathrm{O}\Big(\mu^{\frac{3}{2}}\Big) , \end{aligned}$$

which completes the proof. ■

Using Lemmas 9.15 and 9.16, we now establish the Lyapunov condition (equation (9.2)), assuming, for all $t \in \mathbf{N}$, that $X_t \leq (5/6)\mu$. Using Lemma 9.12, we get for $s_t^2 := \sum_{j=0}^{t-1} \mathrm{Var}\big[\Delta_j \,\big|\, X_j\big]$ that

$$\begin{aligned} \frac{1}{s_t^3} \sum_{j=0}^{t-1} \mathrm{E}\big[|\Delta_j - \mathrm{E}[\Delta_j \mid X_j]|^3 \,\big|\, X_j\big] &\in \mathrm{O}\left(\frac{\mu^{\frac{3}{2}} t}{(\mu t)^{\frac{3}{2}}}\right) \\ &= \mathrm{O}\left(\frac{1}{\sqrt{t}}\right) , \end{aligned}$$

which is $\mathrm{o}(1)$ for $t \in \omega(1)$. The sum of the Δ_j can then be approximated as stated in the following lemma.

▶ **Lemma 9.17.** Let $Y_t := \sum_{j=0}^{t-1} \Delta_j$ and $t \in \omega(1)$. Then

$$\frac{Y_t - \mathrm{E}[Y_t \mid X_0]}{\sqrt{\sum_{j=0}^{t-1} \mathrm{Var}\big[\Delta_j \,\big|\, X_j\big]}}$$

converges in distribution to $\mathrm{N}(0, 1)$. The absolute error of this approximation is $\mathrm{O}(1/\sqrt{t})$. ◀

Likelihood of a Frequency Getting Very Small

We now apply Lemma 9.17 in order to prove how likely it is for a single frequency to either get close to $1/n$ or exceed $5/6$. For this, we use the following estimates for $\Phi(x)$.

▶ **Lemma 9.18 ([Fel68, Chapter VII, Lemma 2]).** For any $x > 0$, it holds that

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

and for $x < 0$, it holds that

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

▶ **Lemma 9.19.** Consider a bit of the UMDA on OneMax and let p_t be its frequency in iteration $t \in \mathbf{N}$. We say that the process *breaks a border at time t* if $\min\{p_t, 1 - p_t\} \le 1/n$. Given $s < 0$ and any starting state $p_0 \le 5/6$, let T_s be the smallest t such that $p_t - p_0 \le s$ holds or a border is broken.

Assume that $\Theta(n)$ other frequencies stay within $[1/6, 5/6]$ until time T_s. Choosing $0 < \alpha < 1$, where $1/\alpha \in o(\mu)$ and $\alpha \in O\big(\sqrt{n}/\mu\big)$, and $-1 < s < 0$ constant, we then have for some constant $\kappa, \zeta > 0$ that

$$\begin{aligned}\Pr\big[T_s \le \alpha s^2\mu \text{ or } p_t \text{ exceeds } \tfrac{5}{6} \text{ before } T_s\big]&\\ \ge \left(\frac{(|s|\alpha)^{\frac{1}{2}}}{\kappa} - \frac{(|s|\alpha)^{\frac{3}{2}}}{\kappa^3}\right)\frac{1}{\sqrt{2\pi}}e^{-\frac{\kappa^2}{2|s|\alpha}} - \frac{\zeta}{\sqrt{\alpha\mu}}&\ .\end{aligned}$$ ◀

Proof. Throughout the analysis, we assume that $X_t \le (5/6)\mu$, since all considerations are stopped when the frequency exceeds $5/6$, that is, when $X_t \ge (5/6)\mu$. By Lemma 9.13, we have, for all $j \in \mathbf{N}$ and $X_j \in [\mu - 1]$, that there are two constants $c_1, \gamma_1 > 0$ such that

$$\mathrm{E}\big[\Delta_j \mid X_j\big] \ge -\sqrt{\frac{\mu}{X_j + 1}}\left(e^{-c_1\mu} + \gamma_1\left(\frac{X_j}{\sqrt{n}} + \frac{X_j}{\mu}\right)\right) .$$

Moreover, according to Lemma 9.15, $\mathrm{Var}\big[\Delta_j \mid X_j\big] \ge \sqrt{c_2}\mu$ for some constant $c_2 > 0$. Since the Lyapunov condition has been established for $Y_t := \sum_{j=0}^{t-1}\Delta_j$ in Lemma 9.17, we know that $\big(Y_t - \mathrm{E}[Y_t \mid X_0]\big)/s_t$ converges in distribution to $\mathrm{N}(0, 1)$ if $t \in \omega(1)$. Hence, we choose $t = \alpha s^2\mu$, which is $\omega(1)$ since $\alpha \in \omega(1/\mu)$ by assumption.

For $s_t^2 := \sum_{j=0}^{t-1}\mathrm{Var}\big[\Delta_j \mid X_j\big]$, we obtain $s_t^2 \ge \alpha s^2 c_2\mu^2$. Hence, recalling that we assume $s < 0$, we get $s_t \ge \sqrt{\alpha c_2}|s|\mu$. The next task is to bound $\mathrm{E}[Y_t]$. Using our bound on $\mathrm{E}\big[\Delta_j \,\big|\, X_j\big]$ and recalling that $X_t \in [0..(5/6)\mu]$ and $\mu \in \omega(1)$, for some constants $\gamma_2, c_3 > 0$, we have

$$\begin{aligned}\mathrm{E}[\Delta_t \mid X_t] &\geq -\left(\mathrm{e}^{-c_1\mu}\sqrt{\frac{\mu}{1}} + \gamma_1 \frac{\frac{5}{6}\mu}{\sqrt{\frac{5}{6}\mu + 1}}\left(\frac{\sqrt{\mu}}{\sqrt{n}} + \frac{1}{\sqrt{\mu}}\right)\right)\\ &\geq -\left(c_3 + \gamma_2\frac{\mu}{\sqrt{n}}\right).\end{aligned}$$

This implies $\mathrm{E}[Y_t] \geq -t\big(c_3 + \gamma_2\mu/\sqrt{n}\big) = -\alpha s^2\mu\big(c_3 + \gamma_2\mu/\sqrt{n}\big)$. Therefore, for some constant $\gamma_3 > 0$ depending on α, we get

$$\begin{aligned}\frac{\mathrm{E}[Y_t]}{s_t} &\geq -\frac{(\alpha s^2\mu)\big(c_3 + \gamma_2\frac{\mu}{\sqrt{n}}\big)}{\sqrt{\alpha c_2}|s|\mu}\\ &\geq -\gamma_3\sqrt{\frac{1}{c_2\alpha}},\end{aligned}$$

using the assumptions $|s| \leq 1$ along with both $\alpha \leq 1$ and $\alpha \in \mathrm{O}\big(\sqrt{n}/\mu\big)$.

To bound $\Pr[Y_t \geq r]$ for arbitrary r, we note that

$$Y_t \geq r \iff \frac{Y_t}{s_t} - \frac{\mathrm{E}[Y_t \mid X_0]}{s_t} \geq \frac{r}{s_t} - \frac{\mathrm{E}[Y_t \mid X_0]}{s_t},$$

and recall that the distribution of $Y_t/s_t - \mathrm{E}[Y_t \mid X_0]/s_t$ converges to $\mathrm{N}(0,1)$ with absolute error $\mathrm{O}\big(1/\sqrt{t}\big)$. Hence, there is a constant $c_4 > 0$ such that

$$\Pr[Y_t \geq r] \geq 1 - \Phi\left(\frac{r}{\sqrt{c_2\alpha}|s|\mu} + \gamma_3\sqrt{\frac{1}{c_2\alpha}}\right) - \frac{c_4}{\sqrt{t}} \tag{9.10}$$

for any r such that the argument of Φ is positive, where Φ denotes the cumulative distribution function of the standard normal distribution.

We focus on the event $E^* \coloneqq \{Y_t \geq 2\mu\sqrt{|s|}\}$, recalling that $s < 0$ and $X_t \geq X_0 \Leftrightarrow Y_t \leq Y_0$. Note that E^* means $g(X_t) - g(X_0) \geq 2\mu\sqrt{|s|}$, and this implies an upper bound on the negative $X_t - X_0$ as follows: function g is steepest at point 0, and by the definition of g and an approximation by an integral [Cor+09, Inequality (A.12)], for any $y \geq 1$, we have

$$\begin{aligned} g(y) - g(0) &\leq \sum_{j=0}^{y-1} \sqrt{\frac{\mu}{j+1}} \\ &\leq \sqrt{\mu}\left(1 + \int_1^y \frac{1}{\sqrt{j}}\,\mathrm{d}j\right) \\ &= \sqrt{\mu}(1 + 2\sqrt{y} - 2\sqrt{1}) \\ &\leq 2\sqrt{y\mu}\ . \end{aligned}$$

Thus, for an $a > 0$, the event $\{g(X_t) - g(X_0) \geq a\}$ is only possible if $X_t \leq X_0 - a^2/(4\mu)$. In other words, the event E^* implies $X_t - X_0 \leq s\mu$, which is equivalent to $p_t - p_0 \leq s$. Hence, in order to complete the proof, we only need a lower bound on the probability of E^*. Setting $r := 2\mu\sqrt{|s|}$ in inequality (9.10), we bound the argument of Φ according to

$$\begin{aligned} \frac{r}{\sqrt{c_2\alpha|s|\mu}} + \frac{\gamma_3}{\sqrt{c_2\alpha}} &\leq \frac{2}{\sqrt{c_2|s|\alpha}} + \frac{\gamma_3}{\sqrt{c_2\alpha}} \\ &\leq \frac{\gamma_4}{\sqrt{c_2|s|\alpha}}\ , \end{aligned}$$

for some constant $\gamma_4 > 0$, since $|s| \leq 1$.

By Lemma 9.18,

$$\begin{aligned} 1 - \Phi\left(\frac{\gamma_4}{\sqrt{c_2|s|\alpha}}\right) &\geq \left(\frac{\sqrt{c_2|s|\alpha}}{\gamma_4} - \frac{\left(\sqrt{c_2|s|\alpha}\right)^3}{\gamma_4^3}\right)\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{\gamma_4^2}{2c_2|s|\alpha}\right) \\ &=: p(\alpha, s)\ , \end{aligned}$$

which means that the frequency changes by s (which is negative) until iteration $\alpha s^2\mu$ with probability at least $p(\alpha, s) - c_4/\sqrt{t} = p(\alpha, s) - c_4/\sqrt{\alpha\mu}$, where the last term stems from the bound on the absolute error of the approximation by the standard normal distribution. Choosing $\kappa := \gamma_4/\sqrt{c_2}$ in the statement of the lemma completes the proof. ■

9.3.4 Proof of the Lower Bound

Finally, we put all previous lemmas together in order to prove our main theorem: Theorem 9.6.

Proof of Theorem 9.6. As outlined above, we distinguish between three regimes for λ. The case of small λ (that is, $\lambda < (1 - c_1)\log_2(n)$) is covered by Theorem 9.8, noting that $\Omega\big(n\log(n)\big)$ dominates the lower bound for the considered range of μ. The case of large λ (that is, $\lambda \in \Omega\big(\sqrt{n}\log(n)\big)$) is covered by Corollary 9.11. We are left with the medium case ($\mu \in \Omega\big(\log(n)\big) \cap \mathrm{o}\big(\sqrt{n}\log(n)\big)$), which is the most challenging one to prove.

In the following, for some $s \in \mathbf{R}$ which we specify later, we consider a phase consisting of $T := s^2\gamma \cdot \min\{\mu, \sqrt{n}\}$ iterations, for the constant $\gamma > 0$ from Lemma 9.10. Without loss of generality, we assume that $\gamma < 1$. We conceptually split individuals (that is, bit strings) of the UMDA into two substrings of length $n/2$ each and apply Lemma 9.10 with respect to the first half of the bits. In the following, we condition on the event that $\Theta(n)$ frequencies from the first half are within the interval $[1/6, 5/6]$ throughout the phase.

We show next that some frequencies from the second half are likely to walk down to the lower border. Let $j \in [n]$ be an arbitrary position from the second half. First, we apply Lemma 9.9. Hence, p_j does not exceed $5/6$ within the phase with probability $\Omega(1)$. In the following, we condition on this event.

We then revisit bit j and apply Lemma 9.19 in order to show that, under this condition, the random walk on its frequency p_j achieves a negative displacement. Note that the event of not exceeding a certain positive displacement (more precisely, the displacement of $5/6 - 1/2 = 1/3$) is positively correlated with the event of reaching a given negative displacement (formally, the state of the conditional stochastic process is always stochastically smaller than of the unconditional process). We can therefore apply Lemma 9.19 for a negative displacement of $s := -5/6$ within T iterations. Note that the condition of the lemma that demands $\Theta(n)$ frequencies to be within $[1/6, 5/6]$ is satisfied by our assumption concerning the first half of the bits. Choosing $\alpha = T/(s^2\mu)$, we get $1/\alpha \in \mathrm{o}\big(\log(n)\big)$ (since $\mu \in \mathrm{o}\big(\sqrt{n}\log(n)\big)$ and $T \in \Theta\big(\min\{\mu, \sqrt{n}\}\big)$), whereby we easily satisfy the assumption $1/\alpha \in \mathrm{o}(\mu)$. As $T \in \mathrm{O}\big(\sqrt{n}\big)$ and s is constant, we also satisfy the assumption $\alpha \in \mathrm{O}\big(\sqrt{n}/\mu\big)$. Moreover, it holds that $\alpha \le \gamma < 1$ by definition. Now, Lemma 9.19 states that the probability of p_j reaching a total displacement of $-5/6$ (or hitting the lower border before) within the phase of length T is, for a constant $\zeta > 0$, at least

$$\left(\frac{(|s|\alpha)^{\frac{1}{2}}}{\kappa} - \frac{(|s|\alpha)^{\frac{3}{2}}}{\kappa^3}\right)\frac{1}{\sqrt{2\pi}}\mathrm{e}^{-\frac{\kappa^2}{2\cdot|s|\alpha}} - \frac{\zeta}{\sqrt{\alpha\mu}} \; . \tag{9.11}$$

In order to bound the last expression from below, we distinguish between two cases for μ.

If $\mu \leq \sqrt{n}$, then $\alpha \in \Omega(1)$ and term 9.11 is at least $\Omega(1)$, since $T \in \Omega(\mu) \subseteq \Omega\big(\log(n)\big) \subseteq \omega(1)$.

If $\mu \geq \sqrt{n}$, then we have $T \in \Omega\big(\sqrt{n}\big)$. Since $1/\alpha \in \mathrm{o}\big(\log(n)\big)$, we estimate term 9.11, for some $\varepsilon \in \mathrm{o}(1)$, from below by that there is a constant $c_1 > 0$ and an $\eta \in \mathrm{o}(1)$ such that

$$\frac{1}{\sqrt{\varepsilon \ln(n)}} \cdot \mathrm{e}^{-\varepsilon \ln(n)} - c_1 \frac{\ln(n)}{n^{\frac{1}{4}}} \geq n^{-\eta} \; .$$

Combining this with the probability of not exceeding $5/6$, the probability of p_j hitting the lower border within T iterations is, in any case, $\Omega(n^{-\eta})$. Note that this argumentation applies to every of the last $n/2$ bits, and, as explained in Section 9.2.2, the bounds derived hold independently for all these bits. Hence, by Chernoff bounds (Theorem 2.19), the number of frequencies from the second half that hit the lower border within T iterations is $\Omega\big(n^{1-\eta}\big)$ with a probability of at least $1 - 2^{-c_2 n^{1-\eta}}$, for some constant $c_2 > 0$.

A frequency that has hit the lower border $1/n$ somewhere in the phase may recover (that is, reach a larger value) by the end of the phase. However, for each bit, the probability of not recovering is, for some $\eta' = \mathrm{o}(1)$, at least, by using Theorem 2.22,

$$\begin{aligned}\left(1 - \frac{1}{n}\right)^{T\lambda} &\geq \mathrm{e}^{-\eta' \ln(n)} \\ &= n^{-\eta'} \; ,\end{aligned}$$

since we consider $T \in \mathrm{O}\big(\sqrt{n}\big)$ iterations and $\lambda \in \mathrm{o}\big(\sqrt{n}\log(n)\big)$ samples per iteration. Again applying Chernoff bounds leaves $\Omega(n^{1-\eta-\eta'})$ bits at the lower border at iteration T with a probability of at least $1-2^{-c_3 n^{1-\eta-\eta'}}$, for some constant $c_3 > 0$. Making use of Lemma 9.7 yields the desired run time bound. ■

9.4 Relaxing the Condition on the Population Size

Theorem 9.6 assumed that $\lambda = (1 + \beta)\mu$ for some constant $\beta > 0$. We think that the lower bound of $\Omega\big(n \log(n)\big)$ holds for all combinations of μ and λ. As a step toward a proof of this conjecture, we extend our lower bound toward all $\mu \leq c \log(n)$ for a sufficiently small constant $c > 0$. This includes the extreme case of $\mu = 1$, for which no matching upper bound has been proven up to date.

▶ **Theorem 9.20.** Let $c_1 > 0$ be a sufficiently small constant, let $c_2 \in \mathrm{O}(1)$, let $\mu \leq c_1 \ln(n)$, and let $\lambda = n^{c_2}$. Then the optimization time of the UMDA on OneMax is $\Omega\big(\lambda + n \log(n)\big)$ with high probability and in expectation. ◀

Proof. The lower bound λ follows since the UMDA will sample the optimum in the first iteration only with a probability of $2^{-\gamma_1 n}$, for some constant $\gamma_1 > 0$. Thus, with high probability, all λ offspring from the first generation need to be evaluated. In the following, we assume $\lambda \in \mathrm{O}\big(n \log(n)\big)$ since otherwise nothing is left to show.

We now follow the ideas underlying the proof of Theorem 9.8 by showing that the best μ individuals from the initial generation are still close to uniform, resulting in many frequencies being set to their minimum $1/n$. Note that the mentioned theorem considered all λ individuals from the initial generation, which are uniform on the search space. Here we focus on the best μ from the initial population, which violates the independence.

By Chernoff bounds (Theorem 2.19), the probability that at least one of the λ initial individuals has $3n/4$ or more 1s is at most $\lambda \mathrm{e}^{-\gamma_2 n} = \mathrm{e}^{-\gamma_3 n}$, for some constants $\gamma_2, \gamma_3 > 0$. In the following, we condition on this not happening.

