UNIVERSITY OF BIRMINGHAM

RUNTIME ANALYSES OF UNIVARIATE ESTIMATION OF DISTRIBUTION ALGORITHMS UNDER LINEARITY, EPISTASIS AND DECEPTION

By

PHAN TRUNG HAI NGUYEN

A thesis submitted to
the University of Birmingham
for the degree of
DOCTOR OF PHILOSOPHY

School of Computer Science
College of Engineering and Physical Sciences
University of Birmingham
February 2021

Acknowledgements

Firstly, I would like to express my sincere gratitude to my supervisor, Dr Per Kristian Lehre, for the continuous support of my Ph.D study and related research, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better supervisor for my Ph.D study.

Besides, I would like to thank the rest of my thesis group, Prof. Jon Rowe and Prof. Ata Kaban, for their insightful comments and encouragement, but also for the hard questions which incented me to widen my research from various perspectives.

Last but not least, I would like to thank my family, my parents and my younger brother, for supporting me mentally throughout the writing of this thesis and my life in general.

Abstract

Estimation of distribution algorithms (EDAs) have been successfully applied to solve many real-world optimisation problems. The algorithms work by building and maintaining probabilistic models over the search space and are widely considered a generalisation of the evolutionary algorithms (EAs). While the theory of EAs has been enriched significantly over the last decades, our understandings of EDAs are sparse and limited. The past few years have seen some progress in this topic, showing competitive performance compared to other EAs on some simple test functions. This thesis studies the so-called univariate EDAs by rigorously analysing their time complexities on different fitness landscapes. Firstly, I show that the algorithms optimise the ONEMAX function as efficiently as the $(1+1)$ EA does. I then investigate the algorithms' ability to cope with dependencies among decision variables. Despite the independence assumption, the algorithms optimise LEADINGONES – a test function with an epistasis level of $(n-1)$ – using at most $\mathrm{O}\left(n^2\right)$ function evaluations under appropriate parameter settings. I also show that if the selection rate μ/λ is above some constant threshold, an exponential runtime is inevitable to optimise the function. Finally, I confirm the common belief that univariate EDAs have difficulties optimising some objective function when deception occurs. By introducing a new test function with a very mild degree of deception, I show that the UMDA takes an exponential runtime unless the selection rate is chosen extremely high, i.e., $\mu/\lambda = \mathrm{O}(1/\mu)$. This thesis demonstrates that while univariate EDAs may cope well with independence and epistasis in the environment, the algorithms suffer even at a mild level of deception and that researchers might need to adopt multivariate EDAs when facing deceptive objective functions.

Table of contents

Chapter 1

Introduction

1.1 Motivations

Optimisation appears in many day-to-day tasks. For example, when travelling between two destinations, there are many routes available, but we usually try to find the shortest one. In essence, optimisation is a problem for which we need algorithms to select the best solution from a pool of candidate solutions in light of some constraint. What makes the problem challenging is that one cannot afford to evaluate all solutions and therefore has to find the best solutions, while only looking at some of the solutions. Optimisation problems arise in finance, scheduling, network design and operation, supply chain management, and many other areas [11].

We shall describe (unconstrained) optimisation problems mathematically. A set of candidate solutions is called the search space, denoted as $\mathscr{X}$, of which a candidate solution x is an element of it, that is, $x \in \mathscr{X}$. The optimisation problem is discrete when the search space is finite. The objective function $f : \mathscr{X} \to \mathbb{R}$, which is a member of a more general problem class F, assigns to each candidate solution $x \in \mathscr{X}$ a real value describing its quality. We say that a candidate solution is fitter than another one if the fitness value of the former is higher then that of the latter. Optimisation then involves the process of finding a global optimum, denoted as x^*, such that

$$f(x^*) = \max\{f(x) : x \in \mathscr{X}\} \qquad \text{(maximisation)},$$

or

$$f(x^*) = \min\{f(x) : x \in \mathscr{X}\} \qquad \text{(minimisation)}.$$

When maximising, f is called a fitness (or utility) function, whereas in minimisation it is viewed as a cost function. In this thesis, we shall consider only the maximisation of pseudo-Boolean functions. However, the result holds for minimisation since minimising a cost function f can be interpreted as maximising a fitness function $-f$. A fitness landscape is often used to visualise the relationship between the candidate solution $x \in \mathscr{X}$ and the value of the fitness function $f(x)$ [132, 113].

An optimisation algorithm computes a solution for an instance of an optimisation problem to some given accuracy [11]. Many optimisation algorithms have been developed since the late 1940s [11]. Each algorithm makes some assumption on the problem class F to which they will be applied. If the closed-form (mathematical representation) of the objective function f is differentiable or has some known convexity property, then popular techniques like gradient descent [84], the simplex method [101, 11], Newton-Raphson method [28] or the ellipsoid method [102] are applicable. However, these assumptions are usually considered very strong, and one can find many real-world scenarios that do not conform to these assumptions [13]. One example is the black-box setting when the analytic form of the objective function f is not known, but one can obtain the function values for any candidate solution $x \in \mathscr{X}$.

The black-box model is a mathematical model of the optimisation scenario where the structure of the objective function is hidden from the algorithms. In a black-box scenario, we assume the existence of an oracle who accepts our query $x \in \mathscr{X}$ and returns the function value $f(x)$ [40]. If one query is sent to the oracle at a time, then at time $t \in \mathbb{N}$ all we know is the sequence $((x_t, f(x_t)) : t \in \mathbb{N})$.

Significant effort have been devoted to developing optimisation algorithms for black-box settings. These black-box methods are often categorised into two classes based on whether the objective function's evaluation is costly. Many algorithms have been developed for expensive black-box functions [67, 53, 60, 120]. Examples of costly black-box functions are drug tests and financial investments [13]. For cheap black-box functions, randomised search heuristics that make use of many random decisions and are easy to implement and parallelise [107] are popular choices. Furthermore, unlike the classical approach to algorithms, where one develops an algorithm with the aim of obtaining some runtime bound and proof of

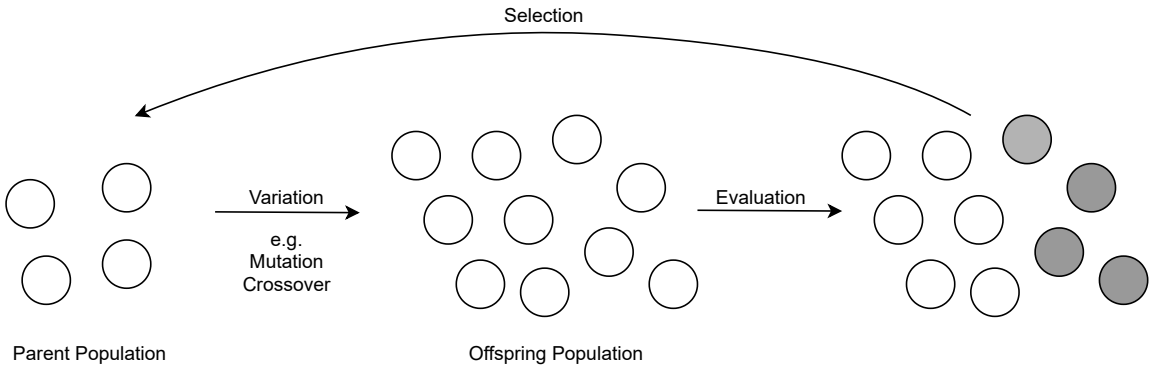

Fig. 1.1 A pictorial illustration of Evolutionary Algorithms

correctness in mind, they are general-purpose algorithms and apply to different problems defined in a given search space [107].

Evolutionary computing is an area of research in which randomised search heuristics inspired by Darwin's principle of the survival of the fittest are applied to solve optimisation problems. Typical members are evolutionary algorithms (EAs [59, 112, 114]), which often represent candidate solutions (also called individuals) to the considered problem as bitstrings of length $n \in \mathbb{N}$. In this case, the space of all solutions is $\mathscr{X} = \{0,1\}^n$, and the objective function f is called a pseudo-Boolean optimisation problem. The class of EAs considered in this thesis follows an iterative process, in which the so-called parent population of $\mu \in \mathbb{N}$ individuals is sampled from the space of all populations $\mathscr{X}^\mu$. In each iteration, another so-called offspring population of $\lambda \in \mathbb{N}$ individuals is created by modifying those in the parent population using variation operators such as mutation and crossover. The objective function f then evaluates the qualities of new individuals. Finally, some fittest individuals in either or both populations are selected to form a new parent population for the next iteration. The algorithms halt and return the fittest individual in the current population if some predefined terminal condition has been fulfilled. Common criteria are when a threshold on the number of iterations has been exceeded or if no progress has been seen for a certain number of iterations [106]. Fig. 1.1 provides an illustration of typical EAs.

For EAs, mutation and crossover are essential for constructing new solutions. In the standard bitwise mutation, each bit in an individual is flipped independently with a fixed mutation probability of χ/n for some constant $\chi > 0$. Let $x = (x_1, x_2, \ldots, x_n)$ be the individual, and $z = (z_1, z_2, \ldots, z_n)$ be the new offspring created by the standard mutation, we have $\Pr(z_i = 1 - x_i) = \chi/n$ and $\Pr(z_i = x_i) = 1 - \chi/n$ for all $i \in \{1, 2, \ldots, n\} =: [n]$. Unlike

mutation where a single individual is altered, crossover involves at least two individuals [107]. One example is the uniform crossover. If $x = (x_1, x_2, \ldots, x_n)$ and $y = (y_1, y_2, \ldots, y_n)$ are two individuals, and $z = (z_1, z_2, \ldots, z_n)$ is the new offspring created by the uniform crossover, then $\Pr(z_i = x_i) = \Pr(z_i = y_i) = 1/2$ for all $i \in [n]$.

Evolutionary algorithms have been employed to tackle many real-world challenges. Early applications include combinatorial optimisation problems like travelling salesman problem [12], minimum spanning tree [105], and knapsack problem [61]. More recent applications range from supply chain management and renewable energy [10, 121, 103] to the creation of music and art [38, 88, 104]. Furthermore, the working principles of evolutionary algorithms have led to a new population-based training method for neural networks in deep learning [62, 89, 21] and new designs of the LSTM cell in the recurrent neural network in neural architecture search, outperforming the current state-of-the-art results in natural language processing [18]. See [117] for more applications.

Although EAs themselves and other EA-inspired methods have obtained remarkable results on many real-world problems, there exist simple problems where the algorithms take a significant amount of time to find a global optimum. These problems could have a high level of epistasis [50, 51], which corresponds to the maximum number of other decision variables each variable depends on [25, 52]. Take a test problem named $\textsc{Plateau}(a,b)$ for $a, b \in \mathbb{N}$ (also called Royal Road function [100, 45, 93]) as an example. Assume that $n = a \cdot b$, the objective value for an input $x = (x_1, x_2, \ldots, x_n) \in \mathscr{X}$ is

$$f(x) = \sum_{i=1}^{a} \left(\prod_{j=(i-1)b+1}^{ib} x_j \right). \tag{1.1}$$

Here, the bitstring x is partitioned into a consecutive non-overlapping blocks of length b. All the bits in each block have to be all ones to increase the overall fitness value by one; otherwise, there is no contribution. In other words, each bit depends on the other $b-1$ bits in its block. For that reason, we say that the function has an epistasis level of $b-1$. One could blame the failure of EAs in this case on the lack of an internal mechanism to learn the interactions among decision variables explicitly. This is because the mutation and crossover operators assume independence between bit positions and do not explicitly provide any 'belief' on which other decision variables a variable is likely to depend. If

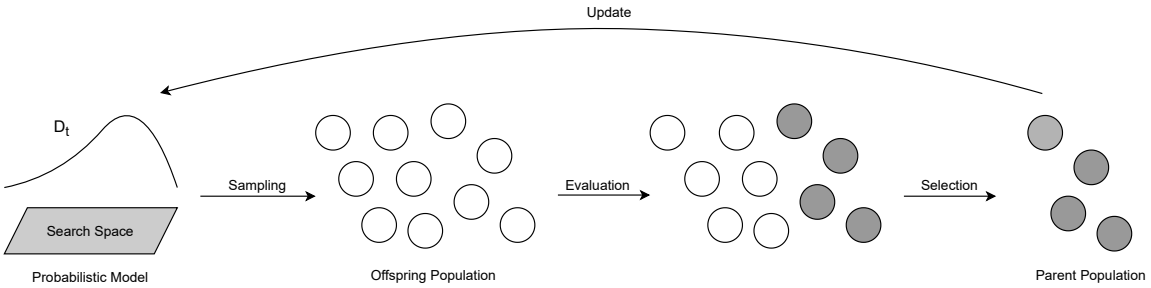

Fig. 1.2 A pictorial illustration of typical EDAs.

some prior knowledge on the objective function f is provided, one can design a problem-specific variation operator to discover the optimal solutions more efficiently. However, such information is not always available under a black-box model, and designing a problem-specific operator might still be laborious.

A more promising approach would be to equip the algorithms with some mechanism to learn (or at least attempt to learn) explicitly the variable dependencies. By doing so, we can reduce the need for human intervention. The algorithm's belief in the distribution of the global optima can be represented by a probabilistic model $\mathscr{D} : \mathscr{X} \to \mathbb{R}$, parameterised by $\theta \in \Theta$, such that $\mathscr{D}(x) \geq 0$ for all $x \in \mathscr{X}$ and $\sum_{x \in \mathscr{X}} \mathscr{D}(x) = 1$. In other words, the algorithm employs a family of distributions $\{\mathscr{D}(\cdot \mid \theta) : \theta \in \Theta\}$. The choice of the distribution family implies the degree of variable dependencies learnable by the algorithms. New offspring are created by sampling the probabilistic model, that is, $x \sim \mathscr{D}(x \mid \theta)$. In essence, our beliefs over time is a sequence $\{\mathscr{D}_t : t \in \mathbb{N}\}$. At the time $t \in \mathbb{N}$, some update mechanism is employed to obtain the next probabilistic model $\mathscr{D}_{t+1}$ from the current one $\mathscr{D}_t$. It is hoped that in the end, the probabilistic model $\mathscr{D}_t$ will converge to an 'optimal distribution' in the search space. Here, the optimal distribution can be interpretted as a particular distribution $\mathscr{D}$, parameterised by a $\theta^* \in \Theta$, from which the probability of sampling the global optima is highest.

Estimation of Distribution Algorithms (EDAs [99, 56, 6, 27, 111]), see Fig. 1.2, follow the approach above. The starting model is the uniform distribution, that is, $\mathscr{D}_0(x) = 1/2^n$ for all $x \in \mathscr{X}$ as there are 2^n search points in the search space $\mathscr{X} = \{0,1\}^n$. EDAs then follow an iterative procedure of belief refinement using the signal from the objective function f. More specifically, the algorithms in an iteration $t \in \mathbb{N}$ rely on a probabilistic model $\mathscr{D}_t$, from which an offspring population of λ individuals is drawn. They then rank these individuals according to the objective function f. The algorithms employ some selection mechanism to select μ out of λ individuals to form the parent population, from which the next probabilistic model

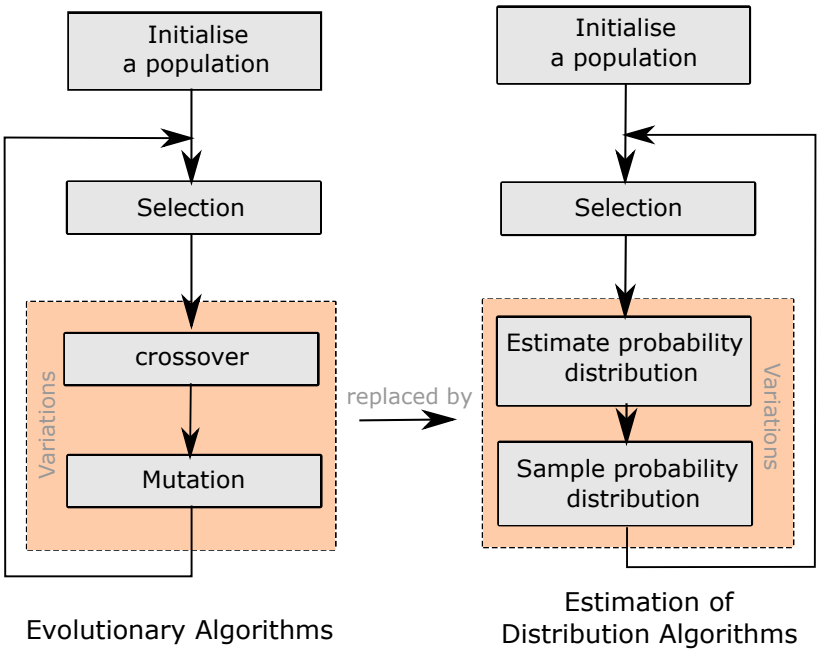

Fig. 1.3 Comparison between EAs and EDAs.

$\mathscr{D}_{t+1}$ is constructed. Different EDAs employ different model-update mechanisms; some are very simple [99, 56, 6], while others are more computationally intensive [27, 98, 55, 111]. Furthermore, each algorithm makes some assumption about the choice for the structure of the probabilistic model $\mathscr{D}$. The simplest class of EDAs assume independence among decision variables, while more complex EDAs allow an arbitrary learnable degree of interactions.

Figure 1.3 [115] illustrates the main difference between EAs and EDAs. While the source of variation in EAs comes from mutation and crossover, EDAs obtain variation via sampling and updating the probabilistic models. They are widely regarded as a generalisation of the EAs in an attempt to learn the interplays among decision variables, but could an EDA always return a solution better than that of an EA on all optimisation problem? The No Free Lunch Theorem (NFLT) [131] provides an answer to this question, and it is no. The theorem states that any two optimisation algorithms are equivalent when their performances are averaged across all possible problems, assuming that each point in the search space is evaluated at most once [131]. Of course, the set of all possible problems is of little interest since a vast majority of them are random, and we are only interested in a tiny subset of those problems (also called real-world problems). The premise of the NFLT is violated on this set, so the average performances of EDAs and EAs may be different. In other words, it only makes sense to compare performances of optimisation algorithms on a problem class [107].

The theory community of evolutionary computing uses tools from probability theory to analyse evolutionary algorithms rigorously for their runtime behaviours. In these studies, we consider the algorithms as infinite stochastic processes. The algorithms halt only when one of the global optima has been found for the first time. We then define the runtime as the number of queries sent to the oracle by the algorithms thus far. Because of the randomness involved, the runtime of a randomised search heuristic is a random variable. Therefore, we are interested in the runtime distribution properties, particularly the expected runtime and in the size of the tails of the runtime distribution [108]. Because the runtime likely grows with the problem instance size, an asymptotic notation is used to describe it. We shall assume that the reader is familiar with the asymptotic notation; otherwise, we refer the reader to [19, Chapter 3] for an introduction. For the rest of the thesis, all asymptotic statements refer to the problem instance size $n \in \mathbb{N}$. There are some pseudo-Boolean functions, e.g., ONEMAX, LEADINGONES and BINVAL, which are commonly used to compare the performances of randomised search heuristics [108]. At a glance, these functions look very simple, but each represents a different characteristic of the fitness landscapes and aims at testing a unique ability of the algorithms [129]. It is hoped that the analyses on test functions could give some insights on the runtime behaviours of the algorithms on more complex problems.

There are some advantages to studying the runtime of the algorithms from a theoretical perspective. First and foremost, the runtime bounds provide a guarantee on the performances of the algorithms for a wide range of algorithm-specific parameters [108]. Moreover, the analyses often reveal the bottleneck of the algorithms, which could lead to a redesign or the creation of a new optimisation technique. One example is the significance-based compact genetic algorithm [32], which has been recently proposed to avoid the so-called genetic drift [4] when the univariate EDAs optimise the LEADINGONES function. The newly proposed algorithm optimises the LEADINGONES function using at most $\mathrm{O}(n \log n)$ function evaluations in expectation as opposed to the conjectured runtime of $\Omega(n^2)$ for the CGA [47]. Finally, theoretical studies have encouraged researchers to develop new proof techniques, tailored to the working principles of randomised search heuristics, which benefit the theory community of evolutionary computing and other communities. One notable example is drift analysis [54, 58], which has been developed significantly to support the runtime analyses of EAs [35, 86]. It is noted that researchers from the field of classic algorithms have started using variants of drift theorems developed in our field (see [35] and references therein).

Runtime analyses are considered challenging tasks because randomised search heuristics are general-purpose optimisation techniques and were not originally designed to support rigorous analyses [74, 107]. Despite that, the theory of EAs has been enriched significantly since the first runtime analysis of an EA in 1992 [97]. However, our understandings of EDAs are still sparse and limited. The past few years have seen some progress in this topic, showing competitive performance compared to other EAs on some simple test functions [35], but it is far from enough. Taking the compact Genetic Algorithm (cGA) and the Univariate Marginal Distribution Algorithm (UMDA) as examples. They are the simplest variants of EDAs [57] that rely on simple probability vector-based models. The first and only rigorous runtime analysis of the cGA, however, was completed in 2006 [39], whereas the first upper bounds on the expected runtime of the UMDA on OneMax and LeadingOnes were not published until 2015 [23]. Many questions remain open whether the EDAs could locate the global optima on different fitness landscapes 'efficiently' and have a competitive performance to other EAs. We note here that efficiency for different landscapes could correspond to different runtime bounds. As the theoretical understanding widens the applicability and improves the performance of evolutionary computing methods [35], in this thesis we aim at obtaining more insights on the performances of the algorithms on different fitness landscapes. To do that, we need some test function whose fitness landscape exhibits the property on which we hope to analyse the runtime behaviours of the algorithms. If such a test function already exists, we then use it; otherwise, a new test function will be proposed if necessary.