We consider an arbitrary individual of the λ initial individuals. Clearly, given that it has k 1s, the actual distribution of 1s is uniform over all permutations of k 1s. This still applies to the selected best μ individuals since OneMax is unbiased with respect to permutations, that is, it only depends on the number of 1s. Hence, we get the following property $(*)$: if we consider an arbitrary individual from the μ best, then every bit in it takes the value 1 with the same probability p^* (not necessarily independently of the other bits). Since the expected number of 1s is bounded by $3n/4$, we have that $p^* \leq 3/4$, otherwise, the expected value would be larger, which we excluded.

Pessimistically assuming that all λ individuals have $3n/4$ 1s, we obtain $p^* = 3/4$ and have established the property $(*)$ independently for all individuals (also when arguing only about the best μ ones) but still not independently for all bits.

We now consider an arbitrary bit position $i \in [n]$ from one of the best μ individuals. If bit i takes the value 0, then the $3n/4$ 1s have to be taken at positions other than i and are uniformly distributed among these positions. Hence, any bit $j \neq i$ takes the value 1 with probability at most $(3n/4)/(n-1)$ and 0 with probability at least $1-(3n/4)/(n-1) = (n/4-1)/(n-1)$. Altogether, independently of the outcome of i, bit j takes the value 0 with probability at least $\min\{1/4, (n/4-1)/(n-1)\} = (n/4-1)/(n-1)$. We iterate this argument over an arbitrary set S^* consisting of at most $n/8$ bits (for example, the first $n/8$ positions). Hence, every of these bits takes the value 0 with probability at least

$$\frac{\frac{n}{4}-\frac{n}{8}}{n-\frac{n}{8}} = \frac{1}{7},$$

independently of the other bits in S^*. As this applies independently to all μ best individuals, each bit in S^* is set to 0 in all μ best individuals with a probability of at least $(1/7)^\mu$, independently of the other bits in S^*.

Due to the independence achieved by the estimations, we can now apply Chernoff bounds (Theorem 2.19) with respect to to the sum of the indicator random variables associated with the events ›bit i is set to 0 in all μ best individuals‹ over all $i \in S^*$. The expected number of such bits is at least $\ell := (n/8)(1/7)^\mu$. If we choose $\mu \leq c_1 \ln(n)$ for a sufficiently small constant $c_1 > 0$, we obtain, for a constant $c' > 0$, that $\ell \geq n^{c'}/8$. Moreover, the probability that fewer than $n^{c'}/9$ bits take the value 0 in all μ best individuals is at most $2^{-\gamma_4 n}$ then, where $\gamma_4 > 0$ is a constant. We assume this to happen and note that the failure probability altogether is at most $2^{-\gamma_5 n}$, for a constant $\gamma_5 > 0$. Now Lemma 9.7 yields the theorem. ■

9.5 Conclusions

We analyzed the UMDA on OneMax and obtained the general bound $\Omega\big(\lambda + \mu\sqrt{n} + n\log(n)\big)$ on its expected run time for combinations of μ and λ where $\lambda \in \Theta(\mu)$ or $\mu \leq c\log(n)$ (for a sufficiently small constant c). This lower-bound analysis is the first of its kind and contributes advanced techniques, including potential functions.

We note that our lower bound for the UMDA is tight in many cases, as has been shown recently [LN17; Wit17]. We also note that although our main result assumes $\lambda \in \Theta(\mu)$, we do not think that larger values of λ can be beneficial. If, for $\alpha \in \omega(1)$, we have $\lambda = \alpha\mu$, the progress due to 2nd-class individuals can be by a factor of at most α bigger. However, also the computational effort per generation would grow by this factor. Still, we have not presented a formal proof for all such cases.

Further run time analyses of the UMDA or other EDAs for other classes of functions are an obvious subject for future research. In this respect, we hope that our technical contributions are useful and can be extended toward a more general lower-bound technique at some point. For such considerations, it is important to note that the behavior per *iteration* is quite different for the UMDA and the cGA/2-MMAS$_{\text{IB}}$ – the common lower bound of $\Omega\big(n\log(n)\big)$ is a result on the expected number of *fitness function evaluations*. Consequently, a general lower-bound technique has to be able to express these differences and most likely focus heavily on how an algorithm works in each single iteration.

10 Upper Bounds of the sig-cGA on LeadingOnes and OneMax

This chapter is based on joint work with Benjamin Doerr [DK18b]. Some parts stem from an unpublished extension [DK18a]. Theorem 10.9 has been slightly generalized and the proof has been adjusted in order to consider the same parameter setting as in Corollary 6.16.

In this chapter, we introduce a new EDA (Algorithm 2) and prove that it optimizes OneMax, LeadingOnes, and BinVal each in $\mathrm{O}(n \log(n))$ function evaluations in expectation and with high probability (Theorems 10.5 and 10.8 and Corollary 10.6, respectively). This result is so far unmatched by any other EDA or EA. Further, we prove that the scGA from Section 6.4.1, which optimizes LeadingOnes in $\mathrm{O}(n \log(n))$ too, has an exponential run time on OneMax in its parameter K (Theorem 10.9).

10.1 Introduction

As we discussed in Chapter 6, it is a nontrivial task to determine how to choose the step size of an EDA. On the one hand, if it is too large, then the frequencies may accidentally reach a wrong border. On the other hand, if it is too small, it may take the algorithm too long to update a frequency to the correct border. Our results in Chapter 9 as well as the results by Sudholt and Witt [SW16a] underline this point by proving lower bounds for various regimes of the step size. Further, a recent result by Lengler et al. [LSW18] suggests a bimodal behavior of the expected run time of the cGA on OneMax for medium step sizes.

All of these results draw the following picture: for a balanced EDA, there exists some inherent noise in the update. Thus, if the parameter responsible for the update of the probabilistic model is large and the speed of convergence high, the algorithm only uses a few samples before it converges. During this time, the noise introduced by the balance-property may not be overcome, resulting in the probabilistic model converging to an incorrect one, as the algorithms are not stable (Theorem 6.11). Hence, the parameter has to be chosen sufficiently small in order to guarantee convergence to the correct model, resulting in a slower optimization time.

As we shall argue in this chapter, the reason for this dilemma is that EDAs only use information from a single iteration when performing an update. Thus, the decision of whether and how a frequency should be changed has to be made on the spot, which may result in harmful decisions.

In order to overcome these difficulties, we propose a conceptually new EDA that has some access to the search history and updates the model only if there is sufficient reason. The *significance-based compact genetic algorithm* (sig-cGA, see Algorithm 2) stores for each position the history of bits of good solutions so far. If it detects that either statistically significantly more 1s than 0s or vice versa were sampled, it changes the corresponding frequency, otherwise not. Thus, the sig-cGA only performs an update when it has proof that it makes sense. This sets it apart from the other EDAs analyzed so far.

We prove that the sig-cGA is able to optimize LeadingOnes and OneMax in $\mathrm{O}\big(n\log(n)\big)$ function evaluations in expectation and with high probability (Theorems 10.5 and 10.8, respectively), which has not been proven before for any other EDA or classical EA (for further details, see Table 10.1).

We also observe that the analysis for LeadingOnes can easily be modified to also show an $\mathrm{O}\big(n\log(n)\big)$ run time for the *binary value* function BinVal (Corollary 10.6), which is a linear function with exponentially growing coefficients. This result is interesting in that it indicates that the sig-cGA has asymptotically the same run time on BinVal and OneMax. In contrast, for the cGA, it is known that the run times on OneMax and BinVal differ significantly [Dro06].

We further show that the scGA, which we proposed in Section 6.4.1 and also optimizes LeadingOnes in $\mathrm{O}\big(n\log(n)\big)$, behaves poorly on OneMax (Theorem 10.9). Its run time is at least $2^{c\min\{n,K\}}$ in expectation and with high probability, where $c > 0$ is a constant and $1/K$ is the step size of the algorithm.

These results – the positive ones for the sig-cGA using a longer history of the search process and the negative ones for other algorithms not exploiting a longer history – suggest that a fruitful direction for the future development of the field of evolutionary computation (EC; not restricted to theory) is the search for algorithms that enrich the classic generational approaches with mechanisms that profit from regarding more than one generation. We discuss this in more detail in the conclusions of this paper. We note that, from the practical point of view, our algorithm not only shows a performance not seen so far with other

Table 10.1: Expected run times (number of fitness evaluations) of various algorithms until they first find an optimum for the two functions OneMax (equation (2.1)) and LeadingOnes (equation (2.2)). For optimal parameter settings, many algorithms have a run time of $\Theta(n\log(n))$ for OneMax and of $\Theta(n^2)$ for LeadingOnes. We note that the $(1+(\lambda,\lambda))$ GA has a $\mathrm{o}(n\log(n))$ run time on OneMax (and even linear run time with a dynamic parameter choice), but we do not see why it should have a performance better than quadratic on LeadingOnes.

Algorithm	OneMax	constraints	LeadingOnes	constraints
$(1+1)$ EA	$\Theta(n\log(n))$ [DJW02]	none	$\Theta(n^2)$ [DJW02]	none
$(\mu+1)$ EA	$\Theta(\mu n + n\log(n))$ [Wit06]	$\mu \in \mathrm{O}(\mathrm{poly}(n))$	$\Theta(\mu n\log(n) + n^2)$ [Wit06]	$\mu \in \mathrm{O}(\mathrm{poly}(n))$
$(1+\lambda)$ EA	$\Theta\left(n\log(n) + \frac{\lambda n\log(\log(\lambda))}{\log(\lambda)}\right)$ [DK15; JJW05]	$\lambda \in \mathrm{O}(n^{1-\varepsilon})$, $\varepsilon > 0$	$\Theta(n^2 + \lambda n)$ [JJW05]	$\lambda \in \mathrm{O}(\mathrm{poly}(n))$
$(1+(\lambda,\lambda))$ GA	$\Theta\left(\max\left\{\frac{n\log(n)}{\lambda}, \frac{n\lambda\log(\log(\lambda))}{\log(\lambda)}\right\}\right)$ [DD18]	$p = \frac{\lambda}{n}, c = \frac{1}{\lambda}$	unknown	–
CSA	$\Omega(n^c)$ [DK18a]	$c > 0$	$\mathrm{O}(n\log(n))$ [MS17]	$\mu \geq 8\ln((4n+6)n)$, restarts
UMDA/PBIL[22]	$\Omega(\lambda\sqrt{n} + n\log(n))$ [Theorem 9.6]	$\mu \in \Theta(\lambda)$	$\mathrm{O}(n\lambda\log(\lambda) + n^2)$ [DL15; LN18]	$\lambda \in \Omega(\log(n)), \mu \in \Theta(\lambda)$
	$\mathrm{O}(\lambda n)$ [LN17; Wit17]	$\mu \in \Omega(\log(n)) \cap \mathrm{O}(\sqrt{n}), \lambda \in \Omega(\mu)$ or $\mu \in \Omega(\sqrt{n}\log(n)), \mu \in \Theta(\lambda)$ or $\mu \in \Omega(\log(n)) \cap \mathrm{o}(n), \mu \in \Theta(\lambda)$		
cGA/2-MMAS$_{\mathrm{IB}}$	$\Omega\left(\frac{\sqrt{n}}{\rho} + n\log(n)\right)$ [SW16a]	$\frac{1}{\rho} \in \mathrm{O}(\mathrm{poly}(n))$	unknown	–
	$\mathrm{O}\left(\frac{\sqrt{n}}{\rho}\right)$ [SW16a]	$\frac{1}{\rho} \in \Omega(\sqrt{n}\log(n)) \cap \mathrm{O}(\mathrm{poly}(n))$		
1-ANT	$\Theta(n\log(n))$ [NW09]	$\rho \in \Theta(1)$	$\mathrm{O}(n^2 \cdot (6\mathrm{e})^{1/(n\rho)})$ [Doe+11b]	none
			$2^{c\min\{n, 1/(n\rho)\}}$ [Doe+11b]	$c \in \Omega(1)$
MMAS*	$\mathrm{O}\left(\frac{n\log(n)}{\rho}\right)$ [NSW09]	$\rho \in \mathrm{O}(1)$	$\mathrm{O}\left(n^2 + \frac{n\log(n)}{\rho}\right)$ [NSW09]	$\rho \in \mathrm{O}(1)$
			$\Omega\left(n^2 + \frac{n}{-\rho\log(2\rho)}\right)$ [NSW09]	$\rho = \frac{1}{\mathrm{poly}(n)}$
scGA (Section 6.4.1)	$2^{c\min\{n, K\}}$ [Theorem 10.9]	$K \geq \beta\log(n), \beta \in \Theta(1)$, $\sigma \in \Theta\left(\frac{1}{K}\right), d \in \Theta(1), c \in \Omega(1)$	$\mathrm{O}(n\log(n))$ [Corollary 6.16]	$K \geq \beta\log(n), \beta \in \Theta(1)$, $\sigma \in \Theta\left(\frac{1}{K}\right), d \in \Theta(1)$
sig-cGA (Algorithm 2)	$\mathrm{O}(n\log(n))$ [Theorem 10.8]	$\varepsilon > 12$	$\mathrm{O}(n\log(n))$ [Theorem 10.5]	$\varepsilon > 12$

[22] The results shown for the PBIL are the results of the UMDA, since the latter is a special case of the former. Wu et al. [WKM17] also analyze the PBIL but with worse results. The only exception to this is the result by Lehre and Nguyen [LN18], which is specifically for the PBIL.

algorithms, it also is easier to use, since, unlike with most other EDAs, the delicate choice of the step size is obsolete.

10.2 Preliminaries

In our analysis, we regard the two classic benchmark functions OneMax and LeadingOnes. When talking about *run time,* we always mean the number of fitness function evaluations of an algorithm until an optimum is sampled for the first time. We state in Table 10.1 the asymptotic run times of a few algorithms on these benchmark functions. We note that

(i) the black-box complexity of OneMax is $\Theta\big(n/\log(n)\big)$ [AW09; DJW06] and

(ii) the black-box complexity of LeadingOnes is $\Theta\big(n\log(\log(n))\big)$ [Afs+13].

However, all black-box algorithms witnessing these run times are highly artificial. Consequently, $\Theta\big(n\log(n)\big)$ appears to be the best run time to aim for for these two benchmark problems.

Since random bit strings with independently sampled entries occur frequently in this work, we shall regularly use the following well-known variance-based additive Chernoff bounds.

▶ **Theorem 10.1 (Variance-based Additive Chernoff Bounds [Doe18, Theorem 10.12, Corollary 10.13]).** Let $n \in \mathbf{N}^+$ and let $(X_i)_{i\in[n]}$ be independent random variables such that, for all $i \in [n]$, it holds that $\mathrm{E}[X_i]-1 \leq X_i \leq \mathrm{E}[X_i]+1$. Further, let $X = \sum_{i=1}^{n} X_i$ and $\sigma^2 = \sum_{i=1}^{n} \mathrm{Var}[X_i] = \mathrm{Var}[X]$. Then, for all $\lambda \geq 0$, abbreviating $m = \min\{\lambda^2/\sigma^2, \lambda\}$, it holds that

$$\Pr\big[X \geq \mathrm{E}[X] + \lambda\big] \leq \mathrm{e}^{-\frac{1}{3}m} \text{ and}$$
$$\Pr\big[X \leq \mathrm{E}[X] - \lambda\big] \leq \mathrm{e}^{-\frac{1}{3}m}\ . \quad ◀$$

Last, we use the $\circ$ operator to denote string concatenation. For a bit string $H \in \{0,1\}^*$, let $|H|$ denote its length, and, for a $k \in [|H|]$, let $H[k]$ denote the *last* k bits in H. In addition to that, let $\emptyset$ denote the empty string.

Algorithm 2: The sig-cGA with parameter ε and significance function ›sig‹ (equation (10.1)) optimizing f

```
$t \leftarrow 0$;
$\boldsymbol{p}^{(t)} \leftarrow \frac{1}{2}$;
for $i \in [n]$ do $H_i \leftarrow \emptyset$ ;
repeat
    $\boldsymbol{x}, \boldsymbol{y} \leftarrow$ offspring sampled with respect to $\boldsymbol{p}^{(t)}$;
    $\boldsymbol{x} \leftarrow$ winner of $\boldsymbol{x}$ and $\boldsymbol{y}$ with respect to $f$;
    for $i \in [n]$ do
        $H_i \leftarrow H_i \circ \boldsymbol{x}_i$;
        if $\mathrm{sig}(\boldsymbol{p}_i^{(t)}, H_i) = \mathrm{up}$ then $\boldsymbol{p}_i^{(t+1)} \leftarrow 1 - 1/n$;
        else if $\mathrm{sig}(\boldsymbol{p}_i^{(t)}, H_i) = \mathrm{down}$ then $\boldsymbol{p}_i^{(t+1)} \leftarrow 1/n$;
        else $\boldsymbol{p}_i^{(t+1)} \leftarrow \boldsymbol{p}_i^{(t)}$;
        if $\boldsymbol{p}_i^{(t+1)} \neq \boldsymbol{p}_i^{(t)}$ then $H_i \leftarrow \emptyset$;
    $t \leftarrow t + 1$;
until termination criterion met;
```

10.3 The Significance-based Compact Genetic Algorithm

Our new algorithm – the *significance-based compact genetic algorithm* (sig-cGA; Algorithm 2) – is a univariate EDA similar to the cGA in that it also samples two offspring each generation and performs the same *selection* procedure. More specifically, let $\boldsymbol{x}$ and $\boldsymbol{y}$ denote both offspring sampled by the frequency vector during an iteration. Given a fitness function f, we rank $\boldsymbol{x}$ above $\boldsymbol{y}$ if $f(\boldsymbol{x}) > f(\boldsymbol{y})$ (as we maximize), and we rank $\boldsymbol{y}$ above $\boldsymbol{x}$ if $f(\boldsymbol{y}) > f(\boldsymbol{x})$. If $f(\boldsymbol{x}) = f(\boldsymbol{y})$, we rank them uniformly at random. The higher-ranked individual is called the *winner,* the other individual the *loser.*

In contrast to the cGA, the sig-cGA keeps a history of bit values of the winner for each position and only performs an update when a statistical significance within a history occurs. This approach far better aligns with the intuitive reasoning that an update should only be performed if there is valid evidence for a different frequency being better suited for sampling good individuals. Note that the sig-cGA is thus not an n-Bernoulli-λ-EDA, since it uses more information than its frequency vector and its samples in order to perform an update.

The key idea behind the update procedure of the sig-cGA is to notice a bias in its histories. More formally, consider a position $i \in [n]$ and any two individuals $\boldsymbol{x}$ and $\boldsymbol{y}$ that are identical except for position i. Assume that $\boldsymbol{x}_i > \boldsymbol{y}_i$. If the probability that $\boldsymbol{x}$ is the winner of the selection is higher than $\boldsymbol{y}$ being the winner, we speak of a *bias in selection* (for 1s) at position i. Analogously, we speak of a bias for 0s if the probability that $\boldsymbol{y}$ wins is higher than the probability that $\boldsymbol{x}$ wins. Usually, a fitness function introduces a bias into the selection and thus into the update.

The sig-cGA performs an update as follows: for each bit position $i \in [n]$, the sig-cGA keeps a history $H_i \in \{0, 1\}^*$ of all the bits sampled by the winner of each iteration since the last time $\boldsymbol{p}_i$ changed – the last bit denoting the latest entry. Observe that if there is no bias in selection at position i, the bits sampled by $\boldsymbol{p}_i$ follow a binomial distribution with a success probability of $\boldsymbol{p}_i$ and $|H_i|$ tries. We call this our *hypothesis*. Now, if we happen to find a sequence (starting from the latest entry) in H_i that significantly deviates from the hypothesis, we update $\boldsymbol{p}_i$ with respect to the bit value that occurred significantly, and we reset the history. We only use the following three frequency values:

- $1/2$: starting value;
- $1/n$: significance for 0s was detected;
- $1 - 1/n$: significance for 1s was detected.

We formalize *significance* by defining the threshold for all $\varepsilon, \mu \in \mathbf{R}^+$, where μ is the expected value of our hypothesis and ε is an algorithm-specific parameter:

$$s(\varepsilon, \mu) = \varepsilon \max\left\{\sqrt{\mu \ln(n)}, \ln(n)\right\} .$$

We say, for an $\varepsilon \in \mathbf{R}^+$, that a binomially distributed random variable X deviates significantly from a hypothesis $Y \sim \mathrm{Bin}(k, p)$, where $k \in \mathbf{N}^+$ and $p \in [0, 1]$, if there exists a $c \in \Omega(1)$ such that

$$\Pr\left[|X - \mathrm{E}[Y]| < s(\varepsilon, \mathrm{E}[Y])\right] \leq n^{-c} .$$

We now state our significance function $\mathrm{sig}\colon \left\{\frac{1}{n}, \frac{1}{2}, 1-\frac{1}{n}\right\} \times \{0, 1\}^* \to \{\text{up}, \text{stay}, \text{down}\}$, which scans a history for a significance. However, it does not scan the entire history but multiple subsequences of a history (always starting from the latest entry). This is done in order to quickly notice a change from an insignificant history to a significant one. Further, we only check in steps of powers of 2, as this is faster than checking each subsequence and we can be off from any length of a subsequence by a constant factor of at most 2. More formally, for all $H \in \{0,1\}^*$, we define, with ε being a parameter of the sig-cGA and recalling that $H[k]$ denotes the last k bits of H, that

$$\begin{aligned}
\mathrm{sig}\Big(\frac{1}{2}, H\Big) &= \begin{cases} \text{up} & \text{if } \exists m \in \mathbf{N}\colon \|H[2^m]\|_1 \geq \frac{2^m}{2} + s\big(\varepsilon, \frac{2^m}{2}\big), \\ \text{down} & \text{if } \exists m \in \mathbf{N}\colon \|H[2^m]\|_0 \geq \frac{2^m}{2} + s\big(\varepsilon, \frac{2^m}{2}\big), \\ \text{stay} & \text{else.} \end{cases}\\
\mathrm{sig}\Big(1 - \frac{1}{n}, H\Big) &= \begin{cases} \text{down} & \text{if } \exists m \in \mathbf{N}\colon \|H[2^m]\|_0 \geq \frac{2^m}{n} + s\big(\varepsilon, \frac{2^m}{n}\big), \\ \text{stay} & \text{else.} \end{cases}\\
\mathrm{sig}\Big(\frac{1}{n}, H\Big) &= \begin{cases} \text{up} & \text{if } \exists m \in \mathbf{N}\colon \|H[2^m]\|_1 \geq \frac{2^m}{n} + s\big(\varepsilon, \frac{2^m}{n}\big), \\ \text{stay} & \text{else.} \end{cases}
\end{aligned} \tag{10.1}$$

We stop at the first (minimum) length 2^m that yields a significance. Thus, we check a history H in each iteration at most $\log_2(|H|)$ times.

We now prove that the probability of detecting a significance at a position when there is no bias in selection (that is, a *false significance*) is small. We use this lemma in our proofs in order to argue that no false significances are detected with high probability.

▶ **Lemma 10.2.** For the sig-cGA, let $\varepsilon \geq 1$. Consider a position $i \in [n]$ of the sig-cGA and an iteration such that the distribution X of 1s of H_i follows a binomial distribution with $k \in \mathbf{N}^+$ trials and success probability $\boldsymbol{p}_i$, that is, there is no bias in selection at position i. Then the probability that $\boldsymbol{p}_i$ changes in this iteration is at most $n^{-\varepsilon/3}\log_2(k)$. ◀

Proof. In order for $\boldsymbol{p}_i$ to change, the number of 0s or 1s in X needs to deviate significantly from the hypothesis, which follows the same distribution as X by assumption. We use Theorem 10.1 in order to show that in such a scenario X deviates significantly from its expected value only with a probability of at most $n^{-\varepsilon/3}\log_2(k)$ for any number of trials at most k.

Let $\boldsymbol{p}'_i = \min\{\boldsymbol{p}_i, 1 - \boldsymbol{p}_i\}$. Note that in order for $\boldsymbol{p}_i$ to change, a significance of values sampled with probability $\boldsymbol{p}'_i$ needs to be sampled. That is, for $\boldsymbol{p}_i = 1/2$, either a significant amount of 1s or 0s needs to occur; for $\boldsymbol{p}_i = 1 - 1/n$, a significant amount of 0s needs to occur; and, for $\boldsymbol{p}_i = 1/n$, a significant amount of 1s needs to occur. Further, let X' denote the number of values we are looking for a significance within $k' \leq k$ trials. That is, if $\boldsymbol{p}_i = 1/2$, then X' is either the number of 1s or 0s; if $\boldsymbol{p}_i = 1 - 1/n$, then X' is the number of 0s; and if $\boldsymbol{p}_i = 1/n$, then X' is the number of 1s.

Given the definition of $\boldsymbol{p}'_i$, we see that $\mathrm{E}[X'] = k'\boldsymbol{p}'_i$ and $\mathrm{Var}[X'] = k'\boldsymbol{p}_i(1 - \boldsymbol{p}_i) \leq k'\boldsymbol{p}'_i$. Since we aim at apply Theorem 10.1, let $\lambda = s(\varepsilon, \mathrm{E}[X']) = s(\varepsilon, k'\boldsymbol{p}'_i)$ and $\sigma^2 = \mathrm{Var}[X']$.

First, consider the case that $\lambda = s(\varepsilon, k'\boldsymbol{p}'_i) = \varepsilon \ln(n)$, that is, that $\sqrt{k'\boldsymbol{p}'_i \ln(n)} \leq \ln(n)$, due to the definition of s, which is equivalent to $k' \leq (1/\boldsymbol{p}'_i)\ln(n)$. Note that $\lambda^2/\sigma^2 \geq \varepsilon^2 \ln(n) \geq \ln(n)$, as $\varepsilon \geq 1$. Thus, $\min\{\lambda^2/\sigma^2, \lambda\} \geq \varepsilon \ln(n)$.

Now consider the case $\lambda = s(\varepsilon, k'\boldsymbol{p}'_i) = \varepsilon\sqrt{k'\boldsymbol{p}'_i \ln(n)}$, that is, that $\sqrt{k'\boldsymbol{p}'_i \ln(n)} \geq \ln(n)$, which is equivalent to $k' \geq (1/\boldsymbol{p}'_i)\ln(n)$. We see that $\lambda \geq \varepsilon \ln(n)$ and $\lambda^2/\sigma^2 \geq \varepsilon^2 \ln(n)$. Hence, as before, we get $\min\{\lambda^2/\sigma^2, \lambda\} \geq \varepsilon \ln(n)$.

Combining both cases and applying Theorem 10.1, we get

$$\begin{aligned}
\Pr\left[X' \geq k'\boldsymbol{p}'_i + s(\varepsilon, k'\boldsymbol{p}'_i)\right] &= \Pr\left[X' \geq \mathrm{E}[X'] + \lambda\right] \\
&\leq \mathrm{e}^{-\frac{1}{3}\min\left\{\frac{\lambda^2}{\sigma^2}, \lambda\right\}} \\
&\leq \mathrm{e}^{-\frac{\varepsilon}{3}\ln(n)} \\
&= n^{-\frac{\varepsilon}{3}} .
\end{aligned}$$

That is, the probability of detecting a (false) significance during k' trials is at most $n^{-\varepsilon/3}$. Since we look for a significance a total of at most $\log_2(k)$ times during an iteration, we get by a union bound that the probability of detecting a significance within a history of length k is at most $n^{-\varepsilon/3}\log_2(k)$. ■

Lemma 10.2 bounds the probability of detecting a false significance within a single iteration if there is no bias in selection. The following corollary trivially bounds the probability of detecting a false significance within any number of iterations.

▶ **Corollary 10.3.** Consider the sig-cGA (Algorithm 2) with $\varepsilon \geq 1$ running for k iterations such that, during each iteration, for each position $i \in [n]$, a 1 is added to H_i with probability $\boldsymbol{p}_i$. Then the probability that at least one frequency changes during an interval of $k' \leq k$ iterations is at most $k'n^{1-\varepsilon/3}\log_2(k)$. ◀

Proof. For any $i \in [n]$ during any of the k iterations, by Lemma 10.2, the probability that $\boldsymbol{p}_i$ changes is at most $n^{-\varepsilon/3} \log_2(k)$. Via a union bound (Theorem 2.17) over all k' relevant iterations and all n frequencies, the statement follows. ■

10.3.1 Efficient Implementation of the sig-cGA

In order to reduce the number of operations performed (computational cost) of the sig-cGA, we only check significance in historic data of lengths that are a power of 2. By saving the whole history but precomputing the number of 1s in the power-of-two intervals, a significance check can be done in time logarithmic in the history length; the necessary updates of this data structure can be done in logarithmic time (per bit-position) as well. With this implementation, the main loop of the sig-cGA has a computational cost of $\mathrm{O}\big(\sum_{i=1}^{n} |H_i|\big)$. Since the histories are never longer than the run time (twice the number of iterations), we see that the computational cost is at most $\mathrm{O}\big(nT \log(T)\big)$, when the run time is T. Since for most EAs working on bit string representations of length n the computational cost is larger than the run time by at least a factor of n, we see that our significance approach is not overly costly in terms of computational cost.

What appears unfavorable though is the memory usage caused by storing the full history. For this reason, we now sketch a way to condense the history so that it only uses space logarithmic in the length of the full history. This approach does not allow to access exactly the number of 1s (or 0s) in all power-of-two length histories. However, it allows for each $\ell \in [|H_i|]$ to access the number of 1s in some interval of length ℓ' with $\ell' \in [\ell..2\ell - 1]$. For reasons of readability, we shall in the subsequent analyses nevertheless regard the original sig-cGA, but it is quite clear that the mildly different accessibility of the history in the now-proposed condensed implementation will not change the asymptotic run times shown in this work.

For our condensed storage of the history, we have a list of blocks, each storing the number of 1s in some discrete interval $[t_1..t_2]$ of length equal to a power of two (including 1). When a new item has to be stored, we append a block of size 1 to the list. Then, traversing the list in backward direction, we check if there are three consecutive blocks of the same size, and if so, we merge the two earliest ones into a new block of twice the size. By this, we always maintain a list of blocks such that, for a certain power 2^k, there are between one and two blocks of length 2^j for all $j \in [0..k - 1]$. This structural property implies both that we only have a logarithmic number of blocks (as we have $k \in \mathrm{O}\big(\log(|H_i|)\big)$) and that we can (in amortized constant time) access all historic intervals consisting of full blocks, which in particular implies that we can access an interval with length in $[2^j..2^{j+1}-1]$ for all $j \in [0..k]$.

10.3.2 Run Time Results for LeadingOnes and OneMax

We now prove our main results, that is, upper bounds of $\mathrm{O}\big(n\log(n)\big)$ for the expected run time of the sig-cGA on LeadingOnes and OneMax. Note that the sig-cGA samples two offspring each iteration. Thus, up to a constant factor of 2, the expected run time is equal to the expected number of iterations until an optimum is sampled. In our proofs, we only consider the number of iterations.

We mention briefly that the sig-cGA is unbiased in the sense of Lehre and Witt [LW12] and as explained in Chapter 5, that is, it treats bit values and bit positions in a symmetric fashion. Consequently, all of our results hold not only for OneMax and LeadingOnes but for the entire respective function classes.

In our proofs, we use the following lemma to bound probabilities split up by the law of total probability (Theorem 2.1).

▶ **Lemma 10.4.** Let $\alpha, \beta, x, y \in \mathbf{R}$ such that $x \leq y$ and $\alpha \leq \beta$. Then

$$\alpha x + (1-\alpha)y \geq \beta x + (1-\beta)y\ . \qquad ◀$$

We start with the expected run time of the sig-cGA on LeadingOnes.

LeadingOnes

We show that the frequencies are set to $1 - 1/n$ sequentially from the most significant bit position to the least significant, that is, from left to right. With high probability, no frequency is decreased until the optimization process is finished. Thus, a frequency $\boldsymbol{p}_i$ will stay at $1/2$ until all of the frequencies to its left are set to $1 - 1/n$. Then $\boldsymbol{p}_i$ will become relevant for selection, as all of the frequencies left to it will only sample 1s with high probability. This results in a significant surplus of 1s being saved at position i, and $\boldsymbol{p}_i$ will be set to $1 - 1/n$ within $\mathrm{O}\big(\log(n)\big)$ iterations and remain there. Then frequency $\boldsymbol{p}_{i+1}$ becomes relevant for selection. As we need to set n frequencies to $1 - 1/n$, we get a run time of $\mathrm{O}\big(n\log(n)\big)$.

▶ **Theorem 10.5.** Consider the sig-cGA with $\varepsilon > 12$ being a constant. Its run time on LeadingOnes is $\mathrm{O}\big(n\log(n)\big)$ with high probability and in expectation. ◀

Proof. We split this proof into two parts and start by showing that the run time is $\mathrm{O}\big(n\log(n)\big)$ with high probability. Then we prove the expected run time.

Run time with high probability. For the first part of the proof, we consider the first $\mathrm{O}\big(n\log(n)\big)$ iterations of the sig-cGA and condition on the event that no frequency decreases during this time, that is, no (false) significance of 0s is detected. Note that for any position $i \in [n]$, the probability of saving a 1 in H_i is at least $\boldsymbol{p}_i$, as the selection with respect to LeadingOnes has a bias for 1s. Thus, by Corollary 10.3, the probability that at least one frequency decreases during $\mathrm{O}\big(n\log(n)\big)$ iterations is at most $\mathrm{O}\big(n^{2-\varepsilon/3}\log^2(n)\big)$, which is, as $\varepsilon > 12$, in $\mathrm{O}\big(n^{-\varepsilon'}\big)$, for an $\varepsilon' > 2$. Thus, with high probability, no frequency decreases during $\mathrm{O}\big(n\log(n)\big)$ iterations.

The main idea now is to show that the leftmost frequency that is different from $1 - 1/n$ has a significant surplus of 1s in its history strong enough so that, after a logarithmic number of iterations, we change such a frequency from its initial value of $1/2$ to $1 - 1/n$. For the second part of the proof, we will use a similar argument, but the frequency will be at $1/n$, and it will take $\mathrm{O}\big(n\log(n)\big)$ steps to get to $1 - 1/n$. Since the calculations for both scenarios are very similar, we combine them in the following.

In order to make this idea precise, we now consider an iteration such that there is a frequency $\boldsymbol{p}_i \in \{1/n, 1/2\}$ such that, for all $j < i$, we have $\boldsymbol{p}_j = 1 - 1/n$. We lower-bound the probability of saving a 1 in H_i in order to get an upper bound on the expected time until we detect the significance necessary to update $\boldsymbol{p}_i$ to $1-1/n$. When considering position i, we assume an empty history although it is most likely not. We can do so, since the sig-cGA checks for a significance in different sub-histories of H_i (starting from the latest entry). Thus, we only consider sub-histories that go as far as the point in time when all indices less than i were at $1 - 1/n$.