1.2 Research questions

There are four EDA-related research questions that capture our attention, and in this thesis, we will try to answer them by conducting rigorous runtime analyses.

Question 1. *How well do univariate* EDA*s cope with linearity in objective functions?*

One typical test function with this property is the OneMax function [97], where flipping a 0-bit to a 1-bit always results in an improvement in the overall fitness. Furthermore, the fitness landscape of the function resembles a hill, and in order to find the global optimum the evolving population needs to climb uphill (also called hill-climbing). Other candidates are the BinVal function [108] and the more general class of linear functions [40]. An algorithm is said to optimise OneMax efficiently if its expected runtime is $O(n \log n)$, where $n \in \mathbb{N}$ in

the problem instance size, since the function's unary unbiased black-box complexity is of order $n \log n$ [82, Theorem 6].

The rationale behind the studying of linear functions stems from its comparatively simple setting with very little interactions between decision variables. Before thinking of more complicated problems, it is necessary to understand how long the algorithms would take to optimise a linear function. If the algorithms get stuck in solving a linear function, then so is likely for other problems.

Question 2. *How long do univariate* EDA*s take to optimise an objective function with a high degree of epistasis?*

Recall that epistasis refers to the maximum number of other decision variables a variable depends on [25, 52]. The algorithms could try to learn such a dependence to locate global optima efficiently. We are interested in this question since the class of univariate EDAs assume independence between decision variables, so it is unsure whether the algorithms could still optimise an objective function with a high degree of epistasis. Likely, real-world optimisation problems often have some dependence between their decision variables, so the answer to this question will shed light on the algorithms' applicability in practice.

One test function with this characteristic is the LEADINGONES function, which has an epistasis level of $n-1$. As discussed in [47], we say that an algorithm optimises the LEADINGONES function efficiently if its expected runtime on the function is $\mathrm{O}\left(n^2\right)$.

Question 3. *What is the runtime behaviour of some univariate* EDA*s when noise is introduced to the objective functions?*

The question concerns noisy fitness evaluations. We would like to investigate how noise could influence the runtime behaviour of univariate EDAs. Noise appears in many real-world applications, and sometimes it is impossible to avoid them (due to machine/human errors), so it is necessary to include them in the theoretical analyses to extend our understanding of the algorithms. In a recent survey, Doerr & Neumann [35] have pointed out that there are many open problems in the noisy optimisation of EDAs. Note in particular that this is not the first time that noise has been considered in runtime analyses of EDAs. A posterior Gaussian noisy model has been taken into account when analysing the runtime of the compact Genetic Algorithm [56] on the ONEMAX function [46].

Question 4. *Deceptive objective functions like the famous* TRAP *function [2] are shown to pose a challenge to many* EA*s. Will there be any difference in the univariate* EDA*s when facing deceptive fitness landscapes?*

The fitness 'signal' is deceptive [124]. Unlike ONEMAX and LEADINGONES, in which an improved fitness means that the algorithm reduces the genotypic distance to an optimum, deception means that this belief is not always the case. The famous TRAP function [108] is a good example. Currently, there is no threshold for efficiency on the TRAP function except the borderline between polynomial and exponential runtimes.

The $(1+1)$ EA is shown to take an expected runtime of $\Omega(n^n)$ on the TRAP function [108, Theorem 2.5]. If we could prove that the UMDA copes well with deception (not necessarily on the TRAP function), such a result is significant as it would show the substantial advantage of univariate probabilistic models over simple genetic operators like mutation and crossover. Furthermore, while there exists some test function on which the simple $(1+1)$ EA takes an exponential runtime as opposed to a polynomial runtime of the UMDA [14], we still do not know any test function where the reverse would happen. In this thesis, we shall try to answer these questions together.

As a final remark, given that time complexity (along with space) is measured to judge the efficiency of an optimisation algorithm and that the currently known runtime results of EDAs are very limited, the answers to these research questions will rapidly reshape and possibly extend the applicability of EDAs in practice.

1.3 Our contributions

In this thesis, we consider the simplest class of univariate EDAs, of which the univariate marginal distribution algorithm (UMDA [99]) and the population-based incremental learning (PBIL [6]) are two notable members. The contributions of this thesis are four-fold.

We first study the UMDA's ability to hill climb by analysing its runtime on the ONEMAX function. By applying the level-based theorem [20] that is a general method to derive upper bounds on the expected runtimes of population-based optimisation algorithms, we show that the UMDA uses at most $O(n\lambda)$ function evaluations to locate the global optimum when the offspring population size is $\mu = \Omega(\log n) \cap O(\sqrt{n})$. Furthermore, if a larger population

size is chosen, i.e., $\mu = \Omega(\sqrt{n}\log n)$, an expected runtime of $\mathrm{O}(\lambda\sqrt{n})$ is guaranteed. These results show that the UMDA requires an expected runtime of $\mathrm{O}(n\log n)$ on ONEMAX under appropriate parameter settings and also improve the best known upper bound of $\mathrm{O}(n\log n\log\log n)$ in [23] by a factor of $\Theta(\log\log n)$. Together with the recent lower bound of $\Omega(\mu\sqrt{n}+n\log n)$ in [71], we conclude a tight bound of $\Theta(n\log n)$ for the UMDA on the ONEMAX function, which matches the performance of the simple $(1+1)$ EA on the same problem.

We then consider objective functions where epistasis is present. The UMDA is currently known to take $\mathrm{O}\left(n\lambda\log\lambda+n^2\right)$ function evaluations in expectation to optimise the LEADINGONES function. The required population sizes are $\lambda = \Omega(\log n)$ and $\mu \leq \lambda/(e(1+\delta))$ for some constant $\delta \in (0,1)$. We show that if the parent population size is slightly larger and $\mu = o(n^{1/k})$ for some constant $k \in [0,1)$, the algorithm requires an exponential runtime of $2^{\Omega(\mu^k)}$. This result points out that $1/e$ is the point for the ratio μ/λ when the runtime turns from polynomial to exponential. We also show that the PBIL with a smoothing parameter $\rho \in (1/e,1]$ needs at most $\mathrm{O}\left(n^2\right)$ function evaluations to optimise the LEADINGONES function under appropriate parameter settings, assuming that all marginals are lower bounded by some constant during the optimisation. Despite the independence assumption, univariate EDAs can still cope very well with epistasis. Finally, for the first time, we introduce noise into the LEADINGONES function (also called prior noise) and show that the UMDA cope well with noise. It has been pointed out in [35] recently that there still exist many open problems in the noisy optimisation in EDAs. We hope that our results here will encourage more to come shortly.

We show that the UMDA suffers when the objective function is mildly deceptive. More specifically, the algorithm takes an exponential expected runtime of $2^{\Omega(\mu^k)}$ to optimise the so-called DLB function when the population sizes are $\mu = o(n/\log n)$ and $\mu \leq \lambda = \mathrm{O}\left(\mu^{2-k}\right)$ for some constant $k \in (0,1]$. Otherwise, an upper bound of $\mathrm{O}\left(n^3\right)$ can be achieved with appropriate choices of μ and λ. In the latter case, the algorithm requires a selection rate of $\mu/\lambda = \mathrm{O}(1/\mu)$, which is extremely high. One might not be aware of this unusual setting in practice. Since the deception level in the DLB function is very mild, we confirm the common belief that univariate EDAs have difficulty optimising deceptive fitness functions.

Methodological contribution: Our analyses demonstrate that the level-based theorem, accompanied by the anti-concentration properties of the Poisson binomial distribution, can yield, relatively quickly, asymptotically tight upper bounds for non-trivial, population-based

algorithms. Unless the variance of the sampled individuals is not too small, the distribution of the population cannot be too concentrated anywhere, even around the mean, yielding sufficient diversity to discover better solutions. Furthermore, since univariate EDAs represent the probabilistic models as probability vectors, the concept of majorisation will play an essential role in obtaining runtime results for these algorithms. We expect that similar arguments will lead to new results in runtime analysis of evolutionary algorithms.

1.4 Outline of the thesis

The thesis is structured as follows.

- Chapter 2 provides a brief introduction to estimation of distribution algorithms, including the representation of the probabilistic graphical models.
- Chapter 3 introduces the test functions, basic probability theory, and methods for deriving runtime of randomised search heuristics. The chapter ends with a summary of the current state-of-the-art runtime bounds for the class of univariate EDAs on test functions.
- Chapter 4 presents runtime analyses of the univariate marginal distribution algorithm on the ONEMAX function. Two different regimes on the parent population size are considered.
- Chapter 5 investigates the ability of univariate EDAs to cope with epistasis. An exponential lower bound is shown for the UMDA on the LEADINGONES function under low selection rate. The chapter also considers the PBIL, a generalisation of the UMDA. We then introduce noise into the fitness evaluation of the LEADINGONES function (so-called prior noise), and in the end, we obtain upper bounds on the expected runtime for the UMDA and the PBIL on the noisy LEADINGONES function.
- Chapter 6 introduces a new test function called DLB to study the behaviour of the UMDA when deception is present. We show that mild deception causes trouble for the UMDA with a small population size as the algorithm assumes independence between decision variables.

- Chapter 7 reiterates the significant contributions of the thesis and points out potential future work.

1.5 Publications

Chapters 4-6 are each derived from different publications. The work presented in Chapter 4 was derived from the joint work with Per Kristian Lehre published at the 2017 Genetic and Evolutionary Conference (GECCO '17) [76] and another joint work with Duc-Cuong Dang and Per Kristian Lehre published at the Springer Algorithmica journal [24]. Chapter 5 presents the results published as joint works with Per Kristian Lehre at GECCO '19 [79] and at the 2018 Conference on Parallel Problem Solving from Nature (PPSN '18) [77]. An extended version of the two papers is under review to be published at the Springer Algorithmica journal. Finally, Chapter 6 uses the material presented at the 2019 Conference on Foundation of Genetic Algorithms (FOGA '19) [78].

Chapter 2

Estimation of Distribution Algorithms

Recall that, in this thesis, we consider only the problem of maximising a pseudo-Boolean function $f : \mathscr{X} \to \mathbb{R}$ in the search space $\mathscr{X} = \{0,1\}^n$. In other words, we aim at finding one of the global optima x^* such that $f(x^*) = \max\{f(x) : x \in \mathscr{X}\}$. We shall assume that the readers have some basic understanding of the concepts in probability theory, including probability space $(\Omega, \mathscr{F}, \Pr)$, events, random variables, probability distributions, expectations, variances, stochastic processes and adapted filtration. Otherwise, we refer the readers to excellent textbooks [125, 43, 44, 95] for more information.

We also recall that a random variable Y is said to follow a Bernoulli distribution with success probability $p \in [0,1]$, denoted as $Y \sim \mathrm{Ber}(p)$, if and only if $\Pr(Y = 1) = p$ and $\Pr(Y = 0) = 1 - p$ [95, p. 25]. If there are $n \in \mathbb{N}$ such random variables (with the same success probability p), then the sum of them (i.e., a random variable X) follows a binomial distribution with n trials and success probability p, denoted as $X \sim \mathrm{Bin}(n, p)$ [95, p. 25]. An extension of the binomial distribution is the Poisson binomial (PB) distribution, in which each of n random variables can have a different success probability [43, p. 263]. More formally, we write $X \sim \mathrm{PB}(p_1, p_2, \ldots, p_n)$ if and only if $X = \sum_{i=1}^{n} X_i$, where $\Pr(X_i = 1) = p_i$ and $\Pr(X_i = 0) = 1 - p_i$ for all $i \in [n]$.

2.1 Probabilistic graphical models $\mathscr{G}(V,E)$

EDAs work by repeatedly refining the belief in the distribution of the global optima in the search space. The belief is represented by a probabilistic model $\mathscr{D} : \mathscr{X} \to \mathbb{R}$. A family of the probabilistic models $\mathscr{D}(x)$ implies the learnable degree of variable dependencies.

Let $X_1, X_2, \ldots, X_n$ be n decision variables of the fitness function $f(x)$. The probabilistic model $\mathscr{D}(x)$ is a specification of a joint probability distribution of all decision variables. In other words, for a bitstring $x = (x_1, x_2, \ldots, x_n) \in \mathscr{X}$ we can express

$$\begin{aligned} \mathscr{D}(x) &= \Pr(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n) \\ &= \Pr(x_1, x_2, \ldots, x_n). \qquad \text{(in short)} \end{aligned}$$

To describe how the probabilistic model $\mathscr{D}(x)$ captures the variable dependencies, we can apply the chain rule [43, p. 106] to decompose the joint probability distribution as follows.

$$\Pr(x_1, x_2, \ldots, x_n) = \Pr(x_1) \Pr(x_2 \mid x_1) \cdots \Pr(x_n \mid x_1, x_2, \ldots, x_{n-1}). \tag{2.1}$$

The first term on the right-hand side, i.e., $\Pr(x_1)$, is of little interest because it is the marginal probability of the decision variable X_1. However, the other conditional probabilities capture the potential variable dependencies. Take the second term $\Pr(x_2 \mid x_1)$ as an example. It tells us the relationship between X_1 and X_2. If the two are mutually independent [43, p. 204], we then have $\Pr(x_2 \mid x_1) = \Pr(x_2)$. Similarly for the conditional probability $\Pr(x_3 \mid x_1, x_2)$ with respect to three variables, X_1, X_2 and X_3. If X_3 is independent of X_1 given X_2, the probability is reduced to $\Pr(x_3 \mid x_2)$. Otherwise, if they are independent, the conditional probability then becomes $\Pr(x_3)$. We note that the decomposition in (2.1) is applicable to any permutation of the n decision variables.

We can describe the decomposition in (2.1) using a graphical model $\mathscr{G}(V,E)$ (also called a Bayesian network [109, 68]), where $V = \{X_1, X_2, \ldots, X_n\}$ and E is a set of directed arcs that connect pairs of nodes and describes the direct variable dependencies. The conditional probability $\Pr(X_i = x_i \mid X_j = x_j)$ is represented by an arc linking node X_j to node X_i, which are called parent and child, respectively [109]. The set of observed values for all parents of a node X_i is denoted as $\mathrm{pa}(X_i)$. More importantly, the structure of $\mathscr{G}(V,E)$ describes the conditional variable independence. For instance, given a simple graphical model of

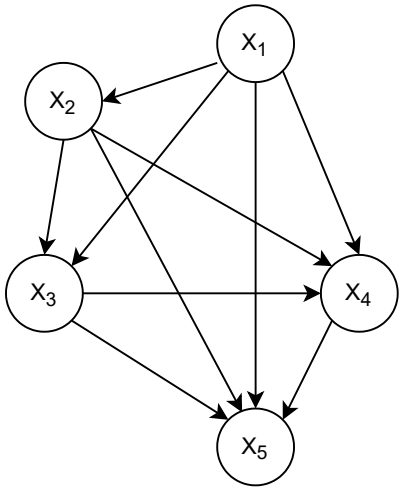

Fig. 2.1 Graph describing the chain rule decomposition for 5 decision variables.

three variables $A \to B \to C$, we have $\Pr(C \mid A,B) = \Pr(C \mid B)$, i.e., A and C are said to be conditionally independent given B [109]. A node without parents is called a root, while any node without children is a leaf [69]. The decomposition in (2.1) for the case $n = 5$ is illustrated in Fig. 2.1. We can now rewrite (2.1) in the form:

$$\Pr(x_1,x_2,\ldots,x_n) = \prod_i \Pr(x_i \mid \mathrm{pa}\,(X_i)). \tag{2.2}$$

We also note that the conditional probability $\Pr(X_i = x_i \mid X_j = x_j)$ can be fully specified using a lookup table, whose rows and columns are possible observed values of the decision variables X_i and X_j, respectively, and the entry (m,n) is the probability of the event $\{X_i = x_m\}$ given that the event of $\{X_j = x_n\}$ has already happened. Furthermore, each column in the table has a sum of one across all rows, which corresponds to a probability mass function for the decision variable X_i conditional on an observed value of X_j. Overall, the number of entries in the table scales as $\Theta(2^K)$, where K is the number of decision variables involved in the conditional probability. It is clear that the last term on the right-hand side of (2.1) requires the construction of a table with $\Theta(2^n)$ entries, which is computationally expensive and usually considered intractable in practice. Furthermore, each node in $\mathscr{G}(V,E)$ needs to store a table describing the relationship between that node and its parents.

Of course, the decomposition based on the chain rule is very general as it applies to any choice of probability distributions. However, it is the absence of arcs in $\mathscr{G}(V,E)$ that conveys

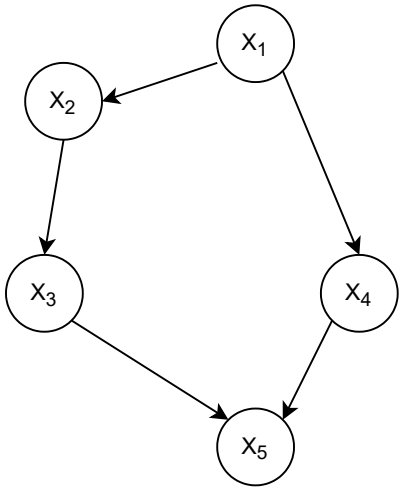

Fig. 2.2 Graph describing the dependencies between 5 decision variables.

interesting information [9]. Consider now a new graphical model of five nodes in Fig. 2.2. In this case, the joint probability distribution can be factorised as follows.

$$\Pr(x_1,x_2,\ldots,x_5) = \Pr(x_1)\Pr(x_2 \mid x_1)\Pr(x_3 \mid x_2)\Pr(x_4 \mid x_1)\Pr(x_5 \mid x_3,x_4).$$

The space complexity of the new graphical model scales at $\Theta(n^3)$ as opposed to $\Theta(n^5)$ of the decomposition based on the chain rule described in Fig. 2.1. In other words, a space-saving factor of $\Theta(n^2)$ is attained.

Sampling from a $\mathscr{G}(V,E)$ is straightforward via ancestral sampling [109, 68]. At first, the root node is sampled using its marginal probability. Once this is done, the root's children are next to go using the probabilities conditional on the root's observed value. More specifically, a child node's value is sampled using a probability mass function specified in its lookup table at a column corresponding to the root's sampled value. The procedure is repeated at the root's grandchildren and only stops when all leaf nodes have been sampled.

We present a simple probabilistic graphical model of five nodes in Example 1.

Example 1. *Consider a probabilistic graphical model with five nodes as in Fig. 2.3, where X_1 is the root node, and X_4 and X_5 are leaf nodes. Each node can only take values in $\{0,1\}$. The marginal distribution of X_1 and conditional distributions of X_2, X_3, X_4, and X_5 are specified as follows.*

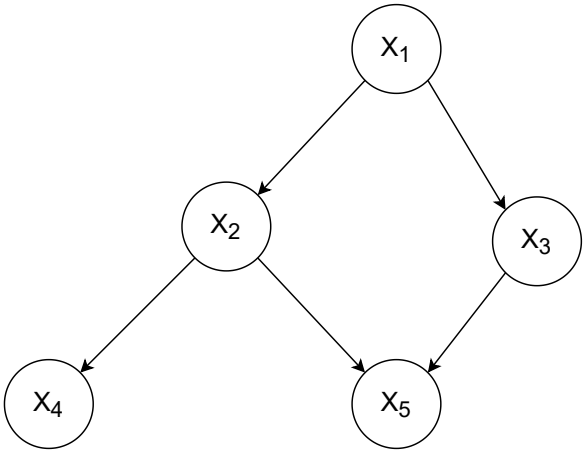

Fig. 2.3 A simple probabilistic graphical model with 5 nodes

X_1=*1*	***0.2***
X_1=*0*	*0.8*

	X_1=*1*	X_1=*0*
X_2=*1*	*0.3*	*0.5*
X_2=*0*	***0.7***	*0.5*

	X_1=*1*	X_1=*0*
X_3=*1*	*0.1*	*0.8*
X_3=*0*	***0.9***	*0.2*

	X_2=*1*	X_2=*0*
X_4=*1*	*0.5*	***0.4***
X_4=*0*	*0.5*	*0.6*

	X_2=*1*, X_3=*1*	X_2=*1*, X_3=*0*	X_2=*0*, X_3=*1*	X_2=*0*, X_3=*0*
X_5=*1*	*0.1*	*0.4*	*0.9*	***0.3***
X_5=*0*	*0.9*	*0.6*	*0.1*	*0.7*

The joint probability of sampling $X_1 = 1, X_2 = 0, X_3 = 0, X_4 = 1, X_5 = 1$ *(or* $\{1,0,0,1,1\}$*) can be factorised as follows.*

$$\begin{aligned}\Pr(\{1,0,0,1,1\}) &= \Pr(X_1 = 1)\Pr(X_2 = 0 \mid X_1 = 1)\Pr(X_3 = 0 \mid X_1 = 1)\\ &\qquad \Pr(X_4 = 1 \mid X_2 = 0)\Pr(X_5 = 1 \mid X_2 = 0, X_3 = 0)\\ &= 0.2 \cdot 0.7 \cdot 0.9 \cdot 0.4 \cdot 0.3\\ &= 0.01512.\end{aligned}$$

Algorithm 1: Framework of EDAs

Data : A finite search space $\mathscr{X}$, a parent population size $\mu \in \mathbb{N}$, an offspring population size $\lambda \in \mathbb{N}$, and the oracle-based model of the considered optimisation problem of size n.

```
1  begin
2      t ← 0
3      D_t ← Unif{X}                                  // initial graphical model
4      repeat
5          for i = 1,2,3,...,λ do                     // sample an offspring population
6              sample P_t(i) ~ D_t
7          evaluate λ individuals in P_t              // send λ queries to the oracle
8          Q_t ← SELECTOR(P_t)                        // select a parent population
9          D_{t+1} ← GRAPHBUILDER(Q_t, D_t)           // build a new graphical model
10         t ← t + 1
11     until some terminal condition is fulfilled
```

2.2 A formal framework for EDAs

The EDAs construct in each iteration a new probabilistic graphical model consisting of n nodes, corresponding to the n decision variables, $X_1, X_2, \ldots, X_n \in \{0,1\}$, of the pseudo-Boolean function $f(x)$. Note that there is a one-to-one mapping from the space of probabilistic graphical models to the space of probability distributions, so the two terms, i.e., probabilistic graphical model and probabilistic model, will be used interchangeably in the context of EDAs. Let $\mathscr{D}_t$ denote the graphical model in an iteration $t \in \mathbb{N}$. After iteration t, the algorithms have constructed a sequence of graphical models $(\mathscr{D}_t : t \in \mathbb{N})$.