Let O denote the event that we save a 1 this iteration, and let A denote the event that at least one of the two offspring during this iteration has a 0 at a position in $[i - 1]$. Note that event A means that the bit at position i of the winning individual is not relevant for selection. Hence, if A occurs, we save a 1 with probability $\boldsymbol{p}_i \in \{1/n, 1/2\}$. Otherwise, that is, the bit at position i is relevant for selection, we save a 1 with probability $1 - \boldsymbol{p}_i^2$ (that is, if we do not sample two 0s). Formally, by Theorem 2.1, we get

$$\Pr[O] = \Pr[A] \cdot \boldsymbol{p}_i + \Pr\left[\overline{A}\right] \cdot (1 - \boldsymbol{p}_i^2)\,,$$

which is a convex combination of $\boldsymbol{p}_i$ and $1 - \boldsymbol{p}_i^2$. Thus, according to Lemma 10.4, we get a lower bound if we decrease the factor of the larger term, namely, $\Pr\left[\overline{A}\right]$. The event $\overline{A}$ occurs if and only if both offspring have only 1s at the positions 1 through $i - 1$:

$$\Pr\left[\overline{A}\right] = \left(1 - \frac{1}{n}\right)^{2(i-1)}\,,$$

as we assumed that all frequencies at indices less than i are already at $1 - 1/n$. Note that this term is minimal for $i = n$. Thus, we get $\Pr\left[\overline{A}\right] \geq \mathrm{e}^{-2}$ by using the well-known inequality $(1 - 1/n)^{n-1} \geq \mathrm{e}^{-1}$ [AS65, Inequality 4.2.34]. Overall, we get, noting that $1 - \boldsymbol{p}_i^2 \geq (3/2)\boldsymbol{p}_i$ for $\boldsymbol{p}_i \in \{1/n, 1/2\}$,

$$\begin{aligned}\Pr[O] &\geq (1 - \mathrm{e}^{-2}) \cdot \boldsymbol{p}_i + \mathrm{e}^{-2} \cdot (1 - \boldsymbol{p}_i^2)\\ &\geq (1 - \mathrm{e}^{-2}) \cdot \boldsymbol{p}_i + \frac{3}{2}\mathrm{e}^{-2}\boldsymbol{p}_i\\ &= \left(1 + \frac{1}{2}\mathrm{e}^{-2}\right) \cdot \boldsymbol{p}_i\,.\end{aligned}$$

Let $X \sim \mathrm{Bin}\big(k, (1 + \mathrm{e}^{-2}/2)\boldsymbol{p}_i\big)$ denote a random variable that is stochastically dominated by the real process of saving 1s at position i. In order to get a bound on the number of iterations $k \in \mathbf{N}^+$ that we need for detecting a significance of 1s, we bound the probability of a significance not occurring in a history of length k, that is, we save fewer than $k\boldsymbol{p}_i + s(\varepsilon, k\boldsymbol{p}_i)$ 1s. We get

$$\Pr[X < k\boldsymbol{p}_i + s(\varepsilon, k\boldsymbol{p}_i)] \leq \Pr\left[X \leq \mathrm{E}[X] - \left(\frac{k}{2}\mathrm{e}^{-2}\boldsymbol{p}_i - s(\varepsilon, k\boldsymbol{p}_i)\right)\right]\,,$$

where the minuend is positive if $(k/2)\mathrm{e}^{-2}\boldsymbol{p}_i > s(\varepsilon, k\boldsymbol{p}_i)$, which is the case for $k > (4/\boldsymbol{p}_i)\mathrm{e}^4\varepsilon^2 \ln(n) > \ln(n)$, since we assume that $\varepsilon > 12$. Let $c = (4/\boldsymbol{p}_i)\mathrm{e}^4\varepsilon^2$. For $k \geq 4c\ln(n)$, we get that $(k/2)\mathrm{e}^{-2}\boldsymbol{p}_i - s(\varepsilon, k\boldsymbol{p}_i) \geq (k/4)\mathrm{e}^{-2}\boldsymbol{p}_i =: \lambda$. By applying Theorem 10.1 for any $k \geq 4c\ln(n)$ and noting that $\mathrm{Var}[X] = k\boldsymbol{p}_i(1 - \boldsymbol{p}_i) \geq \lambda$ and, thus, $\lambda^2/\mathrm{Var}[X] \leq \lambda$, we get, using $\mathrm{Var}[X] \leq k\boldsymbol{p}_i$,

$$\begin{aligned}\Pr[X < k\boldsymbol{p}_i + s(\varepsilon, k\boldsymbol{p}_i)] &\leq \Pr\left[X \leq \mathrm{E}[X] - \frac{k}{4}\mathrm{e}^{-2}\boldsymbol{p}_i\right]\\ &\leq \mathrm{e}^{-\frac{1}{3}\cdot\frac{\lambda^2}{\mathrm{Var}[X]}}\\ &\leq \mathrm{e}^{-\frac{1}{3}\cdot\frac{k^2\mathrm{e}^{-4}\boldsymbol{p}_i^2}{16k\boldsymbol{p}_i}}\\ &= \mathrm{e}^{-\frac{1}{3}\cdot\frac{k\mathrm{e}^{-4}\boldsymbol{p}_i}{16}}\\ &\leq n^{-\frac{1}{3}\cdot\frac{c\mathrm{e}^{-4}\boldsymbol{p}_i}{4}}\\ &= n^{-\frac{\varepsilon^2}{3}} .\end{aligned}$$

Thus, with probability at least $1 - n^{-\varepsilon^2/3}$, the frequency $\boldsymbol{p}_i$ will be set to $1 - 1/n$ after $(4/\boldsymbol{p}_i)\mathrm{e}^4\varepsilon^2 \ln(n) \in \mathrm{O}\big((1/\boldsymbol{p}_i)\log(n)\big)$ iterations. Further, via a union bound (Theorem 2.17) over all n frequencies, the probability of any such frequency not being updated to $1-1/n$ after $\mathrm{O}\big((1/\boldsymbol{p}_i)\log(n)\big)$ iterations is at most $n^{1-\varepsilon^2/3} \leq n^{-47}$, as $\varepsilon > 12$. Hence, with high probability, all frequencies will be set to $1 - 1/n$.

For the first part of this proof, that is, assuming that no frequency is at $1/n$, and taking together the results of all frequencies being updated to $1 - 1/n$, each in time $\mathrm{O}\big((1/\boldsymbol{p}_i)\log(n)\big) = \mathrm{O}\big(\log(n)\big)$, and assuming that no frequency at $1/2$ or $1-1/n$ decreases, all with high probability, yields that all frequencies reach $1-1/n$ within $\mathrm{O}\big(n\log(n)\big)$ iterations. Then the optimum is sampled with probability $(1 - 1/n)^n \geq 1/(2\mathrm{e}) \in \Omega(1)$ according to Theorem 2.22, that is, with constant probability. Hence, we have to wait $\mathrm{O}\big(\log(n)\big)$ additional iterations in order to sample the optimum with high probability.

Expected run time. For the second part of this proof, that is, for the expected run time, we are left to bound the expected time if a frequency decreases during the initial $\mathrm{O}\big(n\log(n)\big)$ iterations, which only happens with a probability of $\mathrm{O}\big(n^{-\varepsilon'}\big)$, where $\varepsilon' > 2$, as we discussed at the beginning of the first part. Due to Corollary 10.3, during $t \in \mathbf{N}^+$ iterations and considering an interval of length $t' \leq t$, no frequency decreases with a probability of at least $1 - t'n^{1-\varepsilon/3}\log_2(t)$. By assuming $t \leq n^{2n}$ and $t' \in \Theta\big(n^2\log(n)\big)$, with high probability, no frequency decreases during such an interval, as $\varepsilon > 12$.

By using the result calculated in the first part, we see that a leftmost frequency $\boldsymbol{p}_i$ at $1/n$ is increased to during $\mathrm{O}\big((1/\boldsymbol{p}_i)\log(n)\big) = \mathrm{O}\big(n\log(n)\big)$ iterations with high probability. Thus, overall, the sig-cGA finds the optimum during an interval of length $t' \in \Theta\big(n^2 \log(n)\big)$ with high probability, as n frequencies need to be increased to $1 - 1/n$. We pessimistically assume that the optimum is only found with a probability of at least $1/2$ during t' iterations. Hence, the expected run time in this case is $2t' \in \Theta(t')$.

Last, we assume that we did not find the optimum during n^{2n} iterations, which only happens with a probability of at most $2^{-n^{2n}/t'}$. Then, the expected run time is at most n^n by pessimistically assuming that all frequencies are at $1/n$.

Combining all of the three different regimes we just discussed, we see that there is a constant $c > 0$ we can upper bound the expected run time by

$$cn \log(n) + cn^{-\varepsilon'} \cdot t' + 2^{-n^{2n}/t'} \cdot n^n \in \mathrm{O}\big(n \log(n)\big) ,$$

which concludes the proof. ■

The proof of Theorem 10.5 shows us that the sig-cGA rapidly makes progress when optimizing LeadingOnes. In fact, after $\mathrm{O}\big(i \log(n)\big)$ iterations, with $i \in [n]$, the sig-cGA finds a solution with fitness i with high probability and in expectation. Thus, from a fixed-budget perspective, the sig-cGA performs very well on LeadingOnes.

The reason that the sig-cGA optimizes LeadingOnes so quickly is that the probability of saving a 1 at position i is increased by a constant factor once all frequencies at positions less than i are at $1 - 1/n$. This boost is a result of position i being the most relevant position for selection, assuming that all bits at positions less than i are 1.

BinVal A very similar boost in relevance occurs when considering the function BinVal, which returns the bit value of a bit string. Formally, BinVal is defined as

$$\textsc{BinVal}(\boldsymbol{x}) = \sum_{i=1}^{n} 2^{n-i} \boldsymbol{x}_i \ .$$

Note that the most significant bit is the leftmost.

BinVal imposes a lexicographic order from left to right on a bit string $\boldsymbol{x}$, since a bit $\boldsymbol{x}_i$ has a greater weight than the sum of all weights at positions greater than i. This is similar to LeadingOnes. The main difference is that, for BinVal, a position i can also be relevant for selection when bits at positions less than i

are 0. More formally, for LeadingOnes, position i is only relevant for selection when all of the bits at positions less than i are 1, whereas position i is relevant for selection for BinVal when all the bits at positions less than i are *the same.* With this insight, we can adapt the proof of Theorem 10.5 for BinVal and get the following corollary.

▶ **Corollary 10.6.** Consider the sig-cGA (Algorithm 2) with $\varepsilon > 12$ being a constant. Its run time on BinVal is $\mathrm{O}\big(n\log(n)\big)$ with high probability and in expectation. ◀

Proof. We can use the same arguments as in the proof of Theorem 10.5. The only difference is in how we define the event A that position $i \in [n]$, which is the leftmost position not at $1 - 1/n$, is *not* relevant for selection. As described above, let A denote the event that there exists a position $j \in [i-1]$ such that, for the two offspring $\boldsymbol{x}$ and $\boldsymbol{y}$ sampled during the iteration we consider, $\boldsymbol{x}_i \neq \boldsymbol{y}_i$ holds. Further, let O denote again the event that we save a 1 this iteration. We see that the following equation also holds for BinVal:

$$\Pr[O] = \Pr[A] \cdot \boldsymbol{p}_i + \Pr\big[\overline{A}\big] \cdot (1 - \boldsymbol{p}_i^2)\ .$$

Again, according to Lemma 10.4, we get a lower bound for $\Pr[O]$ if we get a lower bound for $\Pr\big[\overline{A}\big]$. The event $\overline{A}$ occurs if all the bits sampled at positions in $[i-1]$ have the same value. Thus, a subset of this event is that all of the bits sampled at positions in $[i-1]$ are 1, which yields that

$$\Pr\big[\overline{A}\big] \geq \left(1 - \frac{1}{n}\right)^{2(i-1)}\ .$$

This probability is minimized by choosing $i = n$. Hence, we get $\Pr\big[\overline{A}\big] \geq \mathrm{e}^{-2}$, which is the same lower bound for $\overline{A}$ as in the proof of Theorem 10.5.

Since this is the only part of the proof where the fitness function comes into play, the remaining proof is identical to the proof of Theorem 10.5. ■

BinVal is often considered one extremal case of the class of linear functions, as its weights impose a lexicographic order on the bit positions. The other extreme is OneMax, where all weights are identical and basically no order among the positions exists. Combining the result of Corollary 10.6 with the result of Theorem 10.8, which we prove in the next section, we see that the sig-cGA optimizes both functions in $\mathrm{O}(n \log(n))$. It remains an open question whether the sig-cGA is capable of optimizing any linear function in that time, a feat that the $(1 + 1)$ EA, a classical EAs, is known to be capable of [DJW02]. Contrary to that, it was proven for the cGA, which is an EDA, that it performs worse on BinVal than on OneMax [Dro06]. Especially, Witt [Wit18] recently proved a bound of $\Omega(n^2)$ for the cGA without a margin on LeadingOnes. Thus, a uniform performance on the class of linear functions would be a great feat for an EDA.

We would like to note that the result of Droste [Dro06] considered the cGA without a margin, that is, the frequencies could reach values of 0. Once this is the case, the algorithm is stuck (as it only samples 0 at this position) and the optimization fails. It is unknown up to date whether the cGA still performs worse on BinVal when the frequencies are bound to the interval $[1/n, 1 - 1/n]$. However, the main idea of Droste's proof that frequencies drop very low remains. Thus, if sufficiently many frequencies were to drop to $1/n$, the cGA would still perform badly on BinVal. Note that this is exactly the problem that the sig-cGA circumvents with its update rule, resulting in its run time of $\mathrm{O}(n \log(n))$.

OneMax

For our next result, we make use of the following lemma based on a well-known estimate of binomial coefficients close to the center. A proof was given by, for example, Doerr and Winzen [DW14b]. We use it to show how likely it is that two individuals sampled from the sig-cGA have the same OneMax value.

▶ **Lemma 10.7 ([DW14b, Lemma 8]).** For $c \in \Theta(1)$, $\ell \in \mathbf{N}^+$, let $k \in [\ell/2 \pm c\sqrt{\ell}]$ and let $X \sim \mathrm{Bin}(1/2, \ell)$. Then $\Pr[X = k] \in \Omega(1/\sqrt{\ell})$. ◀

The next theorem shows that the sig-cGA is also able to optimize OneMax within the same asymptotic time like many other EAs. For the proof, we show that during *all* of the $\mathrm{O}(n \log(n))$ iterations, each position can become relevant for selection with a decent probability of $\Omega(1/\sqrt{n})$. In contrast to LeadingOnes, there is no sudden change in the probability that 1s are saved. Thus, it takes $\mathrm{O}(n \log(n))$ iterations to set a frequency to $1 - 1/n$. However, this is done for all frequencies in parallel. Thus, the overall run time remains $\mathrm{O}(n \log(n))$.

▶ **Theorem 10.8.** Consider the sig-cGA with $\varepsilon > 12$ being a constant. Its run time on OneMax is $\mathrm{O}(n \log(n))$ with high probability and in expectation. ◀

Proof. We first show that the run time holds with high probability. Then we prove the expected run time.

Run time with high probability. We consider the first $\mathrm{O}\big(n\log(n)\big)$ iterations and condition on the event that no frequency decreases during that time. This can be argued in the same way as at the beginning in the proof of Theorem 10.5.

The main idea now is to show that, for any frequency at $1/2$, $\mathrm{O}\big(n\log(n)\big)$ iterations are enough in order to detect a significance in 1s. This happens in parallel for all frequencies. For our argument to hold, it is only important that all the other frequencies are at $1/2$ or $1-1/n$, which we condition on.

Similar to the proof of Theorem 10.5, when proving the expected run time, we use that if all frequencies start at $1/n$, they are set to $1-1/n$ with high probability within $\mathrm{O}\big(n^2\log(n)\big)$ iterations in parallel. Thus, we combine both cases in the following.

Let $s \in \{1/2, 1/n\}$ denote the starting value of a frequency. Formally, during any of the $\mathrm{O}\big((n/s)\log(n)\big)$ iterations, let $\ell \in [n]$ denote the number of frequencies at s. Then $n-\ell$ frequencies are at $1-1/n$. Further, consider a position $i \in [n]$ with $\boldsymbol{p}_i = s$. We show that such a position will sample 1s significantly more often than the hypothesis by a factor of $\Theta\big(1/\sqrt{\ell}\big)$. Then $\boldsymbol{p}_i$ will be updated to $1-1/n$ within $\mathrm{O}\big((\ell/s)\log(n)\big)$ iterations.

In order to show that 1s are significantly more often saved than assumed, we proceed as follows: we consider that all bits but bit i of both offspring during any iteration have been sampled. If the number of 1s of both offspring differs by more than one, bit i cannot change the outcome of the selection process – bit i will be 1 with probability $\boldsymbol{p}_i$. However, if the number of 1s differs by at most one, then the outcome of bit i in both offspring has an influence on whether a 1 is saved or not – this introduces a bias toward saving a significant amount of 1s.

Let O denote the event to save a 1 at position i this iteration, and let A denote the event that the numbers of 1s (excluding position i) of both offspring differ by at least two during that iteration. Then the probability to save a 1, conditional on A, is $\boldsymbol{p}_i$.

In the case of $\overline{A}$, we make a case distinction with respect to the absolute difference of the number of 1s of both offspring, excluding position i. If the difference is zero, then a 1 will be saved if not both offspring sample a 0, which happens with probability $1-\boldsymbol{p}_i^2$. If the absolute difference is one, then a 1 will be saved if the winner (with respect to all bits but bit i) samples a 1 (with probability $\boldsymbol{p}_i$) or if it samples a 0, the loser samples a 1, and the loser is chosen during selection, which happens with probability $(1/2)\boldsymbol{p}_i(1 - \boldsymbol{p}_i) \geq (1/4)\boldsymbol{p}_i$. Overall, the probability that a 1 is saved is at least $\boldsymbol{p}_i + (1/4)\boldsymbol{p}_i = (5/4)\boldsymbol{p}_i$ in the case of $\overline{A}$, as this is less than $1 - \boldsymbol{p}_i^2$ for $\boldsymbol{p}_i \in \{1/2, 1/n\}$.

Combining both cases and using Theorem 2.1, we see that

$$\Pr[O] \geq \Pr[A] \cdot \boldsymbol{p}_i + \Pr\left[\overline{A}\right] \cdot \frac{5}{4}\boldsymbol{p}_i\ ,$$

which we lower-bound by determining a lower bound for $\Pr\left[\overline{A}\right]$, according to Lemma 10.4.

With respect to $\Pr\left[\overline{A}\right]$, we first note that the probability of all $n - \ell$ frequencies at $1 - 1/n$ sampling a 1 for both offspring is $(1 - 1/n)^{2(n-\ell)} \geq \mathrm{e}^{-2}$, as $n - \ell \leq n - 1$. Similarly, all frequencies at $1/n$ (but $\boldsymbol{p}_i$) will sample a 0 for both offspring with a probability of at least $(1 - 1/n)^{2(n-1)} \geq \mathrm{e}^{-2}$, too.

Now we only consider the difference of 1s sampled with respect to $\ell' \leq \ell - 1$ (for $\ell \geq 2$) positions with frequencies at $1/2$, that is, all remaining positions but i we did not consider so far. Since all of these frequencies are at $1/2$, the expected number of 1s is $\ell'/2$. Due to Theorem 10.1 (or, alternatively, Chebyshev's inequality), the probability of deviating from this value by more than $\sqrt{\ell'/2}$ is at most a constant $c < 1$. Conditional on sampling a number of 1s in the range of $\ell'/2 \pm \sqrt{\ell'/2}$, the probability to sample $k \in [\ell'/2 - \sqrt{\ell'/2}..\ell'/2 + \sqrt{\ell'/2}]$ 1s is, due to Lemma 10.7, $\Omega\left(1/\sqrt{\ell'}\right)$, since all ℓ' frequencies are at $1/2$. Thus, by the law of total probability (Theorem 2.1), the probability that both offspring have the same number of 1s or differ only by one, that is, $\Pr[\overline{A}]$, is, for a constant $d > 0$, at least $d/\sqrt{\ell'}$. Hence, we get, for a sufficiently small constant $d' > 0$, factoring in the probability of $1 - c$ of the number of 1s being concentrated around $\ell'/2$ and the remaining $n - \ell$ positions only sampling 1s, that

$$\begin{aligned}\Pr[O] &\geq \left(1 - \mathrm{e}^{-4}(1 - c)\frac{d}{\sqrt{\ell'}}\right) \cdot \boldsymbol{p}_i + \mathrm{e}^{-4}(1 - c)\frac{d}{\sqrt{\ell'}} \cdot \frac{5}{4}\boldsymbol{p}_i \\ &\geq \left(1 + \frac{d'}{\sqrt{\ell}}\right)\boldsymbol{p}_i\ .\end{aligned}$$

This means that the sig-cGA expects 1s to occur with probability $\boldsymbol{p}_i$, but they occur with a probability of at least $(1 + d'/\sqrt{\ell})\boldsymbol{p}_i$. Note that for the case $\ell = 1$, that is, $\ell' = 0$, conditional on the remaining $n - \ell$ positions only sampling 1s, we have $\Pr\left[\overline{A}\right] = 1$ and hence $\Pr[O] \geq (1 - \mathrm{e}^{-2}) \cdot \boldsymbol{p}_i + \mathrm{e}^{-2} \cdot (5/4)\boldsymbol{p}_i$. Thus, we use $(1 + d'/\sqrt{\ell})\boldsymbol{p}_i$ as a lower bound for $\Pr[O]$ in all cases for ℓ, for an appropriately chosen d'.

Analogous to the proof of Theorem 10.5, let $X \sim \mathrm{Bin}\big(k, (1+d'/\sqrt{\ell})\boldsymbol{p}_i\big)$ denote a random variable that is stochastically dominated by the real process of saving 1s at position i. We bound the probability of not detecting a significance of 1s after k iterations, that is,

$$\Pr\big[X < k\boldsymbol{p}_i + s(\varepsilon, k\boldsymbol{p}_i)\big] \leq \Pr\left[X \leq \mathrm{E}[X] - \left(\frac{kd'}{\sqrt{\ell}}\boldsymbol{p}_i - s(\varepsilon, k\boldsymbol{p}_i)\right)\right] .$$

Let $k \geq 4(\varepsilon^2/d'^2)(\ell/\boldsymbol{p}_i)\ln(n)$. Then $(kd'/\sqrt{\ell})\boldsymbol{p}_i - s(\varepsilon, k\boldsymbol{p}_i) \geq \big(kd'/(2\sqrt{\ell})\big)\boldsymbol{p}_i =: \lambda$. By noting that $\mathrm{Var}[X] = k\boldsymbol{p}_i(1 - \boldsymbol{p}_i) \geq \lambda$ for d' sufficiently small and, thus, $\lambda^2/\mathrm{Var}[X] \leq \lambda$, we get by applying Theorem 10.1 and using $\mathrm{Var}[X] \leq k\boldsymbol{p}_i$, that

$$\begin{aligned}\Pr\big[X < k\boldsymbol{p}_i + s(\varepsilon, k\boldsymbol{p}_i)\big] &\leq \Pr\left[X \leq \mathrm{E}[X] - \frac{kd'}{2\sqrt{\ell}}\boldsymbol{p}_i\right] \\ &\leq \mathrm{e}^{-\frac{1}{3}\cdot\frac{k^2 d'^2 \boldsymbol{p}_i^2}{4\ell k \boldsymbol{p}_i}} \\ &= \mathrm{e}^{-\frac{1}{3}\cdot\frac{kd'^2}{4\ell}\boldsymbol{p}_i} \\ &\leq \mathrm{e}^{-\frac{1}{3}\varepsilon^2\ln(n)} \\ &= n^{-\frac{1}{3}\varepsilon^2} .\end{aligned}$$

Thus, with a probability of at least $1 - n^{-\varepsilon^2/3}$, frequency $\boldsymbol{p}_i$ is set to $1 - 1/n$ after $4(\varepsilon^2/d'^2)(\ell/\boldsymbol{p}_i)\ln(n) \in \mathrm{O}\big((\ell/\boldsymbol{p}_i)\log(n)\big)$ iterations. Further, via a union bound over all n frequencies, the probability of any such frequency not being updated to $1 - 1/n$ after $\mathrm{O}\big((\ell/\boldsymbol{p}_i)\log(n)\big)$ iterations is at most $n^{1-\varepsilon^2/3} \leq n^{-47}$, as $\varepsilon > 12$. Hence, with high probability, all frequencies are set to $1 - 1/n$.

Since our argument for position i was made for an arbitrary i and independent of the other positions, and since all n frequencies start at $1/2$ (that is, $\ell = n$), we have to wait at most $\mathrm{O}\big(n\log(n)\big)$ iterations until all frequencies are set to $1 - 1/n$ with high probability. Then, with a probability of at least $(1 - 1/n)^n \geq 1/(2\mathrm{e}) \in \Omega(1)$, the optimum is sampled. Hence, after $\mathrm{O}\big(\log(n)\big)$ additional iterations, the optimum is sampled with high probability.

Expected run time. The expected run time can be proven similarly as argued in the second part of the proof of Theorem 10.5. The main difference here is that, assuming all frequencies are at $1/n$, with high probability, all frequencies will increase during $\mathrm{O}\big(n^2 \log(n)\big)$ iterations (in parallel, not sequentially), as we just discussed. Further, since $\varepsilon > 12$, no frequency will decrease during an interval of such length with high probability. ■

Note that although the expected run time of the sig-cGA is asymptotically the same on LeadingOnes and OneMax, the reason is quite different: for LeadingOnes, the sig-cGA sets its frequencies quickly consecutively to $1 - 1/n$, as it only needs $\mathrm{O}\big(\log(n)\big)$ iterations per frequency in expectation. This is due to the bias for saving 1s being very large (constant, in fact) when all frequencies to the left are at $1 - 1/n$, that is, when it is very likely that bit i is relevant for selection. In Section 6.4.1, we exploited this fact in the analysis (and design) of the scGA heavily, which is why it, too, has an expected run time of $\mathrm{O}\big(n \log(n)\big)$ on LeadingOnes. However, when not all frequencies to the left of a position are at $1 - 1/n$, the bias is almost negligible, as it is necessary that bits sampled with frequencies of at most $1/2$ have to sample the same value. Thus, in this case, the probability of this happening declines exponentially in the number of frequencies to the left not being at $1 - 1/n$.

For OneMax, the situation is different. The bias in selection only gets strong (that is, increases by a constant additive term) when a constant number of frequencies is left at $1/2$ and has not reached $1 - 1/n$. More general, when ℓ frequencies are still at $1/2$, the bias only adds a term of roughly $1/\sqrt{\ell}$. Thus, it takes longer in expectation in order to detect a significance for a position. However, the bias is constantly there and, even for $\ell = n$, very large when compared to the bias for LeadingOnes for a position whose frequencies to the left are not all at $1 - 1/n$. Hence, for OneMax, the frequencies can be increased in parallel. This is the major difference to LeadingOnes, where the frequencies are increased sequentially.

10.4 Run Time Analysis for the scGA

Being the closest competitor to the sig-cGA in that it also optimizes LeadingOnes in $\mathrm{O}\big(n \log(n)\big)$ in expectation is the scGA, which we introduced in Section 6.4.1. Recall that the scGA has two additional parameters to the cGA: the bias σ, which makes sure that frequencies are more likely to move toward $1/2$, and the confidence border d, which is sufficient in order to set a frequency to $1 - 1/n$ (and $1 - d$ for $1/n$, conversely).

The intention of the scGA is that each frequency stays around $1/2$ as long as there is no strong bias toward either bit value for its respective position. Once the bias is strong enough, the algorithm is willing to fix the bits for that position. While this approach works well when there is a strong bias in a position (as in LeadingOnes; Corollary 6.16), it fails when the bias is only weak (as in OneMax; Theorem 10.9).

We prove that the scGA is not able to optimize OneMax as fast as the sig-cGA, as it is not able to detect the comparably small bias of $1/\sqrt{n}$ for OneMax when compared to the strong bias of $\Theta(1)$ for LeadingOnes for a frequency whose frequencies to the left are at $1 - 1/n$. Note that the assumptions in Theorem 10.9 for ρ and d are the same as the ones made in Corollary 6.16 in order to prove the expected run time of $\mathrm{O}\big(n \log(n)\big)$ of the scGA on LeadingOnes.

▶ **Theorem 10.9.** Let $\alpha, c > 0$ be constants, and let $\beta \in \Omega(1)$ be sufficiently large. Consider the scGA with $K = \alpha/\sigma \geq \beta \ln(n)$, and $1/2 < d \leq 5/6$ being a constant. Its run time on OneMax is at least $2^{c \min\{n, K\}}$ in expectation and with high probability. ◀

Proof. We only show that the run time is at least $2^{c \min\{n, K\}}$ with high probability. The statement for the expected run time follows by lower-bounding the terms that occur with a probability of $\mathrm{o}(1)$ with 0.

We first prove the bound of 2^{cK}. We do so by showing that each frequency stays in the non-empty interval $(1 - d, d) \subset [1/6, 5/6]$ with high probability. Although OneMax introduces a bias into updating a frequency, it is too tiny in order to compensate the strong drift toward $1/2$ in the update. This proof is very similar to the one of Theorem 6.15.

We lower-bound the expected time it takes the scGA to optimize OneMax by upper-bounding the probability it takes a single frequency to leave the interval $(1 - d, d)$. Thus, we condition during the entire proof implicitly on the event that all frequencies are in the interval $(1 - d, d)$. Note that, in this scenario, the probability to sample the optimum during an iteration is at most $(5/6)^n$, which is exponentially small, even for a polynomial number of iterations.

Consider an index $i \in [n]$ with $\boldsymbol{p}_i \in (1 - d, d)$. We only upper-bound the probability it takes $\boldsymbol{p}_i$ to reach d. Note that the probability of $\boldsymbol{p}_i$ reaching $1 - d$ is at most that large, as OneMax introduces a bias for 1s into the selection process. Hence, we could argue optimistically for $\boldsymbol{p}_i$ reaching $1-d$ as we do for $\boldsymbol{p}_i$ reaching d by swapping 1s for 0s and considering $1-\boldsymbol{p}_i$ instead.

Let T denote the first point in time $t \in \mathbf{N}$ such that $\boldsymbol{p}_i^{(t)} \geq d$. We aim at applying Theorem 3.22 and show that it is unlikely for $\boldsymbol{p}_i$ to reach d within 2^{cK} iterations. Hence, we consider the process $(X_t)_{t\in\mathbf{N}}$ with $X_t \coloneqq \big(1-\boldsymbol{p}_i^{(t)}\big)K$. Note that at the beginning, $\boldsymbol{p}_i$ is at $1/2$, that is, $X_0 = K/2$. We stop once $\boldsymbol{p}_i^{(t)} \geq d$, which is equivalent to $X_t \leq (1-d)K$. Thus, we consider an interval of length $\ell \coloneqq K/2-(1-d)K = (d-1/2)K \in \Theta(K)$, as $d > 1/2$ is a constant.

We now argue how an update to $\boldsymbol{p}_i$ is performed in order to estimate its expected value after an update, which is necessary in order to apply Theorem 3.22. Consider, similar to the proof of Theorem 10.8, that the bits of both offspring $\boldsymbol{x}$ and $\boldsymbol{y}$ for all positions but position i have been determined. If the difference of the number of 1s of both offspring without position i is at least 2, then the outcome of neither $\boldsymbol{x}_i$ nor $\boldsymbol{y}_i$ can change the outcome of the selection process. Thus, $\boldsymbol{p}_i$ increases with probability $\boldsymbol{p}_i(1-\boldsymbol{p}_i)$, as the winner offspring needs to sample a 1 and the loser a 0. Analogously, in this case, the probability that $\boldsymbol{p}_i$ decreases is $\boldsymbol{p}_i(1-\boldsymbol{p}_i)$, too.

If the difference of the number of 1s of both offspring without position i is one, then, in order to increase $\boldsymbol{p}_i$, the winner (with respect to all bits but bit i) needs to sample a 1 and the loser a 0, or the winner needs to sample a 0, the loser a 1, and the loser wins. The first case has a probability of $\boldsymbol{p}_i(1-\boldsymbol{p}_i)$, the second of $(1/2)\boldsymbol{p}_i(1-\boldsymbol{p}_i)$, due to the uniform selection when the offspring have equal fitness. In order to decrease $\boldsymbol{p}_i$, the winner needs to sample a 0, the loser a 1, and the winner has to win, which has a probability of $(1/2)\boldsymbol{p}_i(1-\boldsymbol{p}_i)$.

If the difference of the number of 1s of both offspring without position i is zero, then $\boldsymbol{p}_i$ is increased if any offspring samples a 1 and the other samples a 0. This has probability $2\boldsymbol{p}_i(1-\boldsymbol{p}_i)$. In this case, it is not possible that $\boldsymbol{p}_i$ is decreased.

In order to estimate the probabilities of when $\boldsymbol{p}_i$ increases or decreases, we need to estimate the probabilities that the number of 1s of both offspring differ by at least two, differ by exactly one, and differ by exactly zero. Let q_1 denote the probability that this difference is one, and let q_0 denote the probability that the difference is zero. We now bound these probabilities.

Assume that offspring $\boldsymbol{x}$ has k 1s, where $k \in [0..n-1]$, since we assume that bit i has not been sampled yet. For q_0, the offspring $\boldsymbol{y}$ needs to sample k 1s as well, and for q_1, $\boldsymbol{y}$ needs to sample $k-1$ or $k+1$ 1s (such that the result is still in $[0..n-1]$). Due to Lemma 9.2, the probability for $\boldsymbol{y}$ to have this many 1s is $\mathrm{O}\big(1/\sqrt{n}\big)$, as we assume that all frequencies are in the interval $(1-d, d) \subset [1/6, 5/6]$. Hence, by the law of total probability (Theorem 2.1), we get $q_0 \in \mathrm{O}\big(1/\sqrt{n}\big)$ and $q_1 \in \mathrm{O}\big(1/\sqrt{n}\big)$ and, thus, there is a constant $\gamma > 0$ such that $q_0 \leq \gamma/\sqrt{n}$ and $q_1 \leq \gamma/\sqrt{n}$.

We now consider the drift of X in any iteration $t \in \mathbf{N}$ such that $1/2 < \boldsymbol{p}_i^{(t)} < d$, that is, we show that condition (a) of Theorem 3.22 holds. If $\boldsymbol{p}_i^{(t)}$ increases, it changes by $1/K$, and if it decreases, it changes by $1/K + \sigma$.

$$\begin{aligned}
&\mathrm{E}\Big[X_{t+1} - X_t \,\Big|\, \boldsymbol{p}_i^{(t)}\Big] \\
&= K \cdot \mathrm{E}\Big[\boldsymbol{p}_i^{(t)} - \boldsymbol{p}_i^{(t+1)} \,\Big|\, \boldsymbol{p}_i^{(t)}\Big] \\
&= K\Bigg(\bigg(\frac{1}{K} + \sigma\bigg)\bigg((1 - q_0 - q_1)\boldsymbol{p}_i^{(t)}\big(1 - \boldsymbol{p}_i^{(t)}\big) + q_1 \cdot \frac{1}{2}\boldsymbol{p}_i^{(t)}\big(1 - \boldsymbol{p}_i^{(t)}\big)\bigg) \\
&\qquad - \frac{1}{K}\bigg((1 - q_0 - q_1)\boldsymbol{p}_i^{(t)}\big(1 - \boldsymbol{p}_i^{(t)}\big) + q_1 \cdot \frac{3}{2}\boldsymbol{p}_i^{(t)}\big(1 - \boldsymbol{p}_i^{(t)}\big) + q_0 \cdot 2\boldsymbol{p}_i^{(t)}\big(1 - \boldsymbol{p}_i^{(t)}\big)\bigg)\Bigg) \\
&= K\boldsymbol{p}_i^{(t)}\big(1 - \boldsymbol{p}_i^{(t)}\big)\Bigg(\bigg(\frac{1}{K} + \sigma\bigg)\bigg(1 - q_0 - \frac{1}{2}q_1\bigg) - \frac{1}{K}\bigg(1 + q_0 + \frac{1}{2}q_1\bigg)\Bigg) \\
&= K\boldsymbol{p}_i^{(t)}\big(1 - \boldsymbol{p}_i^{(t)}\big)\Bigg(\frac{1}{K}(-2q_0 - q_1) + \sigma\bigg(1 - q_0 - \frac{1}{2}q_1\bigg)\Bigg) .
\end{aligned}$$

For the negative terms with factor σ, by using that $\sigma = \alpha/K$ and by applying the bounds on q_0 and q_1, we get

$$\begin{aligned}
\mathrm{E}\Big[X_{t+1} - X_t \,\Big|\, \boldsymbol{p}_i^{(t)}\Big] &\geq K\boldsymbol{p}_i^{(t)}\big(1 - \boldsymbol{p}_i^{(t)}\big)\Bigg(\frac{\alpha}{K} - \frac{1}{K}\bigg((2 + \alpha)q_0 + \bigg(1 + \frac{\alpha}{2}\bigg)q_1\bigg)\Bigg) \\
&\geq \boldsymbol{p}_i^{(t)}\big(1 - \boldsymbol{p}_i^{(t)}\big)\bigg(\alpha - (3 + 2\alpha)\frac{\gamma}{\sqrt{n}}\bigg) .
\end{aligned}$$

Due to $\alpha \in \Theta(1)$ and $\gamma/\sqrt{n} \in \mathrm{o}(1)$, there is a sufficiently small constant $\beta > 0$ such that $\alpha - (3 + 2\alpha)\gamma/\sqrt{n} \geq \beta$. Thus, we get

$$\begin{aligned}
\mathrm{E}\Big[X_{t+1} - X_t \,\Big|\, \boldsymbol{p}_i^{(t)}\Big] &\geq \beta\boldsymbol{p}_i^{(t)}\big(1 - \boldsymbol{p}_i^{(t)}\big) \\
&\geq \beta \cdot \frac{1}{6} \cdot \frac{5}{6} ,
\end{aligned}$$

which is constant.