Algorithm 1 describes a general framework for EDAs considered in this thesis. The starting probabilistic model is the uniform distribution over the search space $\mathscr{X}$, denoted as $\mathrm{Unif}\{\mathscr{X}\}$. The algorithms then follow an iterative process. Given the probabilistic model $\mathscr{D}_t$ in an iteration $t \in \mathbb{N}$, the algorithms sample λ individuals independently and identically from $\mathscr{D}_t$ to form the offspring population $P_t \in \{\mathscr{X}^\lambda\}$. Let $P_t(i)$ denote the i-th individual in such

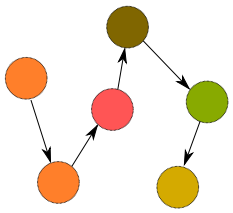

Fig. 2.4 A chain

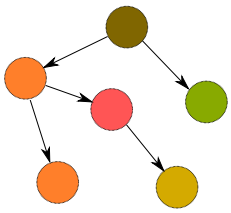

Fig. 2.5 A tree

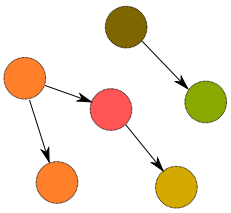

Fig. 2.6 A forest of trees

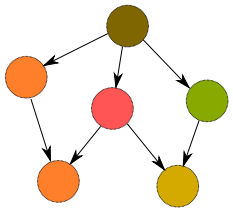

Fig. 2.7 A bayesian network

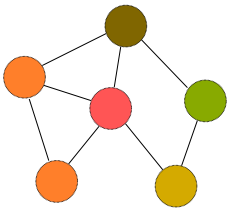

Fig. 2.8 An undirected graph

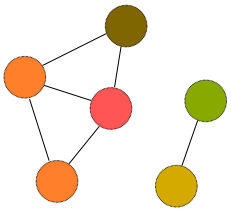

Fig. 2.9 Undirected graphs

population. The algorithms send λ queries in accordance with the λ individuals to the oracle to evaluate their fitness values. These individuals are ranked in descending order of fitness. The algorithm then employs a selection mechanism SELECTOR, e.g., truncation selection, tournament selection, to choose μ out of λ individuals from the offspring population to form the parent population $Q_t \in \{\mathscr{X}^\mu\}$. The GRAPHBUILDER operator then uses the parent population Q_t and the current probabilistic model $\mathscr{D}_t$ to construct a new model $\mathscr{D}_{t+1}$.

Various EDAs have been proposed over the last decades. Each employs a different technique and criterion to create the next graphical model $\mathscr{D}_{t+1}$ (see [73] and references therein). The algorithms halt only if some predefined terminal condition is fulfilled. Some common options are when a user-specified threshold on the number of iterations has exceeded, no progress has been made for a certain number of iterations, or a search point with a given fitness value has been obtained [57, 41, 107].

EDAs are categorised into two classes: univariate and multivariate. The classification is based on the cardinality of the set E of the constructed graphical model $\mathscr{G}(V,E)$. All univariate EDAs have $E = \emptyset$, while multivariate EDAs have $E \neq \emptyset$. In this thesis, we consider only univariate EDAs. However, for completeness, we briefly introduce the class of multivariate EDAs.

2.3 Multivariate EDAs: $E \neq \emptyset$

Because multivariate EDAs can build an arbitrary probabilistic graphical model, the algorithms could learn higher-order interactions between decision variables. Their graphical models could take any shape, e.g., a chain, a tree, a forest. To construct such a complex graph, lots of different probabilities are calculated based on the parent population of μ fittest individuals in order to build their graphical models; some examples are the marginal probability for each variable, entropy, conditional entropy, and mutual information between all pairs of variables [57, 3]. A greedy algorithm is then employed to assemble decision variables to form the graphical model [57]. Some popular multivariate EDAs (and their graphical models) are Mutual Information Maximising Input Clustering (Fig. 2.4 [27]), Bivariate Marginal Distribution Algorithm (Fig. 2.6 [111]), Combining Optimises with Mutual Information Trees (Fig. 2.5 [7]), Bayesian Optimisation Algorithm (Fig. 2.7 [110]), Extended Compact Genetic Algorithm (Fig. 2.9 [55]) and Factorised Distribution Algorithm (Fig. 2.8 [98]).

2.4 Univariate EDAs: $E = \emptyset$ (our focus)

Univariate EDAs have $E = \emptyset$, which means that there is no arcs in their probabilistic graphical models $\mathscr{G}(V,E)$ (where $V = \{X_1, X_2, \ldots, X_n\}$). We have for any $i \in [n]$ that $\mathrm{pa}(X_i) = \emptyset$ and $\Pr(X_i = x_i \mid \mathrm{pa}(X_i)) = \Pr(x_i)$, where $\mathrm{pa}(X_i)$ is the set of observations for parents of the node X_i. In other words, all decision variables are mutually independent. The joint probability distribution is the product of the marginal probabilities of decision variables.

$$\Pr(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} \Pr(x_i).$$

Each X_i takes values in $\{0,1\}$, and the distribution of X_i follows a Bernoulli distribution with some success probability $p_i \in [0,1]$, denoted as $X_i \sim \mathrm{Ber}(p_i)$. The marginal probability distribution of X_i is

$$\Pr(x_i) = (p_i)^{x_i} \cdot (1-p_i)^{1-x_i}.$$

In this situation, the distribution of X_i is fully described by its success probability p_i. Therefore, the probabilistic model $\mathscr{D}_t$ is parameterised by a vector of success probabilities $p_t = (p_{t,1}, p_{t,2}, \ldots, p_{t,n})$. Here, we add the subscript $t \in \mathbb{N}$ to denote the iteration. By

doing this, the probability for an individual $x = (x_1, x_2, \ldots, x_n) \in \mathscr{X}$ to be sampled from the probabilistic model $\mathscr{D}_t$ equals

$$\mathscr{D}_t(x \mid p_t) = \prod_{i=1}^{n} (p_{t,i})^{x_i} (1 - p_{t,i})^{1-x_i}. \tag{2.3}$$

We often call such a probability mass function a product distribution. For simplicity, we often refer to the vector of success probabilities $p_t = (p_{t,1}, p_{t,2}, \ldots, p_{t,n})$ as the probabilistic model in iteration $t \in \mathbb{N}$ in the context of univariate EDAs. As mentioned, the starting model is the uniform distribution, that is, $p_{0,i} = 1/2$ for all $i \in [n]$.

We note that each marginal $p_{t,i}$ for $i \in [n]$ needs to be bounded away from extreme values zero and one; otherwise, the algorithms might never halt. To illustrate this, we consider a univariate EDA optimising some fitness function with a unique global optimum, whose bitstring has a one at bit position i [39]. In some iteration, the algorithm due to a misleading signal sets the i-th marginal to zero, which renders the sampled value at bit position i fixed to 0. Afterwards, the global optimum cannot be sampled, and the algorithm never halts. To overcome this, we have to ensure that any search point in the search space has a positive probability of being sampled in any iteration. This can be achieved by restricting the marginals to be within the interval $[1/n, 1 - 1/n]$, where $n \in \mathbb{N}$ is the problem instance size ($n > 0$). More specifically, if the new value of the marginal (after the update step) drops below $1/n$, the value is adjusted to $1/n$; otherwise, if the value exceeds $1 - 1/n$, it is set to $1 - 1/n$. This procedure is known as capping the marginals. The two values $1/n$ and $1 - 1/n$ are then called the lower and upper borders (or margins), respectively.

Some representatives of the class of univariate EDAs are the Univariate Marginal Distribution Algorithm (UMDA [99]), the compact Genetic Algorithm (cGA [56]) and the Population-Based Incremental Learning Algorithm (PBIL [6]).

To describe these algorithms, we shall introduce further notation. Let $P_t = (x_t^{(1)}, x_t^{(2)}, \ldots, x_t^{(\lambda)})$ denote the offspring population and $\tilde{P}_t = (\tilde{x}_t^{(1)}, \tilde{x}_t^{(2)}, \ldots, \tilde{x}_t^{(\lambda)})$ be the offspring population sorted in descending order according to a fitness function $f(x)$. Let $x_{t,i}^{(j)}$ denote the value sampled at bit position i in the j-th individual in the offspring population P_t (and analogously $\tilde{x}_{t,i}^{(j)}$ for the sorted population $\tilde{P}_t$). Let $X_{t,i} \coloneqq \sum_{j=1}^{\mu} \tilde{x}_t^{(j)}$ be the number of 1s sampled at bit position $i \in [n]$ across the parent population.

2.4.1 compact Genetic Algorithm (cGA)

The cGA, defined in Algorithm 2 [56], operates on two individuals ($\lambda = 2$), which are initially sampled from the uniform distribution (steps 5-7). The algorithm ranks the two individuals in descending order according to a fitness function, where ties are broken uniformly at random (step 8). The probabilistic model is then updated as follows. The marginals stay unchanged at bit positions where the two individuals have the same bit values. Otherwise, a quantity of $1/K$ (where $K > 0$) is shifted towards the direction of the one-bit at each bit position where the two corresponding bits differ (steps 10-15). Afterwards, all marginals are capped to be within the interval $[1/n, 1-1/n]$ (step 16). The parameter K is called the abstract/hypothetical population size.

2.4.2 Univariate Marginal Distribution Algorithm (UMDA)

Unlike the cGA, the UMDA, defined in Algorithm 3 [99], has a larger population of λ individuals (steps 5-7), from which the $\mu \leq \lambda$ fittest individuals, where ties are broken uniformly at random (step 8), are chosen via truncation selection to update the probabilistic model. The algorithm updates each marginal by setting its value to the ratio of the number of 1s sampled at that bit position among the μ fittest individuals to the parent population size μ (also called the frequency of 1s). Afterwards, all marginals are capped to be within the interval $[1/n, 1-1/n]$ (steps 10-11).

2.4.3 Population-Based Incremental Learning (PBIL)

The PBIL, defined in Algorithm 4 [6], is a generalisation of the UMDA. Most operations in the PBIL are similar to those of the UMDA. However, the algorithm takes into account the current values of the marginals when updating the probabilistic model. With a new smoothing parameter $\rho \in (0, 1]$, the probabilistic model is updated using a convex combination between the current values of the marginals and the frequencies of 1s sampled among the μ fittest individuals and then capped to be within the interval $[1/n, 1-1/n]$ (step 11). This update procedure is often known as incremental learning [6]. Note that the UMDA is the PBIL with a maximum smoothing parameter $\rho = 1$. The PBIL can also be seen as a special case of the cross-entropy method [26] on the binary hypercube $\{0, 1\}^n$.

Algorithm 2: compact Genetic Algorithm [56].

Input : An abstract population size $K \in \mathbb{N}$, and an objective function $f : \mathscr{X} \to \mathbb{R}$.

Output : The fittest individual in the population when the terminal condition has been fulfilled.

begin
  $t \leftarrow 0$
  $p_t \leftarrow (1/2, 1/2, \ldots, 1/2)$ `// initial probabilistic model`
  repeat
    for $j = 1, 2$ **do** `// sample two individuals`
      $x_{t,i}^{(j)} \sim \mathrm{Ber}(p_{t,i})$ for all $i \in [n]$
    $P_t \leftarrow (x_t^{(1)}, x_t^{(2)})$ `// offspring population`
    $\tilde{P}_t \leftarrow (\tilde{x}_t^{(1)}, \tilde{x}_t^{(2)}) \leftarrow \textsc{Sort}(P_t)$ `// sort the population`
    for $i = 1, 2, \ldots, n$ **do** `// update the probabilistic model`
      if $\tilde{x}_{t,i}^{(1)} = 1$ *and* $\tilde{x}_{t,i}^{(2)} = 0$ **then**
        $\tilde{p}_{t+1,i} \leftarrow p_{t,i} + 1/K$
      else if $\tilde{x}_{t,i}^{(1)} = 0$ *and* $\tilde{x}_{t,i}^{(2)} = 1$ **then**
        $\tilde{p}_{t+1,i} \leftarrow p_{t,i} - 1/K$
      else
        $\tilde{p}_{t+1,i} \leftarrow p_{t,i}$
      $p_{t+1,i} \leftarrow \max\{1/n, \min\{1 - 1/n, \tilde{p}_{t+1,i}\}\}$ `// cap the marginal`
    $t \leftarrow t + 1$
  until *some terminal condition is fulfilled*

Algorithm 3: Univariate Marginal Distribution Algorithm [99].

Input : An offspring population size $\lambda \in \mathbb{N}$, a parent population size $\mu \in \mathbb{N}$, and an objective function $f : \mathscr{X} \to \mathbb{R}$.

Output : The fittest individual in the population when the terminal condition has been fulfilled.

begin
  $t \leftarrow 0$
  $p_t \leftarrow (1/2, 1/2, \ldots, 1/2)$ `// initial probabilistic model`
  repeat
    for $j = 1, 2, \ldots, \lambda$ **do** `// sample an offspring population`
      $x_{t,i}^{(j)} \sim \mathrm{Ber}(p_{t,i})$ for all $i \in [n]$
    $P_t \leftarrow (x_t^{(1)}, x_t^{(2)}, \ldots, x_t^{(\lambda)})$ `// offspring population`
    $\tilde{P}_t \leftarrow (\tilde{x}_t^{(1)}, \tilde{x}_t^{(2)}, \ldots, \tilde{x}_t^{(\lambda)}) \leftarrow \textsc{Sort}(P_t)$ `// sort the population`
    for $i = 1, 2, \ldots, n$ **do** `// update the probabilistic model`
      $X_{t,i} \leftarrow \sum_{j=1}^{\mu} \tilde{x}_{t,i}^{(j)}$ `// count the number of ones`
      $p_{t+1,i} \leftarrow \max\{1/n, \min\{1 - 1/n, X_{t,i}/\mu\}\}$ `// cap the marginal`
    $t \leftarrow t + 1$
  until *some terminal condition is fulfilled*

Algorithm 4: Population-based Incremental Learning Algorithm [99].

Input : An offspring population size $\lambda \in \mathbb{N}$, a parent population size $\mu \in \mathbb{N}$, a smoothing parameter $\rho \in (0,1]$, and an objective function $f : \mathscr{X} \to \mathbb{R}$.

Output : The fittest individual in the population when the terminal condition has been fulfilled.

begin
  $t \leftarrow 0$
  $p_t \leftarrow (1/2, 1/2, \ldots, 1/2)$ `// initial probabilistic model`
  repeat
    for $j = 1, 2, \ldots, \lambda$ **do** `// sample an offspring population`
      $x_{t,i}^{(j)} \sim \mathrm{Ber}(p_{t,i})$ for all $i \in [n]$
    $P_t \leftarrow (x_t^{(1)}, x_t^{(2)}, \ldots, x_t^{(\lambda)})$ `// offspring population`
    $\tilde{P}_t \leftarrow (\tilde{x}_t^{(1)}, \tilde{x}_t^{(2)}, \ldots, \tilde{x}_t^{(\lambda)}) \leftarrow \textsc{Sort}(P_t)$ `// sort the population`
    for $i = 1, 2, \ldots, n$ **do** `// update the probabilistic model`
      $X_{t,i} \leftarrow \sum_{j=1}^{\mu} \tilde{x}_{t,i}^{(j)}$ `// count the number of ones`
      $p_{t+1,i} \leftarrow \max\{1/n, \min\{1 - 1/n, (1-\rho) p_{t,i} + \rho X_{t,i}/\mu\}\}$ `// update`
    $t \leftarrow t + 1$
  until *some terminal condition is fulfilled*

Chapter 3

Runtime of Univariate Estimation of Distribution Algorithms

In this chapter, we begin with the formal definition for the runtime of the estimation of distribution algorithms followed by three test functions studied in the thesis. We then introduce the two commonly used methods for deriving the runtimes of EDAs. In the end, we provide a summary of the current state-of-the-art runtime bounds for the EDAs on test functions and point out some open questions.

Given a probability space $(\Omega, \mathscr{F}, \Pr)$ and a problem instance size $n \in \mathbb{N}$, we say that an event $A \in \mathscr{F}$ happens with high probability (w.h.p.) if $\Pr(A) \geq 1 - n^{-\varepsilon}$ for some constant $\varepsilon > 0$. Furthermore, we say that the event occurs with overwhelming probability (w.o.p.) if $\Pr(A) \geq 1 - 2^{-\Omega(n^{\varepsilon})}$ [108].

3.1 Definition of Runtime

Recall that in black-box optimisation, we assume that there is an oracle who has knowledge about the fitness function $f(x)$ but refuses to share it with us. We then define the runtime of an EDA on an optimisation problem as the number of queries sent to the oracle, which is equivalent to the number of function evaluations until one of the global optima has been found for the first time. This has become the conventional measure of runtime in the theory community of evolutionary computing. In each iteration, EDAs sample an offspring population of λ individuals and evaluate them by sending λ queries to the oracle, the runtime

is by a factor of λ larger than the number of iterations. Note that if the optimum is present in the middle of an iteration, the algorithms still evaluate the entire population.

Definition 1 (Runtime)**.** *Consider the* EDA *framework in Algorithm 1. Recall that $P_t(i)$ denotes the i-th individual in the population P_t. Let $f_t = \max\{f(P_t(i)) : i \in [\lambda]\}$. The runtime of an* EDA *is formally defined as*

$$T := \min\{t\lambda : f_t = \max\{f(x) : x \in \mathscr{X}\}\}.$$

Considering function evaluations is motivated by the fact that these are often the most expensive operations, assuming that other operations like comparisons and branch instructions can usually be executed very quickly [74]. Although larger population sizes would reduce the number of iterations required [74], this would increase the computational effort within each iteration. Hence, the number of function evaluations seems to be a more appropriate measure for the computational complexity of EDAs. Recall that asymptotic notation is used to describe the runtime (see [19, Chapter 3] for a brief introduction), and all asymptotic statements refer to the problem instance size $n \in \mathbb{N}$.

Given a fixed initialisation, two runs of an EDA on a fixed problem instance will likely require two different numbers of operations due to a high level of randomness involved, and the two outputs are not necessarily identical [108]. In other words, the runtime of an EDA is a random variable, and we are more interested in [108]

1) an expected runtime, e.g., $\mathbb{E}[T] \geq t^*$ for a lower bound or $\mathbb{E}[T] \leq t^*$ for an upper bound,

2) a success probability of the algorithm in t^* steps, e.g., $\Pr(T \leq t^*)$ for an upper bound, or $\Pr(T \geq t^*)$ for a lower bound. We often want bounds that hold with high or overwhelming probability.

Almost any runtime result will be reported in one of these forms, and sometimes one can turn the expected runtime into some bound that holds with high probability or vice versa. See [30, Corollary 6.2] for some examples.

3.2 Three pseudo-Boolean functions for benchmarking

In this thesis, we consider three pseudo-Boolean functions, namely ONEMAX, LEADINGONES, and DLB, which are defined in the search space $\mathscr{X} = \{0,1\}^n$ and have been widely used in runtime analyses of EDAs [39, 23, 77, 71, 127, 133]. These problems have well-understood structures, and each represents a typical characteristic of fitness landscapes and aims at testing a unique ability of the algorithms. They are unimodal functions, and the unique (global) optimum is the all-ones bitstring [108].

The first problem, ONEMAX, as its name may suggest, counts the number of ones in the bitstring and aims at testing the performance of EDAs as a hill climber [129]. Bits in this function have the same contributions to the overall fitness. The optimal solution has a fitness value of n. The function is in the larger class of linear functions and is formally defined as follows.

Definition 2. *For any $x \in \mathscr{X}$,*

$$\textsc{OneMax}(x) := \sum_{i=1}^{n} x_i.$$

In contrast, the LEADINGONES function counts the number of leading ones in the bitstring. Unlike ONEMAX, bits in this function are highly correlated, so it is often used to study the ability of EDAs to cope with epistasis [129], which corresponds to the number of other decision variables a variable depends on [52, 25]. The optimal solution has a fitness value of n.

Definition 3. *For any $x \in \mathscr{X}$,*

$$\textsc{LeadingOnes}(x) := \sum_{i=1}^{n} \prod_{j=1}^{i} x_j.$$

Recall that the PLATEAU(a,b) function, defined in (1.1), which has an epistasis level of $b-1$ because a bit in each of a non-overlapping blocks of length b has to cooperate with the other $b-1$ bits in that block to increase the overall fitness value by one. When $b = 1$, we obtain the ONEMAX function, which has an epistasis level of zero (as $b-1=0$). The LEADINGONES function is special in the sense that the bitstring is partitioned into overlapping blocks. A bit $i \in [n]$ depends on all preceding bits, and consequently, the last

bit position relies on the $n-1$ other bits. For that reason, we say that the LEADINGONES function has an epistasis level of $n-1$.