We now show that condition (b) of Theorem 3.22 holds. Hence, assume an iteration $t \in \mathbf{N}$ such that $X_t > (1-d)K$, which is equivalent to $\boldsymbol{p}_i^{(t)} < d$. Since the step size of the scGA is independent of the fitness function, we can use the same arguments as in the proof of Theorem 6.15. Thus, we define $r(\ell) = 1 + \alpha$ and $\delta = (1+\alpha)^{1/(1+\alpha)} - 1 > 0$. Note that $1 \leq r(\ell) \in \mathrm{o}\bigl(\ell/\log(\ell)\bigr) = \mathrm{o}\bigl(K/\log(K)\bigr)$ holds, as $K \in \omega(1)$. Since $\boldsymbol{p}_i^{(t)}$ can change by at most $1/K + \sigma$, the value X_t can change by at most $1 + \alpha$. Thus, we only need to bound $\Pr\bigl[|X_{t+1} - X_t| \geq j \bigm| \boldsymbol{p}_i^{(t)}\bigr]$ for $j \in [0..1+\alpha]$. For all of these cases, $r(\ell)/\bigl((1+\delta)^j\bigr) \geq 1$. For $j > 1 + \alpha$, the probability of a change is 0. Thus, condition (b) holds for all $j \in \mathbf{N}$.

Overall, by applying Theorem 3.22 and since $\ell \in \Theta(K)$ and $K \geq \beta \ln(n)$ for a sufficiently large value β, there are constants $c_1, c_2, c_3 > 0$ such that

$$\begin{aligned}\Pr\Bigl[T \leq 2^{\frac{c_1\ell}{r(\ell)}}\Bigr] &= \Pr\bigl[T \leq 2^{cK}\bigr]\\ &\leq 2^{-c_2K}\\ &\leq \frac{1}{n^{c_3+1}}\ .\end{aligned}$$

Thus, with a probability of at most $n^{-(c_3+1)}$, the frequency $\boldsymbol{p}_i$ reaches d within 2^{cK} iterations. Via a union bound (Theorem 2.17), the probability of at least one frequency reaching d within 2^{cK} iterations is at most n^{-c_3}. Consequently, with high probability, no frequency reaches d within 2^{cK} iterations. Since the probability to sample the optimum during any of these iterations is at most $(5/6)^n$, as discussed at the beginning of the proof, the optimum is not sampled within 2^{cK} iterations with subconstant probability if $2^{cK} \in \mathrm{o}\bigl((5/6)^n\bigr)$. Hence, the unconditional probabilities remain asymptotically the same.

If $2^{cK} \in \Omega\bigl((5/6)^n\bigr)$, we choose 2^{cn} as a run time bound instead. This concludes the proof. ■

10.5 Conclusions

We introduced a new algorithm – the sig-cGA – that is able to optimize both OneMax and LeadingOnes in time $\mathrm{O}\bigl(n\log(n)\bigr)$ with high probability and in expectation, which is the first result of this kind for an EDA or even an EA. The sig-cGA achieves these run times by only performing an update to its frequency vector once it notices a significance in its history of samples. In contrast to that, typical theoretically investigated EDAs or EAs do not save the entire history of samples but only a small part thereof: EAs save some samples in their population whereas EDAs store information implicitly in their frequency vector.

Since it is quite memory-consuming to store all samples seen so far the longer the sig-cGA runs, we proposed a way of efficiently saving all of the necessary information for the algorithm, which is the number of 1s or 0s seen so far. Currently, the sig-cGA saves new information every iteration. However, whenever both offspring sample the same value, the algorithm does not learn anything. Thus, an even more memory-efficient approach would be to only save a bit value if the one of the winning offspring differs from the respective bit value of the loser. This is how the cGA actually performs an update. However, since the intention of the sig-cGA is to keep its frequencies as long as possible at $1/2$ until it detects a (hopefully correct) significance, this approach reduces the memory necessary only by a constant factor of 2, due to classical Chernoff bounds.

Overall, the sig-cGA trades slightly increased memory (due to its history) for reduced run times, which is appears to be a very good payoff. In this first work, as often in the theory of evolutionary algorithms, we only regarded the two unimodal benchmark functions OneMax and LeadingOnes. Since it has been observed, for example, recently by Doerr et al. [Doe+17], that insights derived from such analyses can lead to wrong conclusions for more difficult functions, an interesting next step would be to analyze the performance of the sig-cGA on objective functions that have true local optima or that have larger plateaus of equal fitness. Two benchmark functions have been suggested in this context, namely jump functions [DJW02], having an easy to reach local optimum with a scalable basin of attraction, and plateau functions [AD18], having a plateau of scalable diameter around the optimum. We are vaguely optimistic that our sig-cGA has a good performance on these as well. We expect that the sig-cGA, as when optimizing OneMax, quickly fixes a large number of bits to the correct value and then, different from classic EAs, profits from the fact that the missing bits are sampled with uniform distribution, leading to a much more efficient exploration of the small subhypercube formed by these undecided bits. Needless to say, transforming this speculation into a formal proof would be a significant step forward to understanding the sig-cGA.

From a broader perspective, our work shows that by taking into account a longer history and only updating the model when the history justifies it, the performance of a classic EDA can be improved and its usability can be increased (since the difficult choice of the model update strength is now obsolete). An interesting question from this viewpoint would be to what extent similar ideas can be applied to other well-known EDAs.

From a very broad perspective, our work suggests that, generally, EC could profit from enriching the iterative evolutionary process with mechanisms that collect and exploit information over several iterations. So far, such learning-based concepts are rarely used in EC. The only theoretical works in this direction propose a history-based choice of the mutation strength [DDY16a] and analyze hyperheuristics that stick to a chosen subheuristic until its performance over the last τ iterations (τ being a parameter of the algorithms) appears insufficient (see, for example, the work by Doerr et al. [Doe+18] and the references therein).

11 Conclusions & Outlook

In this thesis, we furthered the theoretical understanding of univariate estimation-of-distribution algorithms in two ways. First, we analyzed structural properties of an important subclass, which we called n-Bernoulli-λ-EDAs and which subsumes all commonly theoretically investigated EDAs. We gave different characterizations of when these algorithms are unbiased with respect to the problem encoding, and we proved that the update process of the probabilistic model, given a constant function, cannot stay close to a uniform distribution *(stable)* for a long time while also not changing in expectation *(balanced).*

Second, we analyzed the run times of certain EDAs in noisy and unnoisy scenarios. For the noisy setting, we proved for two algorithms that there exist parameter setups such that the run time of each algorithm scales polynomially in the variance of the noise, which is in contrast to mutation-only EAs, which cannot cope with high noise levels [Fri+17]. For the unnoisy setting, all of our results rely on our prior insight that balanced EDAs are prone to random noise from the sampling process *(genetic drift).* We proved a lower bound for the UMDA on the benchmark function OneMax by showing that either the genetic drift is too strong and the optimization is slowed down by converging to a wrong probabilistic model, or the impact of the genetic drift is reduced by only allowing small updates to the model which, consequently, slows down the optimization time as well. Further, we proposed two new algorithms that are stable for a longer period than the commonly investigated EDAs and, thus, can handle genetic drift better. We analyzed these algorithms each on OneMax and LeadingOnes and showed that one of them optimizes both in $\mathrm{O}\big(n \log(n)\big)$, which is the first result of this kind for EDAs and EAs (see Table 10.1), while the other only does so on LeadingOnes and performs badly on OneMax.

Our results suggest the following key insight: different from mutation-only EAs (which usually have a low mutation rate), the frequency vector of a univariate EDA allows for a great initial exploration phase, due to the frequencies all having constant values. The willingness to change a frequency and, thus, the length of this phase is determined by the step size of the algorithm. During this phase, the algorithm has to pick up the signal from the fitness function in order to optimize it efficiently. However, this process is counteracted by the genetic drift (or other noise). In order to reduce the impact of these wrong signals, the step size can be reduced. This leads to a scenario in which the step size takes two roles, as it determines

1. the willingness to adjust the model and
2. how prone the algorithm is to genetic drift.

These roles are somewhat conflicting, since the step size should not be too small for point 1, but it should not bee too large, due to point 2. This conflict is at the core of all of our run time results, and our analyzes show different methods of how to cope with this behavior. Note that the observation above even holds true for the best algorithm we analyzed – the sig-cGA (Algorithm 2) –, whose parameter ε directly impacts both points. The main difference of the sig-cGA to the other EDAs is that it reduces the impact of the genetic drift by using data from multiple iterations.

We would like to mention that the dependency between both points is more complicated than we sketched, as the results by Lengler et al. [LSW18] suggest a bimodal behavior in the run time of the cGA on OneMax with respect to its step size. The reason for this is that a larger step size (large willingness to adjust the model) also means that a wrong update can be reverted more quickly.

Outlook. Although we went in this thesis into detail about different aspects of EDAs, this research area is still young and there are multiple opportunities for future research that we discuss briefly.

One important area is to expand the run time analysis to more functions than OneMax and LeadingOnes, especially in order to better compare EDAs with EAs, for which far more functions have been analyzed. With respect to upper bounds, canonical candidates are linear functions or the Jump function [DJW02]. Recently, a first step toward this direction has been taken with a result by Hasenöhrl and Sutton [HS18], who analyzed the run time of the cGA on Jump.

Regarding lower run time bounds, mainly OneMax has been analyzed so far (see Table 10.1). Analyzing other functions such as LeadingOnes or BinVal (in the hope that parts from the proofs for OneMax carry over) would be very insightful, especially in order to see if the sig-cGA remains the only EDA with an $\mathrm{O}\big(n \log(n)\big)$ run time on OneMax that is also able to optimize LeadingOnes and BinVal in $O(n \log(n))$. For example, Witt [Wit18] recently showed that the cGA (without a margin) has a run time of $\Omega(n^2)$ on BinVal. Similarly, a general lower bound for unbiased EDAs – at least for the subclass of ρ-bounded locally updating n-Bernoulli-λ-EDAs – for OneMax or LeadingOnes would be helpful and result in a more rigorous comparison of EDAs and EAs with univariate variation operators, which have an unbiased black-box complexity of $\Omega(n \log(n))$ [LW12].

Tying in with the last point, combining results for the cGA and the 2-MMAS$_{\text{IB}}$ (which are both ρ-bounded and locally updating) would be a major next step in the theory of EDAs, as these two algorithms seem to behave very similarly (see Table 10.1). However, oftentimes only the cGA is analyzed, since its frequencies only take finitely many values. This begs the question whether there is a difference in these algorithms or whether their results can be translated into one another. An extension to this problem would be to combine the results of all EDAs mentioned in Section 4.3. For example, the results by Lengler et al. [LSW18] suggest a bimodal run time behavior of the cGA on OneMax, where this suggestion is based on assuming that run time results of the UMDA (namely, the results by Witt [Wit17]) carry over to the cGA.

Further, our results from Chapters 7 and 8 suggest that EDAs are well suited to cope with noise. However, more results are needed in order to make such a general claim. Potential candidates for future research are the UMDA or the sig-cGA.

Last, all theoretical results so far only consider univariate EDAs. Expanding the analysis to multivariate EDAs, which are designed to model dependencies, seems challenging but would greatly advance the theoretical field of EDAs. A starting point could be to analyze multivariate EDAs that model linear dependencies on LeadingOnes, such as the algorithm MIMIC by Bonet et al. [BJV96].

Bibliography

[AAT15] Youhei Akimoto, Sandra Astete-Morales, and Olivier Teytaud. **Analysis of runtime of optimization algorithms for noisy functions over discrete codomains**. *Theoretical Computer Science* 605 (2015), 42–50. DOI: 10.1016/j.tcs.2015.04.008 (see page 131).

[AD18] Denis Antipov and Benjamin Doerr. **Precise runtime analysis for plateaus**. In: *Proceedings of the 15th International Conference on Parallel Problem Solving from Nature (PPSN XV)*. 2018, 117–128. DOI: 10.1007/978-3-319-99259-4_10 (see page 219).

[Afs+13] Peyman Afshani, Manindra Agrawal, Benjamin Doerr, Carola Doerr, Kasper Green Larsen, and Kurt Mehlhorn. **The query complexity of finding a hidden permutation**. In: *Space-Efficient Data Structures, Streams, and Algorithms*. Springer-Verlag Berlin Heidelberg, 2013, 1–11. DOI: 10.1007/978-3-642-40273-9_1 (see pages 13, 198).

[Arb+16] Dídac Rodríguez Arbonès, Boyin Ding, Nataliia Y. Sergiienko, and Markus Wagner. **Fast and effective multi-objective optimisation of submerged wave energy converters**. In: *Proceedings of the 14th International Conference on Parallel Problem Solving from Nature (PPSN XIV)*. 2016, 675–685. DOI: 10.1007/978-3-319-45823-6_63 (see page 2).

[AS65] Milton Abramowitz and Irena A. Stegun. **Handbook of mathematical functions with formulas, graphs, and mathematical tables**. Dover Publications, 1965. ISBN: 978-0-486-61272-0 (see pages 120, 206).

[AW09] Gautham Anil and R. Paul Wiegand. **Black-box search by elimination of fitness functions**. In: *Proceedings of the 10th ACM/SIGEVO Workshop on Foundations of Genetic Algorithms (FOGA X)*. 2009, 67–78. DOI: 10.1145/1527125.1527135 (see pages 70, 198).

[Bal94] Shumeet Baluja. **Population-Based Incremental Learning: a method for integrating genetic search based function optimization and competitive learning**. Tech. rep. CMU-CS-94-163. Pittsburgh, PA, USA: Carnegie Mellon University, 1994 (see page 64).

[BE65] Bengt von Bahr and Carl-Gustav Esseen. **Inequalities for the rth absolute moment of a sum of random variables, $1 \leqq r \leqq 2$**. *The Annals of Mathematical Statistics* 36:1 (1965), 299–303. DOI: 10.1214/aoms/1177700291 (see page 117).

[Bey00] Hans-Georg Beyer. **Evolutionary algorithms in noisy environments: theoretical issues and guidelines for practice**. *Computer Methods in Applied Mechanics and Engineering* 186:2–4 (2000), 239–267. DOI: 10.1016/S0045-7825(99)00386-2 (see page 126).

[Bia+09] Leonora Bianchi, Marco Dorigo, Luca Maria Gambardella, and Walter J. Gutjahr. **A survey on metaheuristics for stochastic combinatorial optimization**. *Natural Computing* 8:2 (2009), 239–287. DOI: 10.1007/s11047-008-9098-4 (see page 113).

[Bil95] Patrick Billingsley. **Probability and measure**. 3rd ed. John Wiley & Sons, Inc., 1995. ISBN: 978-81-265-1771-8 (see page 168).

[BJV96] Jeremy S. De Bonet, Charles L. Isbell Jr., and Paul A. Viola. **MIMIC: finding optima by estimating probability densities**. In: *Proceedings of the 9th International Conference on Neural Information Processing Systems (NIPS'96)*. 1996, 424–430. URL: http://papers.nips.cc/paper/1328-mimic-finding-optima-by-estimating-probability-densities (see page 223).

[BLS14] Golnaz Badkobeh, Per Kristian Lehre, and Dirk Sudholt. **Unbiased black-box complexity of parallel search**. In: *Proceedings of the 13th International Conference on Parallel Problem Solving from Nature (PPSN XIII)*. 2014, 892–901. DOI: 10.1007/978-3-319-10762-2_88 (see page 71).

[BOS02] Hans-Georg Beyer, Markus Olhofer, and Bernhard Sendhoff. **On the behavior of $(\mu/\mu_1, \lambda)$-ES optimizing functions disturbed by generalized noise**. In: *Proceedings of the 7th ACM/SIGEVO Workshop on Foundations of Genetic Algorithms (FOGA VII)*. 2002, 307–328 (see page 114).

[BP83] Philip J. Boland and Frank Proschan. **The reliability of K out of N systems**. *The Annals of Probability* 11:3 (1983), 760–764. DOI: 10.1214/aop/1176993520 (see page 145).

[BQT18] Chao Bian, Chao Qian, and Ke Tang. **Towards a running time analysis of the (1+1)-EA for OneMax and LeadingOnes under general bit-wise noise**. In: *Proceedings of the 15th International Conference on Parallel Problem Solving from Nature (PPSN XV)*. 2018, 165–177. DOI: 10.1007/978-3-319-99259-4_14 (see page 133).

[BS07] Hans-Georg Beyer and Bernhard Sendhoff. **Robust optimization – a comprehensive survey**. *Computer Methods in Applied Mechanics and Engineering* 196:33–34 (2007), 3190–3218. DOI: 10.1016/j.cma.2007.03.003 (see page 114).

[Bul15] Peter Bullen. **Dictionary of inequalities**. 2nd ed. Taylor & Francis Group, 2015. ISBN: 978-1-4822-3761-0 (see page 122).

[BW16] Rabi Bhattacharya and Edward C. Waymire. **A basic course in probability theory**. 2nd ed. Springer International Publishing, 2016. ISBN: 978-3-319-47972-9 (see page 32).

[CCM11] Seok-Ho Chang, Pamela C. Cosman, and Laurence B. Milstein. **Chernoff-type bounds for the Gaussian error function**. *IEEE Transactions on Communications* 59:11 (2011), 2939–2944. DOI: 10.1109/TCOMM.2011.072011.100049 (see page 116).

[CDG08] Gianni A. Di Caro, Frederick Ducatelle, and Luca M. Gambardella. **Theory and practice of ant-based routing in dynamic telecommunications networks**. In: *Reflexing Interfaces: The Complex Coevolution of Information Technology Ecosystems*. 2008, 185–216. DOI: 10.4018/978-1-59904-627-3.ch012 (see page 133).

[Che+07] Tianshi Chen, Ke Tang, Guoliang Chen, and Xin Yao. **On the analysis of average time complexity of estimation of distribution algorithms**. In: *Proceedings of the IEEE Congress on Evolutionary Computation (CEC'07)*. 2007, 453–460. DOI: 10.1109/CEC.2007.4424506 (see page 149).

[Che+09a] Tianshi Chen, Per Kristian Lehre, Ke Tang, and Xin Yao. **When is an estimation of distribution algorithm better than an evolutionary algorithm?** In: *Proceedings of the IEEE Congress on Evolutionary Computation (CEC'09)*. 2009, 1470–1477. DOI: 10.1109/CEC.2009.4983116 (see pages 65, 149).