Finally, the DLB function has been introduced very recently by the author to study the behaviour of EDAs when deception is present [78]. Most notably, the problem has an epistasis level of $n-1$ and is mildly deceptive. The bitstring is partitioned into independent blocks of $w \geq 2$ consecutive bits. The problem's difficulty is determined by the width w of each block, which can be altered to increase the deception level. Here, we consider the smallest width of $w = 2$. DLB is similar to the LEADINGONES function in the sense that the fitness depends on the number of leading 11s. However, it attempts to deceive the algorithm by assigning TRAP-like weights to different leftmost non-11 block settings, which we call the active block. Reaching a unique global optimum requires overcoming many mild traps. Each non-overlapping leading block of two consecutive ones (leading 11s, in short) contributes a value of two, while a value of one is awarded if the active block is a 00, and no reward is given for any other block settings. The all-ones bitstring is the optimal solution with a fitness of $(n/2) \cdot 2 = n$ since there are $n/2$ blocks, assuming that n is a multiple of $w = 2$.

Definition 4. *Let n be an even positive integer. For any* $x \in \mathcal{X}$

$$\mathrm{DLB}(x) \coloneqq \begin{cases} n & \text{if } \phi(x) = n/2, \\ 2 \cdot \phi(x) + 1 & \text{if } x_{2\phi(x)+1} + x_{2\phi(x)+2} = 0, \\ 2 \cdot \phi(x) & \text{if } x_{2\phi(x)+1} + x_{2\phi(x)+2} = 1, \end{cases}$$

where

$$\phi(x) \coloneqq \sum_{i=1}^{n/2} \prod_{j=1}^{2i} x_j. \tag{3.1}$$

The auxiliary function $\phi(x)$ counts the number of leading 11s, which is identical to the LOB_2 function previously defined in [64, Definition 13].

3.3 Methods for driving runtimes

3.3.1 Drift theorems

Drift theorems are powerful tools to estimate the expected runtime and tail bounds on the runtime of EDAs [119, 71, 127, 128, 46, 39]. They were brought to the theory community of evolutionary computing by He and Yao [58] and were built around the seminal paper due to Hajek [54]. Almost any drift theorem assumes the following setting.

Definition 5 (Main Setup)**.** *Let $(X_t : t \in \mathbb{N})$ be a stochastic process, adapted to some filtration $(\mathscr{F}_t : t \in \mathbb{N})$, over some state space $\mathscr{X}$. Let $\mathscr{X}_{opt}$ denote the set of all optimal states of the process, in which one of the global optima has been detected. We also assume that we have a real-valued distance/potential/trace function $\phi : \mathscr{X} \to \mathbb{S} \subseteq \mathbb{R}$ [86, 31] satisfying*

$$\phi(X) = \begin{cases} 0 & \text{if } X \in \mathscr{X}_{opt}, \\ > 0 & \text{otherwise.} \end{cases}$$

Our target is the optimal states $\mathscr{X}_{opt}$, which are reached at time t if $\phi(X_t) = 0$. We define a random variable $\tilde{T}$ as follows

$$\tilde{T} = \min\{t \in \mathbb{N} : \phi(X_t) = 0\}.$$

The random variable $\tilde{T}$ denotes the number of iterations until one of the global optima has been found for the first time. By [95, Definition 12.2], it is a stopping time since the event $\{\tilde{T} = n\}$ depends only on the values of the random variables $X_0, X_1, \ldots, X_n$. Because EDAs rely on a population of λ individuals in each iteration, the runtime T, defined in Definition 1, is larger than $\tilde{T}$ by a factor of λ, that is, $T = \lambda \cdot \tilde{T}$.

There are various choices for the potential functions. Take a univariate EDA, which represents the probabilistic model as a vector of success probabilities $p_t = (p_{t,1}, p_{t,2}, \ldots, p_{t,n})$, as an example. The probability vector in which all marginals $p_{t,i}$'s are at the upper border $1 - 1/n$ is often called the optimal distribution. One might define $X_t = \sum_{i=1}^{n} p_{t,i}$ to be the state of the algorithm in iteration t. The optimal state of the algorithms is $X_t = n(1 - 1/n) = n - 1$. Thus, one choice for the potential function is $\phi(X_t) = n - 1 - X_t$, that is, the distance between the current probabilistic model to the optimal model. We note that this potential function has

been previously used for the UMDA [71, 127] and the cGA [128]. See [119, 39] for other choices.

Drift theorems provide bounds on the conditional expectation $\mathbb{E}\left[\tilde{T} \mid \mathscr{F}_0\right]$ by turning the drift in a single iteration into the overall progression. At time $t < \tilde{T}$, the distance from the current state X_t to the optimal state is $\phi(X_t) > 0$ by assumption in Definition 5. The expected single-step change is defined as

$$\mathbb{E}\left[\phi(X_t) - \phi(X_{t+1}) \mid \mathscr{F}_t\right] = \phi(X_t) - \mathbb{E}\left[\phi(X_{t+1}) \mid \mathscr{F}_t\right]$$

since X_t is $\mathscr{F}_t$-measurable [125, p. 93]. We speak of an additive drift when there exists a value $\delta > 0$, such that

$$\phi(X_t) - \mathbb{E}\left[\phi(X_{t+1}) \mid \mathscr{F}_t\right] \geq \delta, \tag{3.2}$$

or

$$\phi(X_t) - \mathbb{E}\left[\phi(X_{t+1}) \mid \mathscr{F}_t\right] \leq \delta. \tag{3.3}$$

Given a starting state X_0 (which is $\mathscr{F}_0$-measurable), $\phi(X_0)$ is the initial distance value. From (3.2), we obtain an upper bound on the expected runtime as follows.

$$\mathbb{E}\left[\tilde{T} \mid \mathscr{F}_0\right] \leq \frac{\phi(X_0)}{\delta},$$

while from (3.3), a lower bound on the expected runtime is guaranteed as follows.

$$\mathbb{E}\left[\tilde{T} \mid \mathscr{F}_0\right] \geq \frac{\phi(X_0)}{\delta}.$$

Note that we are only interested in the stochastic process $(X_t : t \in \mathbb{N})$ until the time point $\tilde{T}$. Hence, we require all conditions to hold for any $t < \tilde{T}$, and we need to multiply each of the conditions above by the indicator function $\mathbb{1}_{\{t<\tilde{T}\}}$, which is itself a $\mathscr{F}_t$-measurable random variable [70]. We now present the additive drift theorem [20].

Theorem 1 (Additive drift)**.** *Consider the setup in Definition 5 for* $\mathbb{S} \subseteq [0,\infty)$*. We define for some* $\delta > 0$ *and a constant* $0 < b < \infty$ *the conditions*

1.1) $\mathbb{E}\left[(\phi(X_t) - \phi(X_{t+1})) \cdot \mathbb{1}_{\{t<\tilde{T}\}} \mid \mathscr{F}_t\right] \geq \delta \cdot \mathbb{1}_{\{t<\tilde{T}\}}$ *for all* $t \in \mathbb{N}$.

1.2) $\mathbb{E}\left[(\phi(X_t) - \phi(X_{t+1})) \cdot \mathbb{1}_{\{t<\tilde{T}\}} \mid \mathscr{F}_t\right] \leq \delta \cdot \mathbb{1}_{\{t<\tilde{T}\}}$ *for all* $t \in \mathbb{N}$.

2) $\phi(X_t) \cdot \mathbb{1}_{\{t<\tilde{T}\}} < b \cdot \mathbb{1}_{\{t<\tilde{T}\}}$ *for all* $t \in \mathbb{N}$, *and*

3) $\mathbb{E}\left[\tilde{T}\right] < \infty$.

If 1.1), 2), and 3) hold, then

$$\mathbb{E}\left[\tilde{T} \mid \mathscr{F}_0\right] \leq \frac{\phi(X_0)}{\delta}. \tag{3.4}$$

if 1.2), 2), and 3) hold, then

$$\mathbb{E}\left[\tilde{T} \mid \mathscr{F}_0\right] \geq \frac{\phi(X_0)}{\delta}. \tag{3.5}$$

Sometimes, the expected single-step change can be bounded from below by a multiple of the current distance value $\phi(X_t)$, that is, there exists some $\delta > 0$ such that

$$\phi(X_t) - \mathbb{E}\left[\phi(X_{t+1}) \mid \mathscr{F}_t\right] \geq \delta \cdot \phi(X_t).$$

In this case, an upper bound on $\mathbb{E}\left[\tilde{T} \mid \mathscr{F}_0\right]$ can be obtained via a so-called multiplicative drift theorem [31].

Theorem 2 (Multiplicative drift)**.** *Consider the setup in Definition 5 for* $\mathbb{S} \subseteq \{0\} \cup [1,\infty)$. *Assume that there exists some* $\delta > 0$ *such that for all* $t \geq 0$ *we have*

$$\mathbb{E}\left[(\phi(X_t) - \phi(X_{t+1})) \cdot \mathbb{1}_{\{t<\tilde{T}\}} \mid \phi(X_t) > 0, \mathscr{F}_t\right] \geq \delta \cdot \phi(X_t) \cdot \mathbb{1}_{\{t<\tilde{T}\}}$$

Then

$$\mathbb{E}\left[\tilde{T} \mid \mathscr{F}_0\right] \leq \frac{1 + \log \phi(X_0)}{\delta}.$$

For completeness, we also introduce the so-called variable drift theorem [92, 66, 8].

Theorem 3 (Variable drift)**.** *Consider the setup in Definition 5 for* $\mathbb{S} \subseteq \{0\} \cup [1,\infty)$ *and* $\phi(X_0) = n$. *Suppose that there is a positive, increasing function* $h : [1,\infty) \to \mathbb{R}_{>0}$ *such that* $1/h$ *is integrable and for all* $x \in \mathbb{S}$ *and* $x > 0$ *we have for all* $t \geq 0$

$$\mathbb{E}\left[(\phi(X_t) - \phi(X_{t+1})) \cdot \mathbb{1}_{\{t<\tilde{T}\}} \mid \phi(X_t) = x, \mathscr{F}_t\right] \geq h(x) \cdot \mathbb{1}_{\{t<\tilde{T}\}}.$$

Then

$$\mathbb{E}\left[\tilde{T} \mid \mathscr{F}_0\right] \leq \frac{1}{h(1)} + \int_1^n \frac{du}{h(u)}.$$

See [85, 83] for more variants of drift theorems.

3.3.2 Level-based theorem

It is not easy to define a good potential function for population-based algorithms. Instead of using drift analysis directly, one can use the level-based theorem, which is a general tool that provides upper bounds on the expected runtime of population-based algorithms on a wide range of optimisation problems. It has successfully yielded the runtimes of the Genetic Algorithms with or without crossover on the class of linear functions and LEADINGONES [20]. Moreover, runtime bounds for the UMDA and the PBIL on the ONEMAX and LEADINGONES problems have been derived using this method [23, 76, 24, 77, 79, 78].

Algorithm 5: Population-based algorithm [20]

Data : A finite search space $\mathscr{X}$, a population size $\lambda \in \mathbb{N}$, an initial population $P_0 \in \mathscr{X}^\lambda$, and a mapping $\mathscr{D}^*$ from the space of populations $\mathscr{X}^\lambda$ to the space of probability distributions over $\mathscr{X}$.

```
1 begin
2     t ← 0
3     repeat
4         for i = 1,2,3,...,λ do
5             sample P_{t+1}(i) ~ D*(P_t)
6         t ← t + 1
7     until some termination condition is fulfilled
```

In essence, the theorem assumes that the studied algorithm can be cast into the framework of Algorithm 5 [20], where $P_t(i)$ denotes the i-th individual in the current population P_t. In an iteration $t \in \mathbb{N}$, the algorithm operates on a population P_t of λ individuals. The theorem is very general in the sense that it makes no assumptions on the fitness functions, selection mechanisms, or generic operators. Rather, it assumes the existence of a mapping $\mathscr{D}^*$ from the space of populations $\mathscr{X}^\lambda$ to the space of the probability distribution over the search space. In an iteration t, the mapping $\mathscr{D}^*$ depends only on the population P_t and is involved in the production of a new population for the next iteration [20]. Furthermore, it also assumes that the search space $\mathscr{X}$ can be partitioned into m disjoint non-empty subsets $A_1, A_2, \ldots, A_m$, where a subset is called a level, and the last level A_m consists of optimal solutions only. One example of such a level definition is the canonical fitness-based partition.

Definition 6 (Canonical partition [118])**.** *For two sets $A, B \subseteq \mathscr{X}$ and the fitness function f, let $A <_f B$ if $f(a) < f(b)$ for all $a \in A$ and all $b \in B$. A canonical fitness-based partition of the search space $\mathscr{X}$ are the disjoint non-empty sets $A_1, A_2, \ldots, A_m$ such that $A_1 <_f A_2 <_f \cdots <_f A_m$ and A_m only contains global optima.*

Based on the definition above, we now let $A_{\geq j} \coloneqq \cup_{i=j}^{m} A_i$ be the set of all individuals in level A_j or higher. Also, let $|P_t \cap A_{\geq j}| \coloneqq |\{i \mid P_t(i) \in A_{\geq j}\}|$ be the number of individuals of the population P_t in levels $A_{\geq j}$. The level-based theorem can be formally stated as follows [20, Theorem 1].

Theorem 4 (Level-based theorem)**.** *Given a partition $A_1, A_2, \ldots, A_m$ of $\mathscr{X}$, define $T \coloneqq \min\{t\lambda \mid |P_t \cap A_m| > 0\}$ to be the first time t that at least one element of level A_m appears in the current population P_t. If there exist $z_1, \ldots, z_{m-1}, \delta \in (0,1]$, and $\gamma_0 \in (0,1)$ such that for any population $P_t \in \mathscr{X}^\lambda$,*

(G1) *for each level $j \in [m-1]$, if $|P_t \cap A_{\geq j}| \geq \gamma_0 \lambda$ then*

$$\Pr_{y \sim \mathscr{D}(P_t)} \left(y \in A_{\geq j+1}\right) \geq z_j.$$

(G2) *for each level $j \in [m-2]$, and all $\gamma \in (0, \gamma_0]$, if $|P_t \cap A_{\geq j}| \geq \gamma_0 \lambda$ and $|P_t \cap A_{\geq j+1}| \geq \gamma \lambda$ then*

$$\Pr_{y \sim \mathscr{D}(P_t)} \left(y \in A_{\geq j+1}\right) \geq (1+\delta)\,\gamma.$$

(G3) *and the population size $\lambda \in \mathbb{N}$ satisfies*

$$\lambda \geq \left(\frac{4}{\gamma_0 \delta^2}\right) \ln\left(\frac{128m}{z_* \delta^2}\right)$$

where $z_* \coloneqq \min_{j \in [m-1]}\{z_j\}$, then

$$\mathbb{E}[T] \leq \left(\frac{8}{\delta^2}\right) \sum_{j=1}^{m-1} \left[\lambda \ln\left(\frac{6\delta\lambda}{4 + z_j \delta \lambda}\right) + \frac{1}{z_j}\right].$$

For interested readers, the proof of the level-based theorem can be seen in [20]. The application of the theorem requires the verification of three conditions (G1), (G2) and (G3) before an upper bound on the runtime can be guaranteed.

We remark here that the UMDA (see Algorithm 3) and the cGA (see Algorithm 2) can be cast into Algorithm 5 since the two algorithms construct the probabilistic graphical model using the parent population only. More specifically, we can describe the two algorithms as $P_{t+1}(i) \sim \textsc{GraphBuilder} \circ \textsc{Selector}(P_t) = \mathscr{D}^*(P_t)$. Although the cGA can be cast into Algorithm 5, applying the level-based theorem to derive runtime bounds for the cGA seems infeasible because the algorithm uses a population of two individuals only, i.e., $\lambda = 2$, while the condition (G3) in Theorem 4 often requires $\lambda = \Omega(\log n)$. For the PBIL (see Algorithm 4), the algorithm does not fit into the framework in Algorithm 5 since it takes the current probabilistic model into account when building the next graphical model (via a convex combination with a smoothing parameter $\rho \in (0,1]$). However, in Chapter 5 we shall show that as long as the offspring population size λ is chosen sufficiently large, the level-based theorem is still applicable for the PBIL.

3.4 Related work

There had only been a handful of rigorous runtime analyses of EDAs by 2015. Recently EDAs have drawn growing attention from the theory community of evolutionary computing [23, 47, 76, 127, 133, 71, 128, 77, 32, 87]. The theoretical analyses in general aim to gain insights on the performances of the algorithms when optimising some test function with a well-understood structure. Recall that we are interested in the runtime that is the number of function evaluations performed by the algorithm until an optimal solution has been found for the first time (see Definition 1).

Droste [39] performed the first rigorous runtime analysis for the cGA (see Algorithm 2) with an abstract population size $K = n^{1/2+\varepsilon}$ for some constant $\varepsilon \in (0,1)$ and obtained a lower bound of $\Omega(K\sqrt{n}) = \Omega(n^{1+\varepsilon})$ on the expected runtime of the cGA on any pseudo-Boolean function. He also proved a tight bound of $\Theta(K\sqrt{n}) = \Theta(n^{1+\varepsilon})$ on the OneMax function (see Definition 2) and, for $K = n^{1+\varepsilon}$, a more general upper bound of $O(nK) = O\left(n^{2+\varepsilon}\right)$ on the class of linear functions. Note that each marginal of the cGA considered is allowed to reach the extreme values zero and one. As discussed, an EDA without margins can prematurely converge to a sub-optimal solution [4]; thus, the runtime bounds of [39] were in fact conditioned on the event that early convergence never happens. As the considered abstract population size is not too small, i.e., $K = n^{1/2+\varepsilon}$, early convergence does not occur during the course of optimisation with probability super-polynomially close to one [39].

More recently, Lengler et al. [87] considered the cGA with $K = \mathrm{O}\left(\sqrt{n}/\log^2 n\right)$, which was not covered by Droste in [39], and obtained a lower bound of $\Omega(K^{1/3}n + n\log n)$ on the expected runtime of the cGA on the OneMax function. Note that if $K = \Theta(\sqrt{n}/\log^2 n)$, the lower bound above will be $\Omega(n^{7/6}/\log^2 n)$, which further tightens the bounds on the expected runtime of the cGA.

Friedrich et al. [47] pointed out two behavioural properties of univariate EDAs at each bit position: a balanced EDA would be sensitive to signals in the fitness, while a stable one would remain neutral under a biasless fitness function. Unfortunately, many univariate EDAs, including the cGA (see Algorithm 2), the UMDA (see Algorithm 3), the PBIL (see Algorithm 4) and some related algorithms are balanced but not stable [47]. Thus, it is conjectured that these algorithms might take a runtime of at least $\Omega(n^2)$ to optimise the LeadingOnes function (see Definition 3). For this particular function, most of the 'active' bits are neutral during the optimisation, and since the algorithms are not stable, the marginals of these neural bits vary considerably due to large variances and very quickly reach some small value. Once this has happened, the algorithms will require a significant amount of time to reverse. A more stable version of the cGA – the so-called stable cGA (or scGA) – was then proposed in [47]. In the scGA, the marginals of neutral bits are kept close to the initial value of $1/2$. Under appropriate settings, a neutral bit when observing 'correct' fitness signals will reach the upper border within an expected time of $\Theta(\log n)$ [47]. Since there are $\Theta(n)$ such neutral bits, the scGA takes a runtime of $\Theta(n\log n)$ to optimise the LeadingOnes function with a success probability polynomially close to one [47]. Furthermore, a recent study by Friedrich et al. [46] showed that the cGA copes with higher levels of Gaussian posterior noise more efficiently than mutation-only heuristics do. More specifically, the algorithm takes at most $\mathrm{O}\left(\sigma^4 n\log^2 n\right)$ function evaluations with high probability to optimise the OneMax function under a posterior noise model, in which noise follows a zero-mean normal distribution with a variance of σ^2 [46]. Very recently, Doerr & Neumann [35] pointed out that the robustness of EDAs to noise is an open problem, and theoretical analyses in this topic would be highly desirable.

Like the cGA, our understandings of the UMDA in terms of runtime on test functions are also sparse and limited. The algorithm was early analysed in a series of papers [14–17], where time complexities of the UMDA on simple unimodal functions were derived. These results showed that the UMDA with margins can avoid early convergence and consequently often outperforms the UMDA without margins. Shapiro [116] investigated the UMDA

with a different selection mechanism rather than truncation selection. In particular, this variant of the UMDA updates the probabilistic model using individuals whose fitness values are as good as the mean fitness of all individuals in the current offspring population. By representing the trajectory of the algorithm as a Markov chain [44], he showed that the population has to be of size at least $\sqrt{n}$ to prevent the probabilistic model from quickly converging to the corner of the hypercube on ONEMAX. This phenomenon is well-known as genetic drift [4, 36]. A decade later, the first upper bound on the expected runtime of the UMDA on ONEMAX was shown in [23]. Working on the standard UMDA using truncation selection, Dang and Lehre [23] via the level-based theorem (see Theorem 4) proved an upper bound of $O(n\lambda \log \lambda)$ on the expected runtime of the UMDA on ONEMAX, assuming a population size $\lambda = \Omega(\log n)$. If $\lambda = \Theta(\log n)$, then the upper bound is $O(n \log n \log\log n)$. Meanwhile, Krejca and Witt [71] obtained a lower bound of $\Omega(\mu\sqrt{n} + n\log n)$ for the UMDA on ONEMAX via drift analysis, where $\lambda = (1+\Theta(1))\mu$. Prior to this work, there was a gap of $\Theta(\log\log n)$ between the best known upper bound of $O(n \log n \log\log n)$ [23] and the lower bound of $\Omega(n\log n)$ [71] on the expected runtime of the UMDA on the ONEMAX function.

For the LEADINGONES function, Dang & Lehre [23] via the level-based theorem showed an upper bound of $O\left(n\lambda \log \lambda + n^2\right)$ for the UMDA with population sizes $\lambda = \Omega(\log n)$ and $\mu \leq \lambda/(e(1+\delta))$ for some constant $\delta \in (0,1)$. For $\lambda = \Omega(\log n) \cap O(n/\log n)$, an expected runtime of $O\left(n^2\right)$ is guaranteed. We are still missing a lower bound on the LEADINGONES function. If the conjectured lower bound of $\Omega(n^2)$ in [47] were confirmed, a tight bound of $\Theta(n^2)$ would be obtained under appropriate parameter settings. Furthermore, we do not know the runtime of the UMDA with a selection rate $\mu/\lambda \geq 1/e$ on the LEADINGONES function.

Wu et al. [133] presented the first and only runtime analysis for the PBIL with a smoothing parameter $\rho \in O(1)$ on the ONEMAX and LEADINGONES functions. The authors argued that due to the use of sufficiently large population size, it is possible to prevent the marginals from reaching the lower border early (i.e., genetic drift) even when a large smoothing parameter is used. The authors showed an upper bound of $O(n^{3/2+c})$ on the ONEMAX function and an upper bound of $O\left(n^{2+c}\right)$ on the LEADINGONES function when a large population size is required, that is, $\lambda = \Omega(n^{1+c})$ for some constant $c \in (0,1)$. Despite being a generalisation of the UMDA, there is still a gap of $\Theta(n^{1/2+c}/\log n)$ between the best known upper bound for the PBIL and the UMDA on the ONEMAX function. Furthermore, there exists another gap of $\Theta(n^c)$ on the LEADINGONES function. It is of particular interest

if these gaps could be closed since that would show whether the PBIL's incremental learning update method brings any benefit compared to the update mechanism employed by the UMDA.

Table 3.1 summarises the latest results (including those presented in this thesis) about the runtime analyses of univariate EDAs on simple benchmark problems; see [129, 35] for recent progress in the theory of EDAs for discrete optimisation.

Table 3.1 Latest expected runtime bounds for univariate EDAs on some benchmark functions.

Problem	Algorithm	Constraints	Expected Runtime	References
OneMax	UMDA	$\lambda = (1+\Theta(1))\mu, \lambda = \text{poly}(n)$	$\Omega(\lambda\sqrt{n}+n\log n)$	[71]
		$\lambda = (1+\Theta(1))\mu,\ \mu = \Omega(\log n)\cap o(n)$	$O(\lambda n)$	[127]
		$\lambda = (1+\Theta(1))\mu,\ \mu = \Omega(\sqrt{n}\log n)$	$O(\lambda\sqrt{n})$	[127]
		$\Omega(\log n) \ni \mu \leq \sqrt{n(1-c)}, \lambda \geq (13e/(1-c))\mu, c \in (0,1)$	$O(\lambda n)$	[Theorem 9]
		$\mu = \Omega(\sqrt{n}\log n), \lambda \geq 294(1+\delta)\mu, \delta > 0$	$O(\lambda\sqrt{n})$	[Theorem 10]
	PBIL *	$\mu = \omega(n), \lambda = \omega(\mu)$	$\omega(n^{3/2})$	[133]
	cGA	$K = n^{1/2+\varepsilon}$	$\Theta(K\sqrt{n})$	[39]
		$K = O\left(\sqrt{n}/\log^2 n\right)$	$\Omega(K^{1/3}n+n\log n)$	[87]
	scGA	$\rho = \Omega(1/\log n), a = \Theta(\rho), c > 0$	$\Omega(\min\{2^{\Theta(n)}, 2^{c/\rho}\})$	[32]
LeadingOnes	UMDA	$\mu/\lambda \leq 1/(e(1+\delta)), \lambda = \Omega(\log n), \delta \in (0,1)$	$O\left(n\lambda\log\lambda+n^2\right)$	[23]
		$\mu/\lambda \leq (1-\delta)^2/e, \delta \in (0,1)$	$\Omega(n\lambda/\log\lambda)$	[Theorem 16]
		$\mu = o(n^{1/k}), k \in [0,1), \mu \leq \lambda \leq e^{1-\varepsilon}\mu/(1+\delta), \varepsilon \in (0,1), 0 < \delta \leq e^{1-\varepsilon}-1$	$2^{\Omega(\mu^k)}$	[Theorem 14]
		prior noise, $O(1) \ni p \in (0,1), \lambda \geq \min\{c\log n, e\mu^2(1+\delta)\}, c > 0, \delta \in (0,1)$	$O\left(n\lambda\log\lambda+n^2\right)$	[Theorem 17]
		$\lambda \geq \mu, \mu = \Omega(n\log n)$	$\Theta(n\lambda/\log(\lambda/\mu))$	[34]
	PBIL	$\lambda = n^{1+c}, \mu = O\left(n^{c/2}\right), c \in (0,1), O(1) \ni \rho \in (0,1]$	$O\left(n^{2+c}\right)$	[133]
		$\lambda = \Omega(\log n), O(1) \ni \rho \in (1/e,1], O(1) \ni \mu/\lambda \leq c(\rho) < 1$	$O\left(n\lambda\log\lambda+n^2\right)$ **	[Theorem 18]
		prior noise, $O(1) \ni \rho \in (0,1), O(1) \ni \mu/\lambda = c(\rho,p), \lambda = \Omega(\log n)$	$O\left(n\lambda\log\lambda+n^2\right)$ **	[Theorem 19]
	scGA	$\rho = \Theta(1/\log n), a = O(\rho)$	$O(n\log n)$	[47]
DLB	UMDA	$\mu = \Omega(\log n), \lambda = \Omega(\mu^2)$	$O\left(n\lambda\log\lambda+n^2\right)$	[Theorem 25]
		$\mu \leq (n/8000)+1, \mu \leq \lambda \leq O\left(n^{2-k}\right), k \in (0,1]$	$2^{\Omega(\mu^k)}$	[Theorem 24]
		$\mu = \Omega(n\log n), \lambda = \Omega(\mu)$	$O(n\lambda)$	[33]

* without margins

** assuming that all marginals are lower bounded by some constant during the course of the optimisation

Chapter 4

UMDA can cope with linearity as efficiently as other EAs do

This chapter is based on the two joint works: [76] with Per Kristian Lehre, and [24] with Duc-Cuong Dang and Per Kristian Lehre. The proof for small population sizes is similar to that in the conference paper [76]. However, the proof for large population sizes in the journal version [24, Theorem 9] has been improved by stating a more precise constraint on the offspring population size λ.

4.1 Open problems

Despite an increasing momentum in the runtime analysis of EDAs over the last few years [14–17], our understanding of the UMDA is still limited. Motivated by this, the current chapter aims at examining the hill-climbing ability of the algorithm by deriving upper bounds on the expected runtime on the ONEMAX function. We note that the function has a fitness landscape resembling a hill, in which bits have equal contributions to the overall fitness [129], and flipping any 0-bit in the bitstring will move it closer to the all-ones bitstring, i.e., the global optimum.

Recall that Dang and Lehre [23] proved an upper bound of $O(n\lambda \log \lambda)$ on the expected runtime of the UMDA on ONEMAX, assuming an offspring population size $\lambda = \Omega(\log n)$. If $\lambda = \Theta(\log n)$, then the upper bound is $O(n \log n \log\log n)$. Recently, Krejca and Witt [71] obtained a lower bound of $\Omega(\mu\sqrt{n} + n\log n)$ for the UMDA on ONEMAX via drift analysis,

assuming that $\lambda = (1+\Theta(1))\mu$. From these results, we observe that the best known upper and lower bounds on the expected runtime for the UMDA on the ONEMAX function still differ by a factor of $\Theta(\log\log n)$. This raises the question of whether the gap could be closed and a tight bound of $\Theta(n\log n)$ might be obtained. The answer to this question is of special interest as the UMDA would take the same asymptotic runtime as the mutation-based EAs to optimise the ONEMAX function [40] despite the fundamental differences in their working principles.

In this chapter, we give in Theorem 4 a more detailed analysis so that an expected runtime of $O(n\log n)$ for the UMDA on the ONEMAX function is derived. This significantly improves the results in [23, 20] and matches the recent lower bound in [71] as well as the performance of the $(1+1)$ EA [40]. More specifically, we choose the two population sizes λ and μ such that $\lambda \geq b\mu$ for a sufficiently large constant $b > 0$, and separate two regimes of small and large parent population sizes. An upper bound of $O(\lambda n)$ is obtained for $\mu = \Omega(\log n) \cap O(\sqrt{n})$, while an upper bound of $O(\lambda\sqrt{n})$ is guaranteed for parent population sizes $\mu = \Omega(\sqrt{n}\log n)$.

Related independent work: Witt [127] (journal version [130]) independently obtained the upper bounds of $O(\lambda n)$ and $O(\lambda\sqrt{n})$ on the expected runtime of the UMDA on ONEMAX for $\mu = \Omega(\log n) \cap o(n)$ and $\mu = \Omega(\sqrt{n}\log n)$, respectively, and $\lambda = \Theta(\mu)$ using an involved drift analysis. While our results do not hold for $\mu = \omega(\sqrt{n}) \cap o(\sqrt{n}\log n)$, our methods yield significantly easier proofs. Furthermore, our analysis also holds when the parent population size μ is not proportional to the offspring population size λ, which is not covered in [127].

4.2 Useful tools

Let $p_t = (p_{t,1}, p_{t,2}, \ldots, p_{t,n})$ be the marginal probabilities in line 3 of Algorithm 3. Let $Y_{t,1}, Y_{t,2}, \ldots, Y_{t,n}$ be n random variables representing the values at n bit positions of an individual obtained by sampling the probability distribution in (2.3) in an iteration $t \in \mathbb{N}$. Let $Y_{t,i:j} := \sum_{k=i}^{j} Y_{t,k}$. We also need the following results in the main proofs.

Let $\lceil\cdot\rceil$ and $\lfloor\cdot\rfloor$ denote the ceiling and floor functions, respectively. We note that individuals in the class of univariate EDAs are produced by the independent sampling of n random variables, $X_1, X_2, \ldots, X_n$, where $X_i \sim \mathrm{Ber}(p_i)$. The sum of these random variables, denoted as $X = \sum_{i=1}^{n} X_i \sim \mathrm{PB}(p_1, p_2, \ldots, p_n)$, is of special interest. The following result shows that

if all success probabilities $p_1, p_2, \ldots, p_n$ are at least some constant, then a deviation by an additive term of $\Theta(\sqrt{n - \lfloor \mathbb{E}[X] \rfloor})$ above the expectation of the sum X occurs with constant probability.

Theorem 5 (Lemma 3 [133]). *Let $X \sim \mathrm{PB}(p_1, p_2, \ldots, p_n)$. If there exists a constant $p_{\min} > 0$ such that $p_i \geq p_{\min}$ for all $i \in [n]$, then*

$$\Pr\left(X \geq \min\left\{\mathbb{E}[X] + (1/p_{\min})\sqrt{n - \lfloor \mathbb{E}[X] \rfloor}, n\right\}\right) \geq \kappa,$$

for some positive constant κ, independent of n.

Theorem 6 (Theorem 3.2 [65]). *Let $X \sim \mathrm{PB}(p_1, p_2, \ldots, p_n)$. If $\mathbb{E}[X]$ is an integer, then*

$$\Pr(X \geq \mathbb{E}[X]) \geq \frac{1}{2}.$$

Theorem 7 (Feige's inequality [42]). *Let $X \sim \mathrm{PB}(p_1, p_2, \ldots, p_n)$. It holds for any $\delta > 0$ that*

$$\Pr(X > \mathbb{E}[X] - \delta) \geq \min\left\{\frac{1}{13}, \frac{\delta}{1+\delta}\right\}.$$

By considering the complementary event, Theorem 7 can be rewritten in the form that provides an upper bound on the lower tail probability $\Pr(X \leq \mathbb{E}[X] - \delta)$.

All results above (also called concentration inequalities) imply that the sampled value for the random variable X will be close to its mean. However, the next theorem that is called an anti-concentration inequality implies that the probability mass cannot be too concentrated at any point.

Theorem 8 (Anti-concentration bound [5]). *Let $X \sim \mathrm{PB}(p_1, p_2, \ldots, p_n)$. Let $\mathrm{Var}[X] = \sum_{i=1}^{n} p_i(1-p_i)$. It holds for any $x \in \{0\} \cup [n]$ that*

$$\Pr(X = x) \leq \frac{\eta}{\sqrt{\mathrm{Var}[X]}},$$

where η is an absolute constant and $\eta < 0.4689$.

A special case of this result where all success probabilities are between $1/6$ and $5/6$ was proven in [119], which was then extended in [71] to the case when there is only a linear

number of p_i's in that interval. More recently, Doerr [30] has given a proof for the result with $\eta = 2$. Despite the simplicity of the new proof, the value $\eta = 2$ is still looser compared to the constant in Theorem 8. We note that this is the first time the result in Theorem 8 has been used in runtime analyses of randomised search heuristics. It will play an essential role in our analyses in Chapter 4 and Chapter 6.

So far, we have considered only one probability distribution. Now we shall examine two different Poisson binomial distributions.

Definition 7 (Majorisation [49])**.** *Given two vectors $p := (p_1, \ldots, p_n)$ and $q := (q_1, \ldots, q_n)$, where $p_1 \geq p_2 \geq \ldots \geq p_n$ and analogously for the q_i's. Vector p is said to majorise vector q if*

$$\sum_{i=1}^{k} p_i \geq \sum_{i=1}^{k} q_i \quad \text{for all } k \in [n-1],$$

and

$$\sum_{i=1}^{n} p_i = \sum_{i=1}^{n} q_i.$$

The following lemma shows one of the main properties of majorisation that will be used frequently in the thesis.

Lemma 1. *Let $p = (p_1, p_2, \ldots, p_n)$ and $q = (q_1, q_2, \ldots, q_n)$ be two vectors of success probabilities. Let $X \sim \mathrm{PB}(p)$ and $Y \sim \mathrm{PB}(q)$. If vector p majorises vector q, then*

a) $\Pr(X = n) \leq \Pr(Y = n)$.

b) $\Pr(X > \mathbb{E}[X] + 1) \leq \Pr(Y > \mathbb{E}[Y] + 1)$.

Proof. We first prove **(a)**. By [90, Proposition F.1.a.], if the vector p majorises the vector q, then

$$\prod_{i=1}^{n} p_i \leq \prod_{i=1}^{n} q_i.$$

The proof is complete by noting that $\Pr(X = n) = \prod_{i=1}^{n} p_i$ and $\Pr(Y = n) = \prod_{i=1}^{n} q_i$.

For the statement **(b)**, we note from [90, Theorem K.1, p. 495] that the probability $\Pr(Y \leq \mathbb{E}[Y] + 1)$ is a Schur-convex function, which means that if the vector p majorises the vector q, then $\Pr(X \leq \mathbb{E}[X] + 1) \geq \Pr(Y \leq \mathbb{E}[Y] + 1)$, which can be rewritten as $\Pr(X > \mathbb{E}[X] + 1) \leq \Pr(Y > \mathbb{E}[Y] + 1)$. ■

We also recall the concept of stochastic domination that offers a way of comparing two random variables (not necessarily defined on the same probability space) [30].

Definition 8 (Definition 8.1 [30])**.** *Let X and Y be two random variables. We say that Y stochastically dominates X, if for all $k \in \mathbb{R}$ we have* $\Pr(X \geq k) \leq \Pr(Y \geq k)$.

Lemma 2 (Corollary 1.8.3 [30])**.** *Let X and Y be two random variables. If X stochastically dominates Y, then* $\mathbb{E}[X] \geq \mathbb{E}[Y]$.

Lemma 3 (Lemma 8.4 [30])**.** *Let $X_1, \ldots, X_n, Y_1, \ldots, Y_n$ be independent random variables. If X_i stochastically dominates Y_i for all $i \in [n]$, then $\sum_{i=1}^{n} X_i$ stochastically dominates $\sum_{i=1}^{n} Y_i$.*

4.3 A runtime of $O(\lambda n)$ for $\mu = \Omega(\log n) \cap O(\sqrt{n})$

Our approach refines the analysis in [23] by considering anti-concentration properties of the random variables involved. We shall apply Theorem 4, so we need to verify the three conditions (G1), (G2) and (G3) before an upper bound on the expected runtime is guaranteed. We first note that the range of values of the marginals are

$$p_{t,i} \in \{k/\mu : k \in [\mu - 1]\} \cup \{1 - 1/n, 1/n\}.$$

Here, we assume that $\mu < n$ and then categorise all marginals into three groups: those at the upper border $1 - 1/n$, those at the lower border $1/n$, and those within the closed interval $[1/\mu, 1 - 1/\mu]$. For OneMax, all bits have the same weight and the fitness is just the sum of these bit values, so the re-arrangement of bit positions will have no impact on the sampling distribution. Recall in Algorithm 3 that $X_{t,i}$ is the number of ones sampled at bit position $i \in [n]$ among the μ fittest individuals in iteration $t \in \mathbb{N}$. Without loss of generality, we can re-arrange the bit-positions so that for two integers $k, \ell \geq 0$, it holds that

- for all $i \in [1,k]$, $1 \leq X_{t,i} \leq \mu - 1$ and $p_{t+1,i} = X_{t,i}/\mu$ (the k-region),
- for all $i \in [k+1, k+\ell]$, $X_{t,i} = \mu$ and $p_{t+1,i} = 1 - 1/n$ (the ℓ-region), and
- for all $i \in [k+\ell+1, n]$, $X_{t,i} = 0$ and $p_{t+1,i} = 1/n$.

We define the levels using the canonical fitness-based partition (see Definition 6) as follows.

$$A_j := \{x \in \{0,1\}^n : \textsc{OneMax}(x) = j - 1\} \quad \text{for each} \quad j \in [n+1]. \tag{4.1}$$

There are $n+1$ levels, namely $A_1, A_2, \ldots, A_{n+1}$, where the (optimal) level A_{n+1} consists of the all-ones bitstring only. For condition (G1) of Theorem 4, we seek a lower bound z_j on the probability of sampling an offspring in levels $A_{\geq j+1}$ in iteration $t+1$, denoted as $\Pr(Y_{t+1,1:n} \geq j) \geq z_j$, assuming that the current population contains at least $\gamma_0\lambda$ individuals in levels $A_{\geq j}$. We also note that the logarithmic factor of $\Theta(\log\lambda)$ in the previous upper bound of $O(n\lambda\log\lambda)$ in [23] results from the lower bound $z_j = \Omega(1/\mu)$. Since we now aim for an upper bound of $O(n\lambda)$ on the expected runtime of the UMDA on ONEMAX, we need a better lower bound of at least $z_j = \Omega(\frac{n-j+1}{n})$.

Following [23], we choose the parameter $\gamma_0 := \mu/\lambda$. For the condition (G1), we assume $|P_t \cap A_{\geq j}| \geq \gamma_0\lambda = \mu$. In this case, we shall refer to A_j as the current level. Together with the two variables k and ℓ, the assumption implies that any of the μ fittest individuals has at least $j-\ell-1$ ones in the k-region. Once the probability vector has been updated, a new offspring with ℓ ones in the ℓ-region can be sampled in iteration $t+1$ with probability at least $(1-1/n)^\ell \geq 1/e$. The new offspring is in levels $A_{\geq j+1}$ if an extra of at least $j-\ell$ ones is sampled at the $n-\ell$ remaining bit positions. This leads us to consider three distinct cases for different configurations of the current population per the two parameters k and j.

Case 1 : $k \geq \mu$.

Recall that $Y_{t+1,1:k}$ denotes the number of ones sampled in the k-region in iteration $t+1$. When $k \geq \mu$, the variance of $Y_{t+1,1:k}$ is not too small; thus, by Theorem 8, the distribution of $Y_{t+1,1:k}$ cannot concentrate much around its mean $\mathbb{E}\left[Y_{t+1,1:k}\right] = j-\ell-1$. In this case, we will show that there is still a constant probability of sampling at least $j-\ell$ ones in the k-region in iteration $t+1$, given that another ℓ ones can be sampled in the ℓ-region with probability $\Omega(1)$. Consequently, an offspring with at least $(j-\ell)+\ell = j$ ones is sampled in iteration $t+1$ with probability

$$\Pr(Y_{t+1,1:n} \geq j) \geq \Pr(Y_{t+1,1:k} \geq j-\ell)\Pr(Y_{t+1,k+1:k+\ell} = \ell) = \Omega(1).$$

Case 2 : $k < \mu$ and $j \geq n+1-\frac{n}{\mu}$.

In this case, the current level is very close to the optimal level A_{n+1}, and any of the μ fittest individuals in the population in iteration t has few zeros. As already shown in [23], the probability of sampling in iteration $t+1$ an offspring in levels $A_{\geq j+1}$ is

$\Omega(\frac{1}{\mu})$. Since the second condition above can be rewritten as $\frac{1}{\mu} \geq \frac{n-j+1}{n}$, it ensures a lower bound of $z_j = \Omega(\frac{1}{\mu}) = \Omega(\frac{n-j+1}{n})$.