[Che+09b] Tianshi Chen, Ke Tang, Guoliang Chen, and Xin Yao. **Rigorous time complexity analysis of Univariate Marginal Distribution Algorithm with margins**. In: *Proceedings of the IEEE Congress on Evolutionary Computation (CEC'09)*. 2009, 2157–2164. DOI: 10.1109/CEC.2009.4983208 (see pages 65, 149).

[Che+10] Tianshi Chen, Ke Tang, Guoliang Chen, and Xin Yao. **Analysis of computational time of simple estimation of distribution algorithms**. *IEEE Transactions on Evolutionary Computations* 14:1 (2010), 1–22. DOI: 10.1109/TEVC.2009.2040019 (see pages 65, 149).

[Chr76] Nicos Christofides. **Worst-case analysis of a new heuristic for the Travelling Salesman Problem**. Tech. rep. 388. Pittsburgh, PA, USA: Carnegie Mellon University, 1976 (see page 1).

[Cor+09] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. **Introduction to algorithms**. 3rd ed. The MIT Press, 2009. ISBN: 978-0-262-03384-8 (see pages 11, 172, 187).

[Cor+14] Dogan Corus, Duc-Cuong Dang, Anton V. Eremeev, and Per Kristian Lehre. **Level-based analysis of genetic algorithms and other search processes**. *Computing Research Repository (arXiv)* (2014). URL: http://arxiv.org/abs/1407.7663 (see pages 30, 34).

[DD18] Benjamin Doerr and Carola Doerr. **Optimal static and self-adjusting parameter choices for the (1+(λ, λ)) genetic algorithm**. *Algorithmica* 80:5 (2018), 1658–1709. DOI: 10.1007/s00453-017-0354-9 (see page 197).

[DDK14] Benjamin Doerr, Carola Doerr, and Timo Kötzing. **The unbiased black-box complexity of partition is polynomial**. *Artificial Intelligence* 216 (2014), 275–286. DOI: 10.1016/j.artint.2014.07.009 (see page 71).

[DDK15] Benjamin Doerr, Carola Doerr, and Timo Kötzing. **Unbiased black-box complexities of Jump functions**. *Evolutionary Computation* 23:4 (2015), 641–670. DOI: 10.1162/EVCO_a_00158 (see page 71).

[DDY16a] Benjamin Doerr, Carola Doerr, and Jing Yang. **k-bit mutation with self-adjusting k outperforms standard bit mutation**. In: *Proceedings of the 14th International Conference on Parallel Problem Solving from Nature (PPSN XIV)*. 2016, 824–834. DOI: 10.1007/978-3-319-45823-6_77 (see page 220).

[DDY16b] Benjamin Doerr, Carola Doerr, and Jing Yang. **Optimal parameter choices via precise black-box analysis**. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'16)*. 2016, 1123–1130. DOI: 10.1145/2908812.2908950 (see page 71).

[DG13] Benjamin Doerr and Leslie A. Goldberg. **Adaptive drift analysis**. *Algorithmica* 65:1 (2013), 224–250. DOI: 10.1007/s00453-011-9585-3 (see pages 31, 49, 103, 117).

[DHK12] Benjamin Doerr, Ashish Ranjan Hota, and Timo Kötzing. **Ants easily solve stochastic shortest path problems**. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'12)*. 2012, 17–24. DOI: 10.1145/2330163.2330167 (see pages 114, 134).

[DJW02] Stefan Droste, Thomas Jansen, and Ingo Wegener. **On the analysis of the (1+1) evolutionary algorithm**. *Theoretical Computer Science* 276:1–2 (2002), 51–81. DOI: 10.1016/S0304-3975(01)00182-7 (see pages 70, 134, 197, 210, 219, 222).

[DJW06] Stefan Droste, Thomas Jansen, and Ingo Wegener. **Upper and lower bounds for randomized search heuristics in black-box optimization**. *Theory of Computing Systems* 39:4 (2006), 525–544. DOI: 10.1007/s00224-004-1177-z (see pages 69, 70, 150, 198).

[DJW12] Benjamin Doerr, Daniel Johannsen, and Carola Winzen. **Multiplicative drift analysis**. *Algorithmica* 64:4 (2012), 673–697. DOI: 10.1007/s00453-012-9622-x (see pages 30, 49).

[DK15] Benjamin Doerr and Marvin Künnemann. **Optimizing linear functions with the $(1 + \lambda)$ evolutionary algorithm—different asymptotic runtimes for different instances**. *Theoretical Computer Science* 561 (2015), 3–23. DOI: 10.1016/j.tcs.2014.03.015 (see page 197).

[DK18a] Benjamin Doerr and Martin S. Krejca. **Significance-based estimation-of-distribution algorithms**. *Computing Research Repository (arXiv)* (2018). URL: http://arxiv.org/abs/1807.03495 (see pages 93, 195, 197).

[DK18b] Benjamin Doerr and Martin S. Krejca. **Significance-based estimation-of-distribution algorithms**. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'18)*. 2018, 1483–1490. DOI: 10.1145/3205455.3205553 (see pages 93, 149, 195).

[DKW11] Benjamin Doerr, Timo Kötzing, and Carola Winzen. **Too fast unbiased black-box algorithms**. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'11)*. 2011, 2043–2050. DOI: 10.1145/2001576.2001851 (see page 75).

[DL15] Duc-Cuong Dang and Per Kristian Lehre. **Simplified runtime analysis of estimation of distribution algorithms**. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'15)*. 2015, 513–518. DOI: 10.1145/2739480.2754814 (see pages 60, 65, 149, 150, 197).

[DL16] Duc-Cuong Dang and Per Kristian Lehre. **Runtime analysis of non-elitist populations: from classical optimisation to partial information**. *Algorithmica* 75:3 (2016), 428–461. DOI: 10.1007/s00453-015-0103-x (see page 133).

[DL17] Carola Doerr and Johannes Lengler. **OneMax in black-box models with several restrictions**. *Algorithmica* 78:2 (2017), 610–640. DOI: 10.1007/s00453-016-0168-1 (see page 71).

[Doe+11a] Benjamin Doerr, Daniel Johannsen, Timo Kötzing, Per Kristian Lehre, Markus Wagner, and Carola Winzen. **Faster black-box algorithms through higher arity operators**. In: *Proceedings of the 11th ACM/SIGEVO Workshop on Foundations of Genetic Algorithms (FOGA XI)*. 2011, 163–172. DOI: 10.1145/1967654.1967669 (see pages 71, 76).

[Doe+11b] Benjamin Doerr, Frank Neumann, Dirk Sudholt, and Carsten Witt. **Runtime analysis of the 1-ANT Ant Colony optimizer**. *Theoretical Computer Science* 412:17 (2011), 1629–1644. DOI: 10.1016/j.tcs.2010.12.030 (see page 197).

[Doe+13] Benjamin Doerr, Thomas Jansen, Dirk Sudholt, Carola Winzen, and Christine Zarges. **Mutation rate matters even when optimizing monotonic functions**. *Evolutionary Computation* 21:1 (2013), 1–27. DOI: 10.1162/EVCO_a_00055 (see page 64).

[Doe+17] Benjamin Doerr, Huu Phuoc Le, Régis Makhmara, and Ta Duy Nguyen. **Fast genetic algorithms**. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'17)*. 2017, 777–784. DOI: 10.1145/3071178.3071301 (see page 219).

[Doe+18] Benjamin Doerr, Andrei Lissovoi, Pietro S. Oliveto, and John Alasdair Warwicker. **On the runtime analysis of selection hyper-heuristics with adaptive learning periods**. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'18)*. 2018, 1015–1022. DOI: 10.1145/3205455.3205611 (see page 220).

[Doe18] Benjamin Doerr. **Probabilistic tools for the analysis of randomized optimization heuristics**. *Computing Research Repository (arXiv)* (2018). URL: https://arxiv.org/abs/1801.06733 (see page 198).

[Dro04] Stefan Droste. **Analysis of the (1+1) EA for a noisy OneMax**. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'04)*. 2004, 1088–1099. DOI: 10.1007/978-3-540-24854-5_107 (see page 114).

[Dro06] Stefan Droste. **A rigorous analysis of the Compact Genetic Algorithm for linear functions**. *Natural Computing* 5:3 (2006), 257–283. DOI: 10.1007/s11047-006-9001-0 (see pages 68, 118, 149, 196, 210).

[DS04] Marco Dorigo and Thomas Stützle. **Ant Colony Optimization**. 1st ed. The MIT Press, 2004. ISBN: 978-0-262-04219-2 (see page 133).

[Dur19] Rick Durrett. **Probability: theory and examples**. Cambridge University Press, 2019. ISBN: 978-1-108-47368-2 (see pages 39, 51).

[DW14a] Benjamin Doerr and Carola Winzen. **Playing Mastermind with constant-size memory**. *Theory of Computing Systems* 55:4 (2014), 658–684. DOI: 10.1007/s00224-012-9438-8 (see page 71).

[DW14b] Benjamin Doerr and Carola Winzen. **Ranking-based black-box complexity**. *Algorithmica* 68:3 (2014), 571–609. DOI: 10.1007/s00453-012-9684-9 (see pages 71, 210).

[DW14c] Benjamin Doerr and Carola Winzen. **Reducing the arity in unbiased black-box complexity**. *Theoretical Computer Science* 545 (2014), 108–121. DOI: 10.1016/j.tcs.2013.05.004 (see page 71).

[ER63] Paul Erdős and Alfréd Rényi. **On two problems of information theory**. *Magyar Tud. Akad. Mat. Kutató Int. Kőzl* 8 (1963), 229–243 (see page 70).

[Fel68] William Feller. **An introduction to probability theory and its applications**. 3rd ed. Vol. 1. John Wiley & Sons, Inc., 1968. ISBN: 978-0-471-25708-0 (see page 186).

[Fel71] William Feller. **An introduction to probability theory and its applications**. 2nd ed. Vol. 2. John Wiley & Sons, Inc., 1971. ISBN: 978-0-471-25709-7 (see page 168).

[FK13] Matthias Feldmann and Timo Kötzing. **Optimizing expected path lengths with Ant Colony Optimization using fitness proportional update**. In: *Proceedings of the 12th ACM/SIGEVO Workshop on Foundations of Genetic Algorithms (FOGA XII)*. 2013, 65–74. DOI: 10.1145/2460239.2460246 (see pages 114, 134).

[FKK16] Tobias Friedrich, Timo Kötzing, and Martin S. Krejca. **EDAs cannot be balanced and stable**. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'16)*. 2016, 1139–1146. DOI: 10.1145/2908812.2908895 (see pages 59, 93, 149).

[FKK18] Tobias Friedrich, Timo Kötzing, and Martin S. Krejca. **Unbiasedness of estimation-of-distribution algorithms**. *Theoretical Computer Science* (2018). DOI: 10.1016/j.tcs.2018.11.001 (see pages 59, 69).

[Fri+16] Tobias Friedrich, Timo Kötzing, Martin S. Krejca, and Andrew M. Sutton. **Robustness of Ant Colony Optimization to noise**. *Evolutionary Computation* 24:2 (2016), 237–254. DOI: 10.1162/EVCO_a_00178 (see pages 113, 133).

[Fri+17] 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 Computations* 21:3 (2017), 477–490. DOI: 10.1109/TEVC.2016.2613739 (see pages 9, 113–115, 130, 131, 221).

[GK16] Christian Gießen and Timo Kötzing. **Robustness of populations in stochastic environments**. *Algorithmica* 75 (2016), 462–489. DOI: 10.1007/s00453-015-0072-0 (see pages 114, 133).

[Gol+17] Daniel Golovin, Benjamin Solnik, Subhodeep Moitra, Greg Kochanski, John Karro, and David Sculley. **Google Vizier: a service for black-box optimization**. In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'17)*. 2017, 1487–1495. DOI: 10.1145/3097983.3098043 (see page 1).

[GP96] Walter J. Gutjahr and Georg Ch. Pflug. **Simulated annealing for noisy cost functions**. *Journal of Global Optimization* 8:1 (1996), 1–13. DOI: 10.1007/BF00229298 (see page 114).

[GR07] Izrail S. Gradshteyn and Iosif M. Ryzhik. **Table Of integrals, series, and products**. 7th ed. Elsevier Science Publishers, 2007. ISBN: 978-0-12-373637-6 (see page 119).

[GS01a] Geoffrey R. Grimmett and David R. Stirzaker. **One thousand exercises in probability**. 2nd ed. Oxford University Press, 2001. ISBN: 978-0-19-857221-3 (see page 22).

[GS01b] Geoffrey R. Grimmett and David R. Stirzaker. **Probability and random processes**. 3rd ed. Oxford University Press, 2001. ISBN: 978-0-19-857222-0 (see pages 14, 15, 18, 20–22, 27, 32, 33, 97).

[Had75] F. Hadlock. **Finding a maximum cut of a planar graph in polynomial time**. *SIAM Journal on Computing* 4 (1975), 221–225. DOI: 10.1137/0204019 (see page 1).

[Haj82] Bruce Hajek. **Hitting-time and occupation-time bounds implied by drift analysis with applications**. *Advances in Applied Probability* 14:3 (1982), 502–525. DOI: 10.2307/1426671 (see page 29).

[HLG99] Georges R. Harik., Fernando G. Lobo, and David E. Goldberg. **The Compact Genetic Algorithm**. *IEEE Transactions on Evolutionary Computations* 3:4 (1999), 287–297. DOI: 10.1109/4235.797971 (see pages 5, 67).

[HP11] Mark Hauschild and Martin Pelikan. **An introduction and survey of estimation of distribution algorithms**. *Swarm and Evolutionary Computation* 1:3 (2011), 111–128. DOI: 10.1016/j.swevo.2011.08.003 (see pages 2, 4).

[HR97] Markus Höhfeld and Günter Rudolph. **Towards a theory of Population-Based Incremental Learning**. In: *Proceedings of the IEEE Congress on Evolutionary Computation (CEC'97)*. 1997, 1–5 (see page 64).

[HS18] Václav Hasenöhrl and Andrew M. Sutton. **On the runtime dynamics of the Compact Genetic Algorithm on Jump functions**. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'18)*. 2018, 967–974. DOI: 10.1145/3205455.3205608 (see page 222).

[HY01] Jun He and Xin Yao. **Drift analysis and average time complexity of evolutionary algorithms**. *Artificial Intelligence* 127:1 (2001), 57–85. DOI: 10.1016/S0004-3702(01)00058-3 (see pages 29, 36).

[HY04] Jun He and Xin Yao. **A study of drift analysis for estimating computation time of evolutionary algorithms**. *Natural Computing* 3:1 (2004), 21–35. DOI: 10.1023/B:NACO.0000023417.31393.c7 (see pages 29, 36).

[Jan13] Thomas Jansen. **Analyzing evolutionary algorithms: the computer science perspective**. Natural Computing Series. Springer-Verlag Berlin Heidelberg, 2013. ISBN: 978-3-642-17338-7. DOI: 10.1007/978-3-642-17339-4 (see page 64).

[JB05] Yaochu Jin and Jürgen Branke. **Evolutionary optimization in uncertain environments—a survey**. *IEEE Transactions on Evolutionary Computations* 9 (2005), 303–317. DOI: 10.1109/TEVC.2005.846356 (see page 113).

[JJW05] Thomas Jansen, Kenneth A. De Jong, and Ingo Wegener. **On the choice of the offspring population size in evolutionary algorithms**. *Evolutionary Computation* 13 (2005), 413–440. DOI: 10.1162/106365605774666921 (see page 197).

[Joh10] Daniel Johannsen. **Random combinatorial structures and randomized search heuristics**. PhD thesis. Universität des Saarlandes, 2010. URL: https://publikationen.sulb.uni-saarland.de/handle/20.500.11880/26072 (see pages 30, 45).

[JZ14] Thomas Jansen and Christine Zarges. **Performance analysis of randomised search heuristics operating with a fixed budget**. *Theoretical Computer Science* 545 (2014), 39–58. DOI: 10.1016/j.tcs.2013.06.007 (see page 131).

[KF09] Daphne Koller and Nir Friedman. **Probabilistic graphical models: principles and techniques**. The MIT Press, 2009. ISBN: 978-0-262-01319-2 (see page 2).

[KGV83] Scott Kirkpatrick, C. D. Gelatt, and Mario P. Vecchi. **Optimization by Simulated Annealing**. *Science* 220:4598 (1983), 671–680. DOI: 10.1126/science.220.4598.671 (see page 1).

[KK18] Timo Kötzing and Martin S. Krejca. **First-hitting times under additive drift**. In: *Proceedings of the 15th International Conference on Parallel Problem Solving from Nature (PPSN XV)*. 2018, 92–104. DOI: 10.1007/978-3-319-99259-4_8 (see pages 29, 39).

[KM12] Timo Kötzing and Hendrik Molter. **ACO Beats EA on a dynamic pseudo-Boolean function**. In: *Proceedings of the 12th International Conference on Parallel Problem Solving from Nature (PPSN XII)*. 2012, 113–122. DOI: 10.1007/978-3-642-32937-1_12 (see page 134).

[Köt+11] Timo Kötzing, Frank Neumann, Dirk Sudholt, and Markus Wagner. **Simple Max-Min Ant Systems and the optimization of linear pseudo-Boolean functions**. In: *Proceedings of the 11th ACM/SIGEVO Workshop on Foundations of Genetic Algorithms (FOGA XI)*. 2011, 209–218. DOI: 10.1145/1967654.1967673 (see page 134).

[Köt16] Timo Kötzing. **Concentration of first hitting times under additive drift**. *Algorithmica* 75:3 (2016), 490–506. DOI: 10.1007/s00453-015-0048-0 (see pages 31, 55).

[KP15] Janusz Kacprzyk and Witold Pedrycz, eds. **Springer handbook of computational intelligence**. Springer-Verlag Berlin Heidelberg, 2015. DOI: 10.1007/978-3-662-43505-2 (see pages 2, 236).

[KW18a] Martin S. Krejca and Carsten Witt. **Lower bounds on the run time of the Univariate Marginal Distribution Algorithm on OneMax**. *Theoretical Computer Science* (2018). DOI: 10.1016/j.tcs.2018.06.004 (see pages 65, 149).

[KW18b] Martin S. Krejca and Carsten Witt. **Theory of estimation-of-distribution algorithms**. *Computing Research Repository (arXiv)* (2018). URL: http://arxiv.org/abs/1806.05392 (see page 4).

[Len17] Johannes Lengler. **Drift analysis**. *Computing Research Repository (arXiv)* (2017). URL: http://arxiv.org/abs/1712.00964 (see pages 29, 30, 34, 45).