Case 3 : $0 \leq k < (1-c)(n-j+1)$.

The remaining cases. Assume further that the parent population size satisfies $\mu \leq \sqrt{n(1-c)}$ for some constant $c \in (0,1)$. By excluding the two cases above, we get $0 \leq k < (1-c)(n-j+1)$. Now, k is relatively small as j is close to n, and ℓ is not too large since the current level is not very close to the optimal level A_{n+1}. This implies that most zeros in the bitstring of any of the μ fittest individuals must lie among bit positions $[k+\ell+1, n]$, and it suffices to sample an extra one from this region in iteration $t+1$, while maintaining ones in the k-region and the ℓ-region, to obtain an offspring with at least $(j-\ell-1)+\ell+1 = j$ ones. We then show that in this case the probability of sampling such an offspring in iteration $t+1$ is still at least $z_j = \Omega(\frac{n-j+1}{n})$.

Theorem 9. *The* UMDA *with a parent population size* $a\ln(n) \leq \mu \leq \sqrt{n(1-c)}$ *for some sufficiently large constant* $a > 0$ *and any constant* $c \in (0,1)$*, and an offspring population size* $\lambda \geq \frac{13e}{1-c}\mu$*, has an expected runtime of* $O(n\lambda)$ *on the* ONEMAX *function.*

Proof. We re-arrange the bit positions per the two variables k and ℓ as explained above. The levels are defined as in (4.1). There are exactly $m = n+1$ levels, namely $A_1, A_2, \ldots, A_{n+1}$, where the (optimal) level A_{n+1} contains the all-ones bitstring only.

Condition (G2)

Let $\gamma_0 = \mu/\lambda$. Assume that $|P_t \cap A_{\geq j}| \geq \gamma_0\lambda = \mu$ and $|P_t \cap A_{\geq j+1}| \geq \gamma\lambda > 0$ for all $\gamma \in (0, \gamma_0]$, we are required to show that the probability of sampling an offspring in iteration $t+1$ in levels $A_{\geq j+1}$ must be at least $(1+\delta)\gamma$ for some $\delta \in (0,1)$. By the re-arrangement of the bit positions, it holds that

$$\sum_{i=k+1}^{k+\ell} X_{t,i} = \mu\ell \quad \text{and} \quad \sum_{i=k+\ell+1}^{n} X_{t,i} = 0. \tag{4.2}$$

In the current population P_t, there are at least $\gamma\lambda$ individuals with at least j ones, and also at least $\mu - \gamma\lambda$ other individuals with exactly $j-1$ ones. Therefore,

$$\sum_{i=1}^{n} X_{t,i} \geq \gamma\lambda\, j + (\mu - \gamma\lambda)(j-1) = \gamma\lambda + \mu(j-1). \tag{4.3}$$

Combining (4.2), (4.3) and noting that $\lambda = \mu/\gamma_0$ yield

$$\begin{aligned}\sum_{i=1}^{k} X_{t,i} &= \sum_{i=1}^{n} X_{t,i} - \sum_{i=k+1}^{k+\ell} X_{t,i} - \sum_{i=k+\ell+1}^{n} X_{t,i} \\ &\geq \gamma\lambda + \mu(j-1) - \mu\ell \\ &= \mu\left(\frac{\gamma}{\gamma_0} + j - 1 - \ell\right).\end{aligned} \tag{4.4}$$

Let $Z = Y_{t+1,1:k} + Y_{t+1,k+\ell+1:n}$ be an integer-valued random variable describing the number of ones sampled in all but the ℓ-region in an offspring in iteration $t+1$. Since $k+\ell \leq n$ and by (4.4), the expected value of the random variable Z is

$$\mathbb{E}[Z] = \sum_{i=1}^{k} p_{t+1,i} + \sum_{i=k+\ell+1}^{n} p_{t+1,i} = \frac{1}{\mu}\sum_{i=1}^{k} X_{t,i} + \frac{n-k-\ell}{n} \geq j - \ell - 1 + \frac{\gamma}{\gamma_0}. \tag{4.5}$$

In order to sample an offspring with at least j ones in iteration $t+1$, it suffices to sample ℓ ones in the ℓ-region and at least $j-\ell$ ones at the other positions. Such an event happens with probability

$$\Pr(Y_{t+1,1:n} \geq j) \geq \Pr(Z \geq j-\ell)\Pr\left(Y_{t+1,k+1:k+\ell} = \ell\right), \tag{4.6}$$

where the probability to obtain ℓ ones in the ℓ-region is

$$\Pr\left(Y_{t+1,k+1:k+\ell} = \ell\right) = \left(1 - \frac{1}{n}\right)^{\ell} \geq \left(1 - \frac{1}{n}\right)^{n-1} \geq \frac{1}{e}. \tag{4.7}$$

We now estimate the probability $\Pr(Z \geq j-\ell)$ using Feige's inequality (see Theorem 7). Since Z takes integer values only, it follows by (4.5) that

$$\Pr(Z \geq j-\ell) = \Pr(Z > j-\ell-1) \geq \Pr\left(Z > \mathbb{E}[Z] - \frac{\gamma}{\gamma_0}\right).$$

Applying Theorem 7 for $\Delta = \gamma/\gamma_0 \leq 1$ and assuming further that the two population sizes μ and λ are chosen such that $1/\gamma_0 = \lambda/\mu \geq 13e/(1-c) = 13e(1+\delta)$ yield

$$\Pr(Z \geq j-\ell) \geq \min\left\{\frac{1}{13}, \frac{\Delta}{\Delta+1}\right\} \geq \frac{\Delta}{13} = \frac{\gamma}{13\gamma_0} \geq e(1+\delta)\gamma. \tag{4.8}$$

Substituting (4.7) and (4.8) into (4.6) yields $\Pr(Y_{1,n} \geq j) \geq (1+\delta)\gamma$, which verifies the condition (G2) of Theorem 4.

Condition (G1)

Assume that $|P_t \cap A_{\geq j}| \geq \gamma_0 \lambda = \mu$. We seek a lower bound of z_j on the probability $\Pr(Y_{t+1,1:n} \geq j)$. By the assumption, any of the μ fittest individuals in the current population is in the current level A_j or higher; however, we consider the worst-case when the μ fittest individuals are all assumed to be in the current level A_j. This will only reduce the marginals $p_{t+1,j+1}, \ldots, p_{t+1,n}$ in the context of the ONEMAX problem. Hence, by Lemma 3, the distribution of $Y_{t+1,1:n}$ for the worst-case scenario is stochastically dominated by $Y_{t+1,1:n}$ for the current population. By Definition 8, a lower bound z_j that holds for the worst case therefore also holds for the original population. The μ fittest individuals in iteration t all have exactly $j-1$ ones, and, therefore, $\sum_{i=1}^{n} X_{t,i} = \mu(j-1)$ and $\sum_{i=1}^{k} X_{t,i} = \mu(j-\ell-1)$. There are four distinct cases that cover all situations according to different values of the variables k and j. We are about to show that in each case, the probability of sampling an offspring in levels $A_{\geq j+1}$ in iteration $t+1$ is lower bounded by $z_j = \Omega(\frac{n-j+1}{n})$.

Case 0 : $k = 0$.

There is no k-region. In this case, we also have $p_{t+1,i} = 1 - 1/n$ for $1 \leq i \leq j-1$ (i.e., the ℓ-region), and $p_{t+1,i} = 1/n$ for $j \leq i \leq n$. To sample an offspring with j ones in iteration $t+1$, it suffices to obtain $\ell = j-1$ ones in the ℓ-region and an extra one at one of the $n-\ell$ remaining bit positions.

$$\Pr(Y_{t+1,1:n} \geq j) \geq \frac{n-j+1}{n} \cdot \left(1 - \frac{1}{n}\right)^{n-1} = \Omega\left(\frac{n-j+1}{n}\right).$$

Case 1 : $k \geq \mu$.

We will apply the anti-concentration inequality in Theorem 8. To obtain a lower bound on the variance of the number of ones sampled in iteration $t+1$ in the k-region, we use the bounds $1/\mu \leq p_{t+1,i} \leq 1 - 1/\mu$, which hold for $1 \leq i \leq k$. We get

$$\mathrm{Var}\left[Y_{t+1,1:k}\right] = \sum_{i=1}^{k} p_{t+1,i}(1 - p_{t+1,i}) \geq \frac{k}{\mu}\left(1 - \frac{1}{\mu}\right) \geq \frac{9k}{10\mu} \geq \frac{9}{10},$$

where the second inequality holds for sufficiently large n because $\mu \geq a \ln(n)$ for some constant $a > 0$. Theorem 8 applied with $\text{Var}\left[Y_{t+1,1:k}\right] \geq 9/10$ now gives

$$\Pr\left(Y_{t+1,1:k} = j - \ell - 1\right) \leq \frac{\eta}{\sqrt{\text{Var}\left[Y_{t+1,1:k}\right]}} < \frac{0.4689}{\sqrt{9/10}} < 0.4943. \tag{4.9}$$

Furthermore, since $\mathbb{E}\left[Y_{t+1,1:k}\right] = \sum_{i=1}^{k} X_{t,i}/\mu = j - \ell - 1$ is an integer, Theorem 6 implies that

$$\Pr\left(Y_{t+1,1:k} \geq j - \ell - 1\right) \geq 1/2. \tag{4.10}$$

By (4.9) and (4.10), the probability of sampling an offspring with at least $j - \ell$ ones in the k-region in iteration $t+1$ is

$$\begin{aligned}
\Pr\left(Y_{t+1,1:k} \geq j - \ell\right) &= \Pr\left(Y_{t+1,1:k} \geq j - \ell - 1\right) - \Pr\left(Y_{t+1,1:k} = j - \ell - 1\right) \\
&= \Pr\left(Y_{t+1,1:k} \geq \mathbb{E}\left[Y_{1,k}\right]\right) - \Pr\left(Y_{t+1,1:k} = j - \ell - 1\right) \\
&> 1/2 - 0.4943 \\
&= 0.0057.
\end{aligned}$$

To obtain an offspring in levels $A_{\geq j+1}$ in iteration $t+1$, it suffices to sample at least $j - \ell$ ones in the k-region and ℓ ones in the ℓ-region. Therefore, using (4.7) and the lower bound above, this event happens with probability at least

$$\begin{aligned}
\Pr\left(Y_{t+1,1:n} \geq j\right) &\geq \Pr\left(Y_{t+1,1:k} \geq j - \ell\right) \Pr\left(Y_{t+1,k+1:k+\ell} = \ell\right) \\
&> 0.0057 \cdot (1/e) \\
&= \Omega\left(\frac{n - j + 1}{n}\right),
\end{aligned}$$

where the bound $\Omega((n - j + 1)/n)$ follows trivially from the constant before.

Case 2 : $1 \leq k < \mu$ and $j \geq n(1 - 1/\mu) + 1$ (or, equivalently, $1/\mu \geq (n - j + 1)/n$).

The probability of sampling an offspring in levels $A_{\geq j+1}$ is then bounded from below by

$$\begin{aligned}\Pr\left(Y_{t+1,1:n} \geq j\right) &\geq \Pr\left(Y_{t+1,1:1} = 1\right) \Pr\left(Y_{t+1,2:k} \geq j-\ell-1\right) \Pr\left(Y_{t+1,k+1:k+\ell} = \ell\right) \\ &\geq p_{t+1,1} \cdot \Pr\left(Y_{t+1,2:k} \geq j-\ell-1\right) \cdot \frac{1}{e}.\end{aligned}$$

Because $Y_{t+1,2:k}$ takes integer values only, and

$$\begin{aligned}\mathbb{E}\left[Y_{t+1,2:k}\right] &= \mathbb{E}\left[Y_{t+1,1:k}\right] - p_{t+1,1} \\ &\geq (\gamma/\gamma_0) + j - 1 - \ell - p_{t+1,1} && \text{(by (4.4))} \\ &\geq j - \ell - 1 - p_{t+1,1}, && \text{(as } \gamma \geq 0)\end{aligned}$$

we then have

$$\begin{aligned}\Pr\left(Y_{t+1,2:k} \geq j-\ell-1\right) &= \Pr\left(Y_{t+1,2:k} > j-\ell-2\right) \\ &= \Pr\left(Y_{t+1,2:k} > (j-\ell-1-p_{t+1,1}) - (1-p_{t+1,1})\right) \\ &\geq \Pr\left(Y_{t+1,2:k} > \mathbb{E}\left[Y_{t+1,2:k}\right] - (1-p_{t+1,1})\right) \\ &\geq \min\left\{\frac{1}{13}, \frac{1-p_{t+1,1}}{2-p_{t+1,1}}\right\} \\ &> (1-p_{t+1,1})/13,\end{aligned}$$

where the second inequality follows from Theorem 7. Putting things together, assuming further that $\mu \geq 14$, and noting that $p_{t+1,1} \geq 1/\mu$ (as in the k-region) and $1/\mu \geq (n-j+1)/n$, we obtain

$$\begin{aligned}\Pr\left(Y_{t+1,1:n} \geq j\right) &> \frac{p_{t+1,1}(1-p_{t+1,1})}{13e} \geq \frac{(1/\mu)(1-1/\mu)}{13e} \\ &\geq \frac{1-1/14}{13e\mu} \geq \frac{1}{14e\mu} \geq \frac{n-j+1}{14en} = \Omega\left(\frac{n-j+1}{n}\right).\end{aligned}$$

Case 3 : $1 \leq k < \mu$ and $j < n(1-1/\mu)+1$.

This case covers all the remaining situations not included by the preceding cases. The second inequality can be rewritten as $n-j+1 \geq n/\mu$. We also have $\mu \leq \sqrt{n(1-c)}$,

so $n/\mu \geq \mu/(1-c)$. It then holds that

$$(1-c)(n-j+1) \geq (1-c)(n/\mu) \geq (1-c)\mu/(1-c) = \mu > k.$$

Thus, the two conditions can be simplified to $1 \leq k < (1-c)(n-j+1)$. In this case, the probability of sampling an offspring in iteration $t+1$ with j ones is

$$\begin{aligned}\Pr(Y_{t+1,1:n} \geq j) &\geq \Pr\left(Y_{t+1,1:k} \geq j-\ell-1\right) \Pr\left(Y_{t+1,k+1:k+\ell} = \ell\right) \Pr\left(Y_{t+1,k+\ell+1:n} \geq 1\right) \\ &\geq \frac{1}{2} \cdot \frac{1}{e} \cdot \frac{n-k-\ell}{n} \\ &= \frac{n-k-\ell}{2en},\end{aligned}$$

where the factor of $1/2$ in the last inequality is due to (4.10). Since $\ell \leq j-1$ and $k < (1-c)(n-j+1)$, it follows that

$$\Pr\left(Y_{t+1,1:n} \geq j\right) > \frac{n-(1-c)(n-j+1)-j+1}{2en} = \Omega\left(\frac{n-j+1}{n}\right).$$

Combining all four cases together yields the probability of sampling an offspring in levels $A_{\geq j+1}$ as follows.

$$\Pr\left(Y_{t+1,1:n} \geq j\right) = \Omega\left(\frac{n-j+1}{n}\right) =: z_j,$$

and $z_* := \min_{j \in [m-1]}\{z_j\} = \Omega(1/n)$. The condition (G1) of Theorem 4 is now satisfied.

Condition (G3)

We have $1/\delta^2 = O(1)$, $1/z_* = O(n)$, and $m = O(n)$. Therefore, there must exist a constant $a > 0$ such that

$$\left(\frac{a}{\gamma_0}\right) \ln(n) \geq \left(\frac{4}{\gamma_0 \delta^2}\right) \ln\left(\frac{128m}{z_* \delta^2}\right).$$

The requirement $\mu \geq a \ln(n)$ now implies that

$$\lambda = \frac{\mu}{\mu/\lambda} \geq \left(\frac{a}{\gamma_0}\right) \ln(n) \geq \left(\frac{4}{\gamma_0 \delta^2}\right) \ln\left(\frac{128m}{z_* \delta^2}\right);$$

hence, condition (G3) is satisfied.

We have verified all three conditions (G1), (G2), and (G3). By Theorem 4 and the bound $z_j = \Omega((n-j+1)/n)$, the expected runtime is

$$\mathbb{E}[T] = O\left(\lambda \sum_{j=1}^{n} \ln\left(\frac{n}{n-j+1}\right) + \sum_{j=1}^{n} \frac{n}{n-j+1}\right).$$

We simplify the two terms separately. The first term is

$$\begin{aligned}
O\left(\lambda \sum_{j=1}^{n} \ln\left(\frac{n}{n-j+1}\right)\right) &= O\left(\lambda \ln \prod_{j=1}^{n} \frac{n}{n-j+1}\right) \\
&= O\left(\lambda \ln\left(\frac{n^n}{n!}\right)\right) \\
&= O\left(\lambda \ln \frac{n^n \cdot e^n}{n^{n+1/2}}\right) && \text{(by Stirling's approx. [19])} \\
&= O(\lambda \ln e^n) && \text{(as } n^n/n^{n+1/2} < 1\text{)} \\
&= O(\lambda n).
\end{aligned}$$

By noting that $\sum_{k=1}^{n}(1/k) \leq 1 + \ln n$ [19, p. 1154], the second term is

$$O\left(\sum_{j=1}^{n} \frac{n}{n-j+1}\right) = O\left(n \sum_{k=1}^{n} \frac{1}{k}\right) = O(n \log n).$$

Since $\lambda \geq \mu = \Omega(\log n)$, the overall expected runtime is

$$\mathbb{E}[T] = O(n\lambda) + O(n \log n) = O(n\lambda). \qquad \blacksquare$$

4.4 A runtime of $O(\lambda\sqrt{n})$ for $\mu = \Omega(\sqrt{n}\log n)$

For larger parent population sizes $\mu \geq c\sqrt{n}\log n$ for some constant $c > 0$, we aim for an upper bound of $O(\lambda\sqrt{n})$ on the expected runtime of the UMDA on the ONEMAX function. We shall rely on a property of the UMDA on the ONEMAX function proven in [127] to derive such an upper bound. Note that he same upper bound of $O(\lambda\sqrt{n})$ has been obtained in [127]. Our proof here is simpler and also relaxes the strong requirement of $\lambda = \Theta(\mu)$ assumed in [127] to any population sizes $\lambda = \Omega(\mu)$. However, it requires the ratio $\lambda/\mu \geq 294$, which was not required in [127].

To begin with, Witt [127, Lemma 3.2] found that when the parent population size is $\mu \geq c\sqrt{n}\log n$, the probability that the marginal of an arbitrary bit position $i \in [n]$ drops below the value $1/4$ within the $n^{cc'}$ first iterations is at most $O(n^{-cc'})$ for some constant $c' > 0$. The result also means that all marginals exhibit a positive drift towards the upper border $1 - 1/n$ (also known as genetic drift) and are unlikely to drop by a large amount [127]. In this situation, we assume that all marginals are lower bounded by a constant $p_{\min} = 1/4$ during the optimisation. Moreover, we re-arrange the bit positions such that for two integers $0 \leq k, \ell \leq n$ such that $k + \ell = n$ and

- it holds for all $1 \leq i \leq k$ that $p_{\min} \leq p_{t+1,i} \leq 1 - \frac{1}{\mu}$ (i.e., the k-region), and
- it holds for all $k+1 \leq i \leq n$ that $p_{t+1,i} = 1 - \frac{1}{n}$ (i.e., the ℓ-region).

We also note that $k > 0$; otherwise, the global optimum has been sampled.

In the preceding section, we used the canonical fitness-based partition, defined in (4.1). Two consecutive levels differ by a one, and there is a total of $n+1$ levels. The desired expected runtime of $O(\lambda\sqrt{n})$ implies an expected number of $O(\sqrt{n})$ iterations before the global optimum has been found for the first time because there are λ function evaluations performed in each iteration of the UMDA. Also, the two conditions (G1) and (G2) of the level-based theorem only concern with the probability of making progress between two consecutive levels. Therefore, the application of the level-based theorem with the level definition in (4.1) still results in an expected number of $\Theta(n)$ iterations even when the current level gets increased by one in each iteration. In other words, the level definition in (4.1) does not help us to show the upper bound of $O(\lambda\sqrt{n})$, and we need a new partition of the search space that should ideally consist of $O(\sqrt{n})$ levels only. This also means that, under the 'ideal' partition, some level contains individuals with different ONEMAX-values.

Let us now consider Theorem 5. Given a random variable $X \sim \mathrm{PB}(p_1, p_2, \ldots, p_n)$ where $p_i = \Omega(1)$ for all $i \in [n]$, there is a constant probability of deviating by a distance of $\Theta(\sqrt{n - \mathbb{E}[X]})$ above the expected value $\mathbb{E}[X] = \sum_{i=1}^{n} p_i = \Omega(n)$. Inspired by this result, we partition the search space into the m disjoint subsets $A_1, A_2, \ldots, A_m$ as follows.

$$\begin{aligned} A_i &:= \{x \in \{0,1\}^n : f_{i-1} \leq \textsc{OneMax}(x) < f_i\} \text{ for each } i \in [m-1], \\ \text{and } A_m &:= \{1^n\}, \end{aligned} \tag{4.11}$$

where the sequence $(f_i : i \in \mathbb{N})$ is defined with some constant $d \in (0,1)$ as

$$f_0 \coloneqq 0 \quad \text{and} \quad f_{i+1} \coloneqq f_i + \lceil d\sqrt{n - f_i} \rceil. \tag{4.12}$$

The range of d will be specified later, but for now note that $m = \min\{i \in \mathbb{N} : f_i = n\}$. The following result shows that the sequence $(f_i : i \in \mathbb{N})$ is well-behaved: it starts at 0 and increases steadily (at least 1 per level), then eventually reaches n exactly and remains there afterwards. Moreover, the number of levels satisfies $m = O(\sqrt{n})$.

Lemma 4. *For any $n \in \mathbb{N}$, any constant $d \in (0,1]$ independent of n and the sequence $(f_i : i \in \mathbb{N})$ defined according to (4.12), it holds that*

(i) *$f_i \leq n$ for all $i \in \mathbb{N}$, and $\exists j \in \mathbb{N}$: $f_j = n$,*

(ii) *if $m = \min\{i \in \mathbb{N} : f_i = n\}$ then $m = O(\sqrt{n})$.*

Proof. We first prove **(i)**: it is easy to see that f_i are all integers, i.e. $f_i \in \mathbb{N}$ for all $i \in \mathbb{N}$. Due to the ceiling function, if $f_i < n$, then $f_{i+1} \geq f_i + 1$, in other words starting with $f_0 = 0$, the sequence will increase steadily until it hits n exactly or overshoots it. Assuming the later case of overshooting, that is, $\exists k \geq 0$: $f_k \leq n-1$ and $f_{k+1} \geq n+1$ (and after that $f_{k+2}, \ldots$ are ill-defined). By the definition of the sequence, the property $1 + x > \lceil x \rceil$ of the ceiling function and $d \leq 1$, we have

$$1 + \sqrt{n - f_k} > \lceil \sqrt{n - f_k} \rceil \geq \lceil d\sqrt{n - f_k} \rceil = f_{k+1} - f_k \geq 2,$$

this implies that $f_k < n-1$ or $f_k \leq n-2$. Repeating the above argument again gives that $1 + \sqrt{n - f_k} > 3$, and $f_k < n-4$. After a finite number of repetitions we will conclude that $f_k < 0$ which is a contradiction. Therefore, the sequence must reach n strictly at one specific point in time and remain there afterwards.

To bound m in **(ii)**, we will apply drift analysis with respect to the deterministic sequence $(f_i : i \in \mathbb{N})$ and interpret m as the hitting time of the value n. Let $f_i \mapsto n - f_i$ be the potential function. The value of m is then equivalent to the number of time steps taken for the potential function to reach the target value of 0. By (4.12), the single-step change is $(n - f_i) - (n - f_{i+1}) = f_{i+1} - f_i = \lceil d\sqrt{n - f_i} \rceil \geq d\sqrt{n - f_i}$. By applying the variable drift theorem (see Theorem 3) for a positive, increasing function $x \mapsto d\sqrt{x}$ where $d \in (0,1)$, the

number of steps is bounded from above by

$$\frac{1}{d} + \int_1^n \frac{\partial x}{d\sqrt{x}} = \frac{1}{d} + \frac{2}{d}\sqrt{x}\Big|_1^n = \frac{2\sqrt{n}-1}{d} = \mathrm{O}\left(\sqrt{n}\right),$$

which completes the proof. ■

To satisfy condition (G2) of the level-based theorem, we shall prove the following result, which says that given a random variable following a Poisson binomial distribution, if all success probabilities are not too small, then the probability of exceeding the expectation is at least a constant. To do that, we first recall the following tail bound for a binomially distributed random variable [29, Theorem 12].

Lemma 5 (Theorem 12 [29])**.** *Let $n \in \mathbb{N}$ and $p \in [0,1]$. Let $X \sim \mathrm{Bin}(n,p)$. If $1/n \leq p < 1-1/n$, then $\Pr(X > \mathbb{E}[X]+1) \geq 0.037$.*

We are now ready to prove a lower bound on the probability of exceeding the expectation for a Poisson binomial distribution.

Lemma 6. *Let $n \in \mathbb{N}$. Let $Y \sim \mathrm{PB}(p_1, p_2, \ldots, p_n)$, where $p_i \geq 1/4$ for all $i \in [n]$. Then, $\Pr(Y \geq \mathbb{E}[Y]) \geq 0.00925$.*

Proof. Let us express $Y = \sum_{i=1}^n Y_i$, where $Y_i \sim \mathrm{Ber}(p_i)$ and $p_i \geq 1/4 =: p_{\min}$ for all $i \in [n]$. We introduce n Bernoulli random variables $Z_1, Z_2, \ldots, Z_n$ with success probabilities $z_1, z_2, \ldots, z_n$, and $Z = \sum_{i=1}^n Z_i$ denotes the sum of these random variables. Let $m := \lfloor (\mathbb{E}[Y] - n \cdot p_{\min})/(1-p_{\min}) \rfloor = \lfloor (4\mathbb{E}[Y]-n)/3 \rfloor$. We choose the success probabilities $z_1, z_2, \ldots, z_n$ such that

$$z_i = \begin{cases} 1 & \text{for all } i \in [m] \\ q & \text{for } i = m+1, \\ p_{\min} & \text{for all } i \in \{m+2, \ldots, n\}, \end{cases}$$

where $p_{\min} \leq q \leq 1$ by the definition of m. In other words, we construct the new random variables $Z_1, Z_2, \ldots, Z_n$ by shifting the success probabilities of random variables $Y_1, Y_2, \ldots, Y_n$ to the Z_i's with smaller indices so long as $\sum_{i=1}^n p_i = \sum_{i=1}^n z_i$. By Definition 7, the vector

$(z_1,\dots,z_n)$ majorises the vector $(p_1,\dots,p_n)$. Therefore,

$$\begin{aligned}\Pr(Y \geq \mathbb{E}[Y]) &\geq \Pr(Y > \mathbb{E}[Y]+1)\\ &\geq \Pr(Z > \mathbb{E}[Z]+1). \qquad \text{(by Lemma 1)}\end{aligned}$$

Furthermore, we always get $Z_i = 1$ for all $i \in [m]$. Let $Z_{m+2,n} = \sum_{i=m+2}^{n} Z_i$. We also know $\mathbb{E}[Z_{m+2,n}] + q = \mathbb{E}[Z] - m$. Thus,

$$\begin{aligned}\Pr(Z > \mathbb{E}[Z]+1) &\geq \Pr(Z_{m+1} + \ldots + Z_n > \mathbb{E}[Z]+1-m)\\ &\geq \Pr(Z_{m+1}=1)\cdot\Pr(Z_{m+2,n} > \mathbb{E}[Z]-m)\\ &= q\cdot\Pr(Z_{m+2,n} > \mathbb{E}[Z_{m+2,n}]+q)\\ &\geq (1/4)\cdot\Pr(Z_{m+2,n} > \mathbb{E}[Z_{m+2,n}]+1). \qquad (\text{as } 1 \geq q \geq p_{\min})\end{aligned}$$

Because $Z_{m+2,n} \sim \mathrm{Bin}(n-m-1,1/4)$, by Lemma 5 we obtain

$$\Pr(Z_{m+2,n} > \mathbb{E}[Z_{m+2,n}]+1) \geq 0.0370.$$

Putting everything together yields

$$\Pr(Y \geq \mathbb{E}[Y]) \geq \Pr(Z > \mathbb{E}[Z]+1) \geq (1/4)\cdot 0.0370 = 0.00925,$$

which completes the proof. ■

Lemma 7. *Let $n \in \mathbb{N}, x \in \mathbb{R}, y \in \mathbb{R}, a \in [0,1)$ and $d \in (0,1)$. If $n-1 \geq y \geq x-a$, then*

$$x + \lceil d\sqrt{n-x}\rceil \leq y + 3\sqrt{n-\lfloor y\rfloor}.$$

Proof. We start by considering the function $g(x) = x + d\sqrt{n-x}$. Because

$$\frac{\partial g}{\partial x} = 1 - \frac{d}{2\sqrt{n-x}} > 0$$

for $d \in (0,1)$, $g(x)$ is a monotonically increasing function. This means that for $x \leq y+a$, we get $g(x) \leq g(y+a)$. We also obtain

$$\begin{aligned} x + \lceil d\sqrt{n-x} \rceil &\leq (y+a) + d\sqrt{n-(y+a)} + 1 && \text{(by } \lceil x \rceil \leq x+1\text{)} \\ &\leq y + d^*\sqrt{n-y} && \text{(see below)} \\ &\leq y + d^*\sqrt{n-\lfloor y \rfloor}. && \text{(by } \lfloor y \rfloor \leq y\text{)} \end{aligned}$$

The second inequality holds if the value d^* is chosen such that

$$d^* \geq \max_y h(y), \quad \text{where} \quad h(y) := \frac{1+a+d\sqrt{(n-a)-y}}{\sqrt{n-y}}.$$

Because $\frac{\partial h}{\partial y} > 0$, $h(y)$ is also a monotonically increasing function, which implies that $\max_y h(y) = h(y_{\max}) = h(n-1) = 1+a+d\sqrt{1-a} < 3$. Therefore, the second inequality above holds when $d^* \geq 3$, which completes the proof. ■

We are now ready to prove an upper bound of $O(\lambda\sqrt{n})$ on the expected runtime of the UMDA with a parent population size $\mu = \Omega(\sqrt{n}\log n)$ on the ONEMAX function.

Theorem 10. *The* UMDA *with a parent population size $\mu \geq c\sqrt{n}\log n$ for some sufficiently large constant $c > 0$ and an offspring population size $\lambda \geq 294(1+\delta)\mu$ for some constant $\delta > 0$ has an expected runtime of $O(\lambda\sqrt{n})$ on the* ONEMAX *function.*

Proof. We apply the level-based theorem. The levels are defined in (4.11). By Lemma 4, there is a total of $m = O(\sqrt{n})$ levels. Let $\gamma_0 := \mu/\lambda$. We also assume that all marginals $p_{t,1}, p_{t,2}, \ldots, p_{t,n}$ are at least $1/4$ during the course of the optimisation. In the end, we shall relax this assumption to obtain the overall expected runtime.

Condition (G2)

We assume that $|P_t \cap A_{\geq j}| \geq \gamma_0\lambda = \mu$ and $|P_t \cap A_{\geq j+1}| \geq \gamma\lambda$ for all $\gamma \in (0, \gamma_0]$. Additionally, we make the pessimistic assumption that $|P_t \cap A_{\geq j+2}| = 0$, which means that the current population contains exactly $\gamma\lambda$ individuals in level A_{j+1} and at least $\mu - \gamma\lambda$ individuals in level A_j. Furthermore, we also assume that the at least $\gamma\lambda$ individuals in level A_{j+1} have exactly f_j ones and those in level A_j have exactly f_{j-1} ones. We call this the worst-case scenario. This assumption will reduce the marginals $p_{t,1}, p_{t,2}, \ldots, p_{t,n}$ in the context of the ONEMAX problem. Recall that the random variable $Y_{t+1,1:n}$ denotes the number of ones in

an offspring sampled in iteration $t+1$. By Lemma 3, the distribution of $Y_{t+1,1:n}$ for the worst case setting is still stochastically dominated by the distribution of $Y_{t+1,1:n}$ for the original population. Therefore, by Definition 8, a lower bound z_j that holds for the worst-case setting also holds for the original population. Let $f^* := n - 1/d^2$, where $d \in (0,1)$ is a constant and will be specified later. Let $j^* \in \mathbb{N}$ be an index such that $f_i \leq f^*$ for all $i \leq j^*$ and $f_i > f^*$ for all $i > j^*$. We will verify condition (G2) via a case distinction.

Case 1 : $j < j^* - 1$.

In this regime, we have $d\sqrt{n - f_j} > d\sqrt{n - (n - 1/d^2)} = 1$, and there always exists at least an integer in the interval $[f_{j-1}, f_j)$, which means that we will be able to ignore the ceiling function in (4.12). Recall the existence of two integers k and ℓ as discussed earlier. In this case,

$$\sum_{i=1}^{n} X_{t,i} = \gamma\lambda \cdot f_j + (\mu - \gamma\lambda) \cdot f_{j-1} = \mu\left(f_{j-1} + \frac{\gamma}{\gamma_0}\left(f_j - f_{j-1}\right)\right),$$

and, by noting that $\sum_{i=k+1}^{n} X_{t,i} = \mu\ell$,

$$\sum_{i=1}^{k} X_{t,i} = \sum_{i=1}^{n} X_{t,i} - \sum_{i=k+1}^{n} X_{t,i} = \mu\left(f_{j-1} + \frac{\gamma}{\gamma_0}\left(f_j - f_{j-1}\right) - \ell\right).$$

Recall that $Y_{t+1,1:k}$ is a random variable denoting the number of ones in the k-region in an offspring sampled in iteration $t+1$. The expected value of $Y_{t+1,1:k}$ is

$$\mathbb{E}\left[Y_{t+1,1:k}\right] = \frac{1}{\mu}\sum_{i=1}^{k} X_{t,i} = \left(f_{j-1} - \ell\right) + \frac{\gamma}{\gamma_0}\left(f_j - f_{j-1}\right). \tag{4.13}$$

Due to the assumption $p_{t,i} \geq p_{\min} = 1/4$, the variance of $Y_{t+1,1:k}$ is

$$\begin{aligned}
\operatorname{Var}\left[Y_{t+1,1:k}\right] &= \sum_{i=1}^{k} p_{t+1,i}(1-p_{t+1,i}) \\
&\geq p_{\min}\left(k - \sum_{i=1}^{k} p_{t+1,i}\right) && \text{(as } p_{t+1,i} \geq p_{\min}\text{)} \\
&= \frac{1}{4}\left(n - \ell - \mathbb{E}\left[Y_{t+1,1:k}\right]\right) && \text{(as } k+\ell = n\text{)} \\
&= \frac{1}{4}\left(n - \ell - f_{j-1} - \frac{\gamma}{\gamma_0}\left(f_j - f_{j-1}\right) + \ell\right) && \text{(by (4.13))} \\
&= \frac{1}{4}\left(n - f_{j-1} - \frac{\gamma}{\gamma_0}\left(f_j - f_{j-1}\right)\right) \\
&= \frac{1}{4}\left(n - f_{j-1} - \frac{\gamma}{\gamma_0} d\sqrt{n - f_{j-1}}\right) && \text{(by (4.12))} \\
&\geq \frac{1}{4}\left(n - f_{j-1} - d\left(n - f_{j-1}\right)\right) && \text{(as } \gamma/\gamma_0 \leq 1\text{).}
\end{aligned}$$

Therefore,

$$\operatorname{Var}\left[Y_{t+1,1:k}\right] \geq \frac{1}{4}\left(n - f_{j-1}\right)(1-d). \tag{4.14}$$

The probability of sampling an offspring in levels $A_{\geq j+1}$ in iteration $t+1$ is bounded from below by

$$\begin{aligned}
\Pr\left(Y_{t+1,1:n} \geq f_j\right) &\geq \Pr(Y_{t+1,1:k} \geq f_j - \ell) \cdot \Pr(Y_{t+1,k+1:n} = \ell) \\
&\geq \Pr\left(Y_{t+1,1:k} \geq f_j - \ell\right) \cdot \left(1 - \frac{1}{n}\right)^{\ell} \\
&\geq \frac{1}{e} \cdot \Pr\left(Y_{t+1,1:k} \geq f_j - \ell\right),
\end{aligned}$$

and

$$\begin{aligned}
\Pr\left(Y_{t+1,1:k} \geq f_j - \ell\right) \geq{}& \Pr\left(Y_{t+1,1:k} \geq \mathbb{E}\left[Y_{t+1,1:k}\right]\right) \\
&- \Pr\left(\mathbb{E}\left[Y_{t+1,1:k}\right] \leq Y_{t+1,1:k} < f_j - \ell\right).
\end{aligned} \tag{4.15}$$

By Lemma 6, we know that $\Pr\left(Y_{t+1,1:k} \geq \mathbb{E}\left[Y_{t+1,1:k}\right]\right) \geq 0.00925$. Also, by Theorem 8, we have

$$\begin{aligned}
\Pr(\mathbb{E}\left[Y_{t+1,1:k}\right] \leq Y_{t+1,1:k} < f_j - \ell) &\leq \frac{\eta\left(f_j - \ell - \mathbb{E}[Y_{t+1,1:k}]\right)}{\sqrt{\mathrm{Var}\left[Y_{t+1,1:k}\right]}} \\
&= \eta\left(1 - \frac{\gamma}{\gamma_0}\right)\frac{f_j - f_{j-1}}{\sqrt{\mathrm{Var}\left[Y_{t+1,1:k}\right]}} && \text{(by (4.13))} \\
&\leq \eta\left(1 - \frac{\gamma}{\gamma_0}\right)\frac{d\sqrt{n - f_{j-1}}}{\sqrt{(1/4)(1-d)(n-f_{j-1})}} && \text{(by (4.14))} \\
&= 2\eta\left(1 - \frac{\gamma}{\gamma_0}\right)\frac{d}{\sqrt{1-d}} \\
&< \left(1 - \frac{\gamma}{\gamma_0}\right)\frac{d}{\sqrt{1-d}}. && \text{(as } \eta < 1/2)
\end{aligned}$$

Thus, (4.15) becomes

$$\Pr(Y_{t+1,1:k} \geq f_j - \ell) \geq 0.00925 - \left(1 - \frac{\gamma}{\gamma_0}\right)\frac{d}{\sqrt{1-d}} \geq 0.00925\frac{\gamma}{\gamma_0}. \tag{4.16}$$

The last inequality holds if the constant d is chosen such that $\frac{d}{\sqrt{1-d}} \leq 0.00925$, which is equivalent to $0.00925^{-2}d^2 + d - 1 \leq 0$. The discriminant of this quadratic equation is $\Delta = 1 + 4 \cdot 0.00925^{-2} > 0$. The equation has two real solutions d_1 and d_2 as follows.

$$d_1 = -\frac{(1+\sqrt{\Delta}) \cdot 0.00925^2}{2} < 0,$$

and

$$d_2 = \frac{(-1+\sqrt{\Delta}) \cdot 0.00925^2}{2} > 0.$$

Therefore, if we choose any value of d such that $0 < d \leq d_2 \approx 0.009207$, then the inequality (4.16) always holds. The probability of sampling an offspring in levels $A_{\geq j+1}$ is therefore bounded from below by

$$\Pr(Y_{t+1,1:n} \geq f_j) \geq \frac{1}{e} \cdot 0.00925 \cdot \frac{\gamma}{\gamma_0} \geq (1+\delta)\gamma.$$

The last inequality holds if we choose the population sizes μ and λ of the UMDA such that $\mu/\lambda = \gamma_0 \leq 0.00925/((1+\delta)e)$ for some constant $\delta \in (0,1)$.

Case 2 : $j \geq j^* - 1$.

In this regime, we have $d\sqrt{n-f_j} \leq d\sqrt{n-(n-1/d^2)} = 1$. Without the ceiling function in (4.12), the difference between the two values f_{j-1} and f_j can be less than one, meaning that there may not exist any integer in the interval $[f_{j-1}, f_j)$ for some value of j. Thus, the ceiling function ensures that these levels (close to the optimal level A_m) are non-empty. In this case, the fitness values of two individuals in two consecutive levels differ by exactly one and we can re-use the proof of Theorem 9, in which the condition (G2) is satisfied by choosing the selection rate $\mu/\lambda = \gamma_0 \leq 1/(13(1+\delta))$ for some constant $\delta \in (0,1)$.

Putting the two cases together, condition (G2) of the level-based theorem can be verified by choosing the selection rate $\mu/\lambda \leq 0.00925/(e(1+\delta))$ for some constant $\delta \in (0,1)$, which can be simplified to $\lambda \geq 294(1+\delta)\mu$.

Condition (G1)

Assume that $|P_t \cap A_{\geq j}| \geq \gamma_0\lambda = \mu$. This means that the μ fittest individuals in the current population P_t belong to levels $A_{\geq j}$. In other words,

$$\sum_{i=1}^{n} X_{t,i} \geq \mu f_{j-1},$$

and

$$\sum_{i=1}^{k} X_{t,i} = \sum_{i=1}^{n} X_{t,i} - \sum_{i=k+1}^{n} X_{t,i} \geq \mu f_{j-1} - \mu\ell = \mu(f_{j-1} - \ell).$$

The expected value of $Y_{t+1,1:n}$ is

$$\mathbb{E}[Y_{t+1,1:n}] = \sum_{i=1}^{n} p_{t+1,i} = \frac{1}{\mu}\sum_{i=1}^{k} X_{t,i} + \sum_{i=k+1}^{n}\left(1-\frac{1}{n}\right) \geq f_{j-1} - \frac{\ell}{n}. \tag{4.17}$$

An individual belonging to the higher levels $A_{\geq j+1}$ must have at least f_j ones. The probability of sampling an offspring in levels $A_{\geq j+1}$ is denoted as $\Pr(Y_{t+1,1:n} \geq f_j)$. By Lemma 7 for

some constant $d^* \geq 1/p_{\min} = 4$ and $n-1 \geq \mathbb{E}[Y_{t+1,1:n}] \geq f_{j-1} - \ell/n$, we obtain

$$\begin{aligned}
&\Pr\left(Y_{t+1,1:n} \geq f_j\right) \\
&= \Pr\left(Y_{t+1,1:n} \geq f_{j-1} + \lceil d\sqrt{n - f_{j-1}}\rceil\right) && \text{(by (4.12))} \\
&\geq \Pr\left(Y_{t+1,1:n} \geq \mathbb{E}[Y_{t+1,1:n}] + 3\sqrt{n - \lfloor \mathbb{E}[Y_{t+1,1:n}]\rfloor}\right) && \text{(by Lemma 7)} \\
&> \Pr\left(Y_{t+1,1:n} \geq \mathbb{E}[Y_{t+1,1:n}] + 4\sqrt{n - \lfloor \mathbb{E}[Y_{t+1,1:n}]\rfloor}\right) \\
&\geq \kappa && \text{(by Theorem 5)},
\end{aligned}$$

where κ is a positive constant, independent of n. Hence, the probability of sampling an offspring in levels $A_{\geq j+1}$ in iteration $t+1$ is bounded from below by $z_j := \kappa$, and $z^* = \min_{j\in[m-1]}\{z_j\} = \kappa$.