[LL02] Pedro Larrañaga and Jose A. Lozano. **Estimation of distribution algorithms: a new tool for evolutionary computation**. Springer US, 2002. DOI: 10.1007/978-1-4615-1539-5 (see page 2).

[LN17] Per Kristian Lehre and Phan Trung Hai Nguyen. **Improved runtime bounds for the Univariate Marginal Distribution Algorithm via anti-concentration**. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'17)*. 2017, 1383–1390. DOI: 10.1145/3071178.3071317 (see pages 6, 10, 65, 149, 150, 192, 197).

[LN18] Per Kristian Lehre and Phan Trung Hai Nguyen. **Level-based analysis of the Population-Based Incremental Learning algorithm**. In: *Proceedings of the 15th International Conference on Parallel Problem Solving from Nature (PPSN XV)*. 2018, 105–116. DOI: 10.1007/978-3-319-99259-4_9 (see pages 64, 149, 197).

[LS18] Johannes Lengler and Angelika Steger. **Drift analysis and evolutionary algorithms revisited**. *Combinatorics, Probability and Computing* 27:4 (2018), 643–666. DOI: 10.1017/S0963548318000275 (see page 49).

[LSW18] Johannes Lengler, Dirk Sudholt, and Carsten Witt. **Medium step sizes are harmful for the Compact Genetic Algorithm**. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'18)*. 2018, 1499–1506. DOI: 10.1145/3205455.3205576 (see pages 7, 68, 94, 103, 149, 195, 222, 223).

[LW12] Per Kristian Lehre and Carsten Witt. **Black-box search by unbiased variation**. *Algorithmica* 64:4 (2012), 623–642. DOI: 10.1007/s00453-012-9616-8 (see pages 8, 70, 71, 75, 204, 223).

[LW13] Per Kristian Lehre and Carsten Witt. **General drift analysis with tail bounds**. *Computing Research Repository (arXiv)* abs/1307.2559 (2013). URL: http://arxiv.org/abs/1307.2559 (see page 49).

[LW14] Per Kristian Lehre and Carsten Witt. **Concentrated hitting times of randomized search heuristics with variable drift**. In: *Proceedings of the 25th International Symposium on Algorithms and Computation (ISAAC'14)*. 2014, 686–697. DOI: 10.1007/978-3-319-13075-0_54 (see page 30).

[LW16] Andrei Lissovoi and Carsten Witt. **MMAS versus population-based EA on a family of dynamic fitness functions**. *Algorithmica* 75:3 (2016), 554–576. DOI: 10.1007/s00453-015-9975-z (see page 134).

[Mit64] Dragoslav S. Mitrinović. **Elementary inequalities**. P. Noordhoff Ltd., 1964 (see page 119).

[MP96] Heinz Mühlenbein and Gerhard Paaß. **From recombination of genes to the estimation of distributions I. binary parameters**. In: *Proceedings of the 4th International Conference on Parallel Problem Solving from Nature (PPSN IV)*. 1996, 178–187. DOI: 10.1007/3-540-61723-X_982 (see pages 5, 65).

[MR95] Rajeev Motwani and Prabhakar Raghavan. **Randomized algorithms**. Cambridge University Press, 1995. ISBN: 978-0-521-47465-8 (see pages 27, 35).

[MRC09] Boris Mitavskiy, Jonathan E. Rowe, and Chris Cannings. **Theoretical analysis of local search strategies to optimize network communication subject to preserving the total number of links**. *International Journal of Intelligent Computing and Cybernetics* 2:2 (2009), 243–284. DOI: 10.1108/17563780910959893 (see page 45).

[MS17] Alberto Moraglio and Dirk Sudholt. **Principled design and runtime analysis of abstract convex evolutionary search**. *Evolutionary Computation* 25:2 (2017), 205–236. DOI: 10.1162/EVCO_a_00169 (see page 197).

[MU05] Michael Mitzenmacher and Eli Upfal. **Probability and computing: randomized algorithms and probabilistic analysis**. 1st ed. Cambridge University Press, 2005. ISBN: 978-0-521-83540-4 (see pages 11, 13, 18, 21, 26, 50, 54, 182, 184).

[Müh92] Heinz Mühlenbein. **How genetic algorithms really work: mutation and hillclimbing**. In: *Proceedings of the 2nd International Conference on Parallel Problem Solving from Nature (PPSN II)*. 1992, 15–26 (see pages 4, 12, 149).

[NSW09] Frank Neumann, Dirk Sudholt, and Carsten Witt. **Analysis of different MMAS ACO algorithms on unimodal functions and plateaus**. *Swarm Intelligence* 3:1 (2009), 35–68. DOI: 10.1007/s11721-008-0023-3 (see page 197).

[NSW10] Frank Neumann, Dirk Sudholt, and Carsten Witt. **A few ants are enough: ACO with iteration-best update**. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'10)*. 2010, 63–70. DOI: 10.1145/1830483.1830493 (see pages 65, 66, 149).

[NW09] Frank Neumann and Carsten Witt. **Runtime analysis of a simple Ant Colony Optimization algorithm**. *Algorithmica* 54:2 (2009), 243–255. DOI: 10.1007/s00453-007-9134-2 (see page 197).

[Oll+17] Yann Ollivier, Ludovic Arnold, Anne Auger, and Nikolaus Hansen. **Information-geometric optimization algorithms: a unifying picture via invariance principles**. *Journal of Machine Learning Research* 18 (2017), 18:1–18:65. URL: http://jmlr.org/papers/v18/14-467.html (see page 60).

[OW11] Pietro Simone Oliveto and Carsten Witt. **Simplified drift analysis for proving lower bounds in evolutionary computation**. *Algorithmica* 59:3 (2011), 369–386. DOI: 10.1007/s00453-010-9387-z (see pages 31, 54).

[OW12] Pietro Simone Oliveto and Carsten Witt. **Erratum: simplified drift analysis for proving lower bounds in evolutionary computation**. *Computing Research Repository (arXiv)* (2012). URL: http://arxiv.org/abs/1211.7184 (see page 54).

[OW15] Pietro Simone Oliveto and Carsten Witt. **Improved time complexity analysis of the Simple Genetic Algorithm**. *Theoretical Computer Science* 605 (2015), 21–41. DOI: 10.1016/j.tcs.2015.01.002 (see page 176).

[Pai+15] Tiago Paixão, Golnaz Badkobeh, Nick Barton, Doğan Çörüş, Duc-Cuong Dang, Tobias Friedrich, Per Kristian Lehre, Dirk Sudholt, Andrew M. Sutton, and Barbora Trubenová. **Toward a unifying framework for evolutionary processes**. *Journal of Theoretical Biology* 383 (2015), 28–43. DOI: 10.1016/j.jtbi.2015.07.011 (see page 60).

[PHL15] Martin Pelikan, Mark Hauschild, and Fernando G. Lobo. **Estimation of distribution algorithms**. In: *Springer Handbook of Computational Intelligence [KP15]*. Springer-Verlag Berlin Heidelberg, 2015, 899–928. DOI: 10.1007/978-3-662-43505-2_45 (see page 2).

[PRS15] Adam Prügel-Bennett, Jonathan Rowe, and Jonathan Shapiro. **Run-time analysis of Population-Based Evolutionary Algorithm in noisy environments**. In: *Proceedings of the 13th ACM/SIGEVO Workshop on Foundations of Genetic Algorithms (FOGA XIII)*. 2015, 69–75. DOI: 10.1145/2725494.2725498 (see page 132).

[PSC06] Martin Pelikan, Kumara Sastry, and Erick Cantú-Paz. **Scalable optimization via probabilistic modeling: from algorithms to applications**. Vol. 33. Studies in Computational Intelligence. Springer-Verlag Berlin Heidelberg, 2006. DOI: 10.1007/978-3-540-34954-9 (see page 2).

[RS14] Jonathan E. Rowe and Dirk Sudholt. **The choice of the offspring population size in the (1,λ) evolutionary algorithm**. *Theoretical Computer Science* 545 (2014), 20–38. DOI: 10.1016/j.tcs.2013.09.036 (see pages 45, 46).

[Rud76] Walter Rudin. **Principles of mathematical analysis**. 3rd ed. McGraw-Hill Education, 1976. ISBN: 978-0-07-085613-4 (see page 46).

[Rud97] Günter Rudolph. **Convergence properties of evolutionary algorithms**. Verlag Dr. Kovač, 1997. ISBN: 978-3-86064-554-3 (see page 12).

[RV11] Jonathan E. Rowe and Michael D. Vose. **Unbiased black box search algorithms**. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'11)*. 2011, 2035–2042. DOI: 10.1145/2001576.2001850 (see pages 70, 76).

[SGL07] Kumara Sastry, David E. Goldberg, and Xavier Llorà. **Towards billion-bit optimization via a parallel estimation of distribution algorithm**. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'07)*. 2007, 577–584. DOI: 10.1145/1276958.1277077 (see page 113).

[SH00] Thomas Stützle and Holger H. Hoos. **MAX-MIN Ant System**. *Future Generation Computer Systems* 16:8 (2000), 889–914. DOI: 10.1016/S0167-739X(00)00043-1 (see pages 65, 133, 134).

[Sha06] Jonathan L. Shapiro. **Diversity loss in general estimation of distribution algorithms**. In: *Proceedings of the 9th International Conference on Parallel Problem Solving from Nature (PPSN IX)*. 2006, 92–101. DOI: 10.1007/11844297_10 (see page 60).

[Sim13] Dan Simon. **Evolutionary optimization algorithms**. John Wiley & Sons, Inc., 2013. ISBN: 978-0-470-93741-9 (see page 2).

[ST03] Mikhail A. Semenov and Dmitri A. Terkel. **Analysis of convergence of an evolutionary algorithm with self-adaptation using a stochastic Lyapunov function**. *Evolutionary Computation* 11:4 (2003), 363–379. DOI: 10.1162/106365603322519279 (see page 30).

[ST12] Dirk Sudholt and Christian Thyssen. **A simple ant colony optimizer for stochastic shortest path problems**. *Algorithmica* 64:4 (2012), 643–672. DOI: 10.1007/s00453-011-9606-2 (see pages 114, 134).

[Sud13] Dirk Sudholt. **A new method for lower bounds on the running time of evolutionary algorithms**. *IEEE Transactions on Evolutionary Computations* 17:3 (2013), 418–435. DOI: 10.1109/TEVC.2012.2202241 (see page 12).

[SW16a] Dirk Sudholt and Carsten Witt. **Update strength in EDAs and ACO: how to avoid genetic drift**. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'16)*. 2016, 61–68. DOI: 10.1145/2908812.2908867 (see pages 5–7, 10, 66, 68, 91, 94, 103, 149, 150, 154, 169, 195, 197).

[SW16b] Dirk Sudholt and Carsten Witt. **Update strength in EDAs and ACO: how to avoid genetic drift**. *Computing Research Repository (arXiv)* (2016). URL: http://arxiv.org/abs/1607.04063 (see pages 154, 161).

[War16] Lutz Warnke. **On the method of typical bounded differences**. *Combinatorics, Probability and Computing* 25:2 (2016), 269–299. DOI: 10.1017/S0963548315000103 (see page 33).

[Wil91] David Williams. **Probability with martingales**. Cambridge University Press, 1991. ISBN: 978-0-521-40605-5 (see page 51).

[Wit06] Carsten Witt. **Runtime analysis of the $(\mu + 1)$ EA on simple pseudo-Boolean functions**. *Evolutionary Computation* 14 (2006), 65–86. DOI: 10.1162/evco.2006.14.1.65 (see page 197).

[Wit13] Carsten Witt. **Tight bounds on the optimization time of a randomized search heuristic on linear functions**. *Combinatorics, Probability and Computing* 22:2 (2013), 294–318. DOI: 10.1017/S0963548312000600 (see page 12).

[Wit17] Carsten Witt. **Upper bounds on the runtime of the Univariate Marginal Distribution Algorithm on OneMax**. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'17)*. 2017, 1415–1422. DOI: 10.1145/3071178.3071216 (see pages 6, 7, 10, 65, 149, 150, 192, 197, 223).

[Wit18] Carsten Witt. **Domino convergence: why one should hill-climb on linear functions**. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'18)*. 2018, 1539–1546. DOI: 10.1145/3205455.3205581 (see pages 68, 149, 210, 223).

[WKM17] Zijun Wu, Michael Kolonko, and Rolf H. Möhring. **Stochastic runtime analysis of the Cross-Entropy Algorithm**. *IEEE Transactions on Evolutionary Computations* 21:4 (2017), 616–628. DOI: 10.1109/TEVC.2017.2667713 (see page 197).

List of Publications

Articles in Refereed Journals

[1] **Escaping local optima using crossover with emergent diversity**. *IEEE Transactions on Evolutionary Computations* 22:3 (2018), 484–497. DOI: 10.1109/TEVC.2017.2724201. Joint work with Duc-Cuong Dang, Tobias Friedrich, Timo Kötzing, Per Kristian Lehre, Pietro S. Oliveto, Dirk Sudholt, and Andrew M. Sutton.

[2] **Unbiasedness of estimation-of-distribution algorithms**. *Theoretical Computer Science* (2018). DOI: 10.1016/j.tcs.2018.11.001. Joint work with Tobias Friedrich and Timo Kötzing.

[3] **Robustness of Ant Colony Optimization to noise**. *Evolutionary Computation* 24:2 (2016), 237–254. DOI: 10.1162/EVCO_a_00178. Joint work with Tobias Friedrich, Timo Kötzing, and Andrew M. Sutton.

[4] **The Compact Genetic Algorithm is efficient under extreme Gaussian noise**. *IEEE Transactions on Evolutionary Computations* 21:3 (2017), 477–490. DOI: 10.1109/TEVC.2016.2613739. Joint work with Tobias Friedrich, Timo Kötzing, and Andrew M. Sutton.

[5] **Routing for on-street parking search using probabilistic data**. *AI Communications* (2019). DOI: 10.3233/AIC-180574. Joint work with Tobias Friedrich, Ralf Rothenberger, Tobias Arndt, Danijar Hafner, Thomas Kellermeier, Simon Krogmann, and Armin Razmjou.

[6] **Lower bounds on the run time of the Univariate Marginal Distribution Algorithm on OneMax**. *Theoretical Computer Science* (2018). DOI: 10.1016/j.tcs.2018.06.004. Joint work with Carsten Witt.

Articles in Refereed Conference Proceedings

[7] **Probabilistic routing for on-street parking search**. In: *Proceedings of the 24th Annual European Symposium on Algorithms (ESA'16)*. 2016,

6:1–6:13. doi: 10.4230/LIPIcs.ESA.2016.6. Joint work with Tobias Arndt, Danijar Hafner, Thomas Kellermeier, Simon Krogmann, Armin Razmjou, Ralf Rothenberger, and Tobias Friedrich.

[8] **Memory-restricted routing with tiled map data**. In: *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC'18)*. 2018, 3347–3354. doi: 10.1109/SMC.2018.00567. Joint work with Thomas Bläsius, Jan Eube, Thomas Feldtkeller, Tobias Friedrich, J. A. Gregor Lagodzinski, Ralf Rothenberger, Julius Severin, Fabian Sommer, and Justin Trautmann.

[9] **Escaping local optima with diversity mechanisms and crossover**. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'16)*. 2016, 645–652. doi: 10.1145/2908812.2908956. Joint work with Duc-Cuong Dang, Tobias Friedrich, Timo Kötzing, Per Kristian Lehre, Pietro S. Oliveto, Dirk Sudholt, and Andrew M. Sutton.

[10] **Emergence of diversity and its benefits for crossover in genetic algorithms**. In: *Proceedings of the 14th International Conference on Parallel Problem Solving from Nature (PPSN XIV)*. 2016, 890–900. doi: 10.1007/978-3-319-45823-6_83. Joint work with Duc-Cuong Dang, Tobias Friedrich, Timo Kötzing, Per Kristian Lehre, Pietro Simone Oliveto, Dirk Sudholt, and Andrew M. Sutton.

[11] **Significance-based estimation-of-distribution algorithms**. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'18)*. 2018, 1483–1490. doi: 10.1145/3205455.3205553. Joint work with Benjamin Doerr.

[12] **EDAs cannot be balanced and stable**. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'16)*. 2016, 1139–1146. doi: 10.1145/2908812.2908895. Joint work with Tobias Friedrich and Timo Kötzing.

[13] **Robustness of Ant Colony Optimization to noise**. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'15)*. 2015, 17–24. doi: 10.1145/2739480.2754723. Joint work with Tobias Friedrich, Timo Kötzing, and Andrew M. Sutton.

[14] **The benefit of recombination in noisy evolutionary search**. In: *Proceedings of the 26th International Symposium on Algorithms and Computation (ISAAC'15)*. 2015, 140–150. DOI: 10.1007/978-3-662-48971-0_13. Joint work with Tobias Friedrich, Timo Kötzing, and Andrew M. Sutton.

[15] **Graceful scaling on uniform versus steep-tailed noise**. In: *Proceedings of the 14th International Conference on Parallel Problem Solving from Nature (PPSN XIV)*. 2016, 761–770. DOI: 10.1007/978-3-319-45823-6_71. Joint work with Tobias Friedrich, Timo Kötzing, and Andrew M. Sutton.

[16] **Fast building block assembly by Majority Vote crossover**. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'16)*. 2016, 661–668. DOI: 10.1145/2908812.2908884. Joint work with Tobias Friedrich, Timo Kötzing, Samadhi Nallaperuma, Frank Neumann, and Martin Schirneck.

[17] **First-hitting times for finite state spaces**. In: *Proceedings of the 15th International Conference on Parallel Problem Solving from Nature (PPSN XV)*. 2018, 79–91. DOI: 10.1007/978-3-319-99259-4_7. Joint work with Timo Kötzing.

[18] **First-hitting times under additive drift**. In: *Proceedings of the 15th International Conference on Parallel Problem Solving from Nature (PPSN XV)*. 2018, 92–104. DOI: 10.1007/978-3-319-99259-4_8. Joint work with Timo Kötzing.

[19] **Lower bounds on the run time of the Univariate Marginal Distribution Algorithm on OneMax**. In: *Proceedings of the 14th ACM/SIGEVO Workshop on Foundations of Genetic Algorithms (FOGA XIV)*. 2017, 65–79. DOI: 10.1145/3040718.3040724. Joint work with Carsten Witt.