Condition (G3)

We have $1/\delta^2 = O(1)$, and $m = O(\sqrt{n})$. Therefore, there must exist a constant $c > 0$ such that

$$\left(\frac{c}{\gamma_0}\right)\sqrt{n}\ln(n) \geq \left(\frac{4}{\gamma_0\delta^2}\right)\ln\left(\frac{128m}{z_*\delta^2}\right).$$

The requirement $\mu \geq c\sqrt{n}\ln(n)$ now implies that

$$\lambda = \frac{\mu}{\mu/\lambda} \geq \left(\frac{c}{\gamma_0}\right)\sqrt{n}\ln(n) \geq \left(\frac{4}{\gamma_0\delta^2}\right)\ln\left(\frac{128m}{z_*\delta^2}\right);$$

hence, condition (G3) is satisfied.

Having satisfied all three conditions and assuming that all marginals are at least $1/4$ during the whole run and recalling that $m = O(\sqrt{n})$, Theorem 4 guarantees the upper bound

$$O\left(\lambda\sum_{j=1}^{m}\frac{1}{z_j} + \sum_{j=1}^{m}\frac{1}{z_j}\right) = O(m\lambda) = O\left(\lambda\sqrt{n}\right).$$

To obtain the overall expected runtime $\mathbb{E}[T]$, we first recall that the probability for an arbitrary marginal to drop below $1/4$ during the n^{cc_1} first iterations is at most $O(n^{-cc_1})$ [127, Lemma 3.2]. By the union bound [95, Lemma 1.2], at least one in n marginals $p_{t,1}, p_{t,2}, \ldots, p_{t,n}$ drops below $1/4$ during the n^{cc_1} first iterations with probability at most

$\mathrm{O}\left(n^{-cc_1+1}\right)$, and thus the complementary event occurs with probability at least $1-\mathrm{O}\left(n^{-cc_1+1}\right)$, i.e., super-polynomially close to one for c large enough. In the latter event, the UMDA with offspring population sizes $\lambda = \Omega(\log n)$ and parent population sizes $\mu \leq \lambda/(13e(1+\delta))$ for any constant $\delta \in (0,1)$ is known to take an expected runtime of $\mathrm{O}(n\lambda \log \lambda)$ to optimise the ONEMAX function [23, Theorem 5]. By the law of total expectation (also called tower rule [95, Lemma 2.5])[1], the overall expected runtime is

$$\begin{aligned}\mathbb{E}[T] &= \mathrm{O}\left(\lambda\sqrt{n}\right)\cdot\left(1-\mathrm{O}\left(n^{-cc_1+1}\right)\right)+\mathrm{O}(n\lambda\log\lambda)\cdot\mathrm{O}\left(n^{-cc_1+1}\right)\\ &= \mathrm{O}\left(\lambda\sqrt{n}\right)+\mathrm{O}(n\lambda\log\lambda)\cdot\mathrm{O}\left(n^{-cc_1+1}\right).\end{aligned}$$

If the constant c is chosen sufficiently large, the second term will be subsumed by the first term, and the expected runtime is therefore $\mathbb{E}[T] = \mathrm{O}(\lambda\sqrt{n})$. ■

4.5 Conclusion

We have analysed the expected runtime of the the UMDA on the ONEMAX function to test the algorithm's ability as a hill climber. Two upper bounds of $\mathrm{O}(\lambda n)$ and $\mathrm{O}(\lambda\sqrt{n})$ were obtained for $\mu = \Omega(\log n)\cap\mathrm{O}(\sqrt{n})$ and $\mu = \Omega(\sqrt{n}\log n)$, respectively. Although our result assumed that $\lambda \geq (1+\beta)\mu$ for some positive constant $\beta > 0$, it did not require that $\lambda = \Theta(\mu)$ as in [127]. Note that if $\lambda = \Theta(\log n)$, a tight bound of $\Theta(n\log n)$ on the expected runtime of the UMDA on ONEMAX was obtained, matching the well-known tight bound of $\Theta(n\log n)$ for the $(1+1)$ EA on the class of linear functions [40]. Although we did not obtain a runtime bound when the parent population size is $\mu = \omega(\sqrt{n})\cap\mathrm{o}(\sqrt{n}\log n)$, our results finally closed the existing gap of $\Theta(\log\log n)$ between the best known upper bound of $\mathrm{O}(n\log n\log\log n)$ for $\lambda = \Omega(\mu)$ [23] and the relatively new lower bound of $\Omega(\mu\sqrt{n}+n\log n)$ for $\lambda = (1+\Theta(1))\mu$ [71].

[1] In this thesis, two names, the law of total expectation and the tower rule, are used interchangeably.

Chapter 5

UMDA and PBIL may cope well with epistasis

This chapter is based on two joint works, [77] and [79], both with Per Kristian Lehre. The results from these two papers have been heavily reworked for this chapter. Some proof has been simplified. The complex constraint on the parent population size μ and the smoothing parameter ρ for the PBIL *on the* LEADINGONES *function in [77, Theorem 3] has been stated more precisely.*

5.1 Open problems

In evolutionary computing, epistasis usually refers to the interplays between decision variables of a fitness function. It corresponds to the maximum number of other variables on which each of the n decision variables depends [25, 52]. Unlike the ONEMAX function in which variables are mutually independent, there exist many fitness functions which require the decision variables to cooperate in a particular way to have a positive impact on the overall fitness. In this situation, the algorithm's ability to cope with epistasis is essential for its success in tackling those problems. LEADINGONES (see Definition 3) is a typical benchmark function to test such an ability from a theoretical perspective. The function counts the number of leading 1s in the bitstring and has an epistasis degree of $n-1$. We are interested in the expected runtimes of the considered univariate EDAs on this function because it remains open whether the class of univariate EDAs that assume variable dependencies could efficiently

optimise the function. Friedrich et al. [47] pointed out that univariate EDAs might take at least $\Omega(n^2)$ function evaluations since these algorithms are not stable, and the function has many 'neutral' bits. Therefore, in this context, by efficiency, we usually mean an expected runtime of $\mathrm{O}\left(n^2\right)$ [47].

In 2015, Dang and Lehre [23], via the level-based theorem, obtained an upper bound of $\mathrm{O}\left(n\lambda\log\lambda+n^2\right)$ on the expected runtime for the UMDA with an offspring population size $\lambda=\Omega(\log n)$ and a parent population size $\mu\leq\lambda/(e(1+\delta))$ for some constant $\delta>0$ on the LEADINGONES function. For $\lambda=\Omega(\log n)\cap\mathrm{O}\left(n/\log n\right)$, the bound above becomes $\mathrm{O}\left(n^2\right)$. Under any other settings $\mu\geq\lambda/e$, it is still unknown whether the UMDA could optimise the function in expected polynomial runtime. Furthermore, we are also missing a lower bound on the expected runtime, that is necessary to understand how the algorithm copes with variable dependencies.

For the PBIL, the only rigorous analysis on the LEADINGONES function was published in 2017 [133]. The authors argued that the algorithm with a sufficiently large population size can avoid making wrong decisions early (so-called genetic drift) even when the smoothing parameter is large. They also showed an expected runtime of $\mathrm{O}\left(n\lambda\right)$ on the LEADINGONES function. And yet the required offspring population size still remains large, i.e., $\lambda=\Omega(n^{1+c})$ [133, Theorem 2]. It raises the question of whether a tighter upper bound of $\mathrm{O}\left(n^2\right)$ can be obtained for the PBIL on the LEADINGONES function. Furthermore, the answer to this question is of particular interest because it might be considered as the first step towards showing the potential advantage of incremental learning over the update mechanism used by the UMDA.

In this chapter, we shall show a lower bound of $\Omega(n\lambda/\log\lambda)$ on the expected runtime of the UMDA with $\lambda\geq e\mu/(1-\delta)^2$ for any constant $\delta\in(0,1)$ on the LEADINGONES function. On the other hand, a lower bound of $2^{\Omega(\mu^k)}$ for any constant $k\in(0,1)$ is proved when $\mu=o(n^{1/k})$ and $\lambda\leq\mu e^{1-\varepsilon}/(1+\delta)$ for any constants $\varepsilon\in(0,1)$, and $0<\delta\leq e^{1-\varepsilon}-1$ in expectation and with probability $1-2^{-\Omega(\mu)}-2^{-\Omega(n)}$. We also consider the PBIL and show that the algorithm finds the global optimum using at most $\mathrm{O}\left(n^2\right)$ function evaluations under appropriate parameter settings, assuming that all marginals stay above some constant during the whole run. In the end, we introduce prior noise to the fitness evaluation of the LEADINGONES function and show that the two algorithms can cope well with noise.

5.2 Useful tools

We first recall some well-known results that will be used frequently in the main proofs. By considering the sum of $n \in \mathbb{N}$ independent Bernoulli random variables, the following result shows that a constant factor deviation from the expectation happens with probability negatively exponential in the expectation.

Theorem 11 (Chernoff bounds [30]). *Let $X_1, X_2, \ldots, X_n$ be independent random variables taking values in $[0,1]$. Let $X = \sum_{i=1}^{n} X_i$. Then for all $\delta \in [0,1]$*

a) $\Pr(X \geq (1+\delta)\mathbb{E}[X]) \leq e^{-\delta^2 \mathbb{E}[X]/3}$.

b) $\Pr(X \leq (1-\delta)\mathbb{E}[X]) \leq e^{-\delta^2 \mathbb{E}[X]/2}$.

Furthermore, the next result implies that a deviation from the expectation by more than an additive term of $t \geq 0$ occurs with probability negatively exponential in t^2/n. This is a concentration inequality; other bounds have already been introduced in Section 4.2.

Theorem 12 (Chernoff-Hoeffding bound [30]). *Let $X_1, X_2, \ldots, X_n$ be independent random variables taking values in $[0,1]$. Let $X = \sum_{i=1}^{n} X_i$. Then for all $t \geq 0$*

a) $\Pr(X \geq \mathbb{E}[X] + t) \leq e^{-2t^2/n}$,

b) $\Pr(|X \leq \mathbb{E}[X] - t) \leq e^{-2t^2/n}$.

All inequalities above provide bounds on the probability of observing a particular value for a random variable by sampling from its probability distribution once. We now consider the scenario in which the distribution has been sampled multiple times. Let $X_1, X_2, \ldots, X_m$ be m independent and identically distributed (i.i.d.) random variables with the same cdf $F_X(x)$. We also let $x_1, x_2, \ldots, x_m$ be realisations of these random variables. The empirical distribution function is defined as follows.

$$\widehat{F}_m(x) := \frac{1}{m} \sum_{i=1}^{m} \mathbb{1}_{\{x_i \leq x\}}.$$

The following result is known as Dvoretzky–Kiefer–Wolfowitz (DKW) inequality due to [91]. It provides an estimate of how close the empirical distribution function is to the (true)

distribution function from which the samples are drawn. The result follows by replacing $\varepsilon = \varepsilon'\sqrt{m}$ into [91, Corollary 1].

Theorem 13 (DKW inequality)**.** *Let $X_1, X_2, \ldots, X_m$ be m i.i.d. real-valued random variables with a cdf $F_X(x)$. Let $\widehat{F}_m(x)$ be the empirical distribution function. It holds for any integer m and any $\varepsilon > 0$ that*

$$\Pr\left(\sup_{x\in\mathbb{R}} \left|\widehat{F}_m(x) - F_X(x)\right| > \varepsilon\right) \leq 2e^{-2m\varepsilon^2}.$$

We also need the following two elementary results.

Lemma 8 (AM-GM inequality [94])**.** *Let $a_1, \ldots, a_n$ be n non-negative real numbers. It holds that*

$$\sum_{i=1}^{n} \frac{a_i}{n} \geq \prod_{i=1}^{n} a_i^{\frac{1}{n}}.$$

Lemma 9 ([log])**.** *Let $n > 0$ and $x > 0$. Then $\ln(x) \leq n(x^{1/n} - 1)$.*

5.3 UMDA

Before we get to analysing the runtime, we introduce some notation. Let $C_{t,i}$ for all $i \in [n]$ denote the number of individuals having at least i leading 1s in iteration t, and let $D_{t,i}$ be the number of individuals having exactly $i-1$ leading 1s. For the special case $i = 1$, $D_{t,i}$ consists of individuals that do not have any leading 1s.

Once the population has been sampled in line 7 of Algorithm 3, the algorithm invokes truncation selection to select the μ fittest individuals (out of a population of λ) to update the probability vector. We take this μ-cutoff into account by defining a random variable

$$Z_t := \max\{i \in \{0\} \cup [n] : C_{t,i} \geq \mu\}, \tag{5.1}$$

which tells us how many marginals, counting from position one, are set to the upper border $1 - 1/n$ in iteration t. Furthermore, we define another random variable

$$Z_t^* := \max\{i \in \{0\} \cup [n] : C_{t,i} > 0\} \tag{5.2}$$

to be the number of leading 1s of the fittest individual(s).

On the distributions of $C_{t,i}$ and $D_{t,i}$

In order to analyse the distributions of the random variables $C_{t,i}$ and $D_{t,i}$, we shall take an alternative view on the sampling process at an arbitrary bit position $i \in [n]$ in iteration $t \in \mathbb{N}$ via the principle of deferred decisions[1] [95, p. 9]. We imagine that the process samples the values of the first bit for λ individuals. Once this has finished, it moves on to the second bit and so on until the population is sampled. To be more specific, we now look at the first bit in iteration t. The number of 1s sampled in the first bit position follows a binomial distribution with parameters λ and $p_{t,1}$, i.e., $C_{t,1} \sim \mathrm{Bin}(\lambda, p_{t,1})$. Thus, the number of 0s at the first bit position is $D_{t,1} = \lambda - C_{t,1}$. For completeness, we always assume that $C_{t,0} = \lambda$.

Having sampled the first bit for λ individuals, we note that the bias due to selection in the second bit position comes into play only if the first bit is a 1. If this is the case, then a one is more preferred to a 0 at the second-bit position. Among the $C_{t,1}$ fittest individuals, the probability of sampling a 1 at the second bit position is $p_{t,2}$; thus, the number of individuals having at least 2 leading 1s is binomially distributed with parameters $C_{t,1}$ and $p_{t,2}$, that is, $C_{t,2} \sim \mathrm{Bin}(C_{t,1}, p_{t,2})$, and the number of 0s equals $D_{t,2} = C_{t,1} - C_{t,2}$. Among the $D_{t,1}$ last individuals, since for these individuals the first bit is a 0, there is no bias between a 1 and a 0 at the second bit position, and the number of 1s sampled there follows a binomial distribution with parameters $D_{t,1} = \lambda - C_{t,1}$ and $p_{t,2}$.

We can generalise this result for an arbitrary bit position $i \in [n]$. The number of individuals having at least i leading 1s follows a binomial distribution with $C_{t,i-1}$ trials and a success probability $p_{t,i}$, i.e.,

$$C_{t,i} \sim \mathrm{Bin}(C_{t,i-1}, p_{t,i}), \tag{5.3}$$

and

$$D_{t,i} = C_{t,i-1} - C_{t,i} \sim \mathrm{Bin}(C_{t,i-1}, 1 - p_{t,i}). \tag{5.4}$$

Furthermore, the number of 1s sampled among the $\lambda - C_{t,i-1}$ remaining individuals is binomially distributed with $\lambda - C_{t,i-1}$ trials and a success probability $p_{t,i}$. Let $(\mathscr{F}_t : t \in \mathbb{N})$ be a filtration induced from the population $(P_t : t \in \mathbb{N})$ [125, p. 93]. If we consider the expectations of these random variables, by the tower property of conditional expectation (or tower rule [95, Theorem 2.7]) and the fact that $p_{t,i}$ is $\mathscr{F}_{t-1}$-measurable [125, p. 93], we then

[1] We were also inspired by the work in [127], which also applied the principle of deferred decisions, split the set of offspring into different types and argued that random variables follow binomial distributions with the number of certain individuals as first parameter.

get

$$\begin{aligned}\mathbb{E}[C_{t,i} \mid \mathscr{F}_{t-1}] &= \mathbb{E}[\mathbb{E}[C_{t,i} \mid C_{t,i-1}, \mathscr{F}_{t-1}] \mid \mathscr{F}_{t-1}] \\ &= \mathbb{E}[C_{t,i-1} \cdot p_{t,i} \mid \mathscr{F}_{t-1}] \\ &= \mathbb{E}[C_{t,i-1} \mid \mathscr{F}_{t-1}] \cdot p_{t,i},\end{aligned} \tag{5.5}$$

and similarly

$$\mathbb{E}[D_{t,i} \mid \mathscr{F}_{t-1}] = \mathbb{E}[C_{t,i-1} \mid \mathscr{F}_{t-1}] \cdot (1 - p_{t,i}). \tag{5.6}$$

By the end of this sampling process, we will obtain a population that is sorted in descending order according to the LEADINGONES-values.

5.3.1 An exponential runtime under low selection rates

Recall that we aim at showing that the UMDA takes an exponential runtime to optimise the LEADINGONES function when the selection rate is not sufficiently high, as required in [24, Theorem 7]. For any constant $\delta \in (0,1)$, we define two intermediate values

$$\alpha = \alpha(n) \coloneqq \log_{1-\frac{1}{n}} \frac{\mu}{\lambda(1-\delta)}, \tag{5.7}$$

$$\beta = \beta(n) \coloneqq \log_{1-\frac{1}{n}} \frac{\mu}{\lambda(1+\delta)}. \tag{5.8}$$

Clearly, we always get $\alpha \leq \beta$. We also define a stopping time $\tau \coloneqq \min\{t \in \mathbb{N} : Z_t \geq \alpha\}$ to be the first hitting time of the value α for the random variable Z_t. We then consider two phases:

(1) until the random variable Z_t hits the value α for the first time ($t \leq \tau$),

(2) after the random variable Z_t has hit the value α for the first time ($t > \tau$).

Phase 1: before the fitness of the μ-th fittest individual hits the threshold α

The algorithm starts with an initial population P_0 sampled from a uniform distribution $p_0 = (1/2, \ldots, 1/2)$. An initial observation is that the all-ones bitstring cannot be sampled in the population P_0 with overwhelming probability since the probability of sampling it from the uniform distribution is 2^{-n}, then by the union bound it appears in the population P_0 with

probability at most $\lambda \cdot 2^{-n} = 2^{-\Omega(n)}$ since we only consider the offspring population of size at most polynomial in the problem instance size n (i.e., $\lambda \in \text{poly}(n)$). The following lemma states the expectations of the random variables Z_0^* and Z_0 (see [81] for a similar result).

Lemma 10. $\mathbb{E}[Z_0] \leq \mathbb{E}[Z_0^*] \leq \log \lambda + 2$.

Proof. Because $Z_0 \leq Z_0^*$, we have $\mathbb{E}[Z_0] \leq \mathbb{E}[Z_0^*]$. We are left to bound the expectation $\mathbb{E}[Z_0^*]$. Recall that $Z_0^* = \max\{i : C_{0,i} > 0\}$ and let $f := \textsc{LeadingOnes}$. The probability of sampling an individual with more than k leading 1s (where $k < n$) is $\Pr(f(x) > k) = (1/2)^{k+1} = 2^{-(k+1)}$, thus $\Pr(f(x) \leq k) = 1 - \Pr(f(x) > k) = 1 - 2^{-(k+1)}$. The event $Z_0^* \leq k$ implies that the λ individuals all have at most k leading 1s, i.e.,

$$\Pr(Z_0^* \leq k) = \prod_{i=1}^{\lambda} \Pr(f(x_0^{(i)}) \leq k) = (1 - 2^{-(k+1)})^{\lambda},$$

and $\Pr(Z_0^* > k) = 1 - (1 - 2^{-(k+1)})^{\lambda}$. Because Z_0^* is an integer-valued random variable, by [95, Lemma 2.9] we then get

$$\begin{aligned}
\mathbb{E}[Z_0^*] &\leq \sum_{k=0}^{\infty} \Pr(Z_0^* > k) \\
&= \sum_{k=0}^{\infty} (1 - (1 - 2^{-(k+1)})^{\lambda}) \\
&\leq \log \lambda + \sum_{k=\log \lambda}^{\infty} (1 - (1 - 2^{-(k+1)})^{\lambda}) \\
&\leq \log \lambda + \lambda \cdot \sum_{k=\log \lambda}^{\infty} 2^{-(k+1)} && \text{(by Bernoulli's ineq. [30])} \\
&\leq \log \lambda + \lambda \cdot 2^{-\log \lambda + 1} \\
&= \log \lambda + 2.
\end{aligned}$$

which completes the proof. ■

Lemma 11. *It holds for any $t \in \mathbb{N}$ and $i \geq Z_t + 2$ that $X_{t,i} \sim \text{Bin}(\mu, p_{t,i})$.*

Proof. By the definition of the random variable Z_t, we know that $C_{t,Z_t} \geq \mu$ and $C_{t,Z_t+1} < \mu$. Consider bit position $j := Z_t + 2$. We then obtain from (6.4) that among the $C_{t,j-1}$ fittest individuals there are $C_{t,j} \sim \text{Bin}\left(C_{t,j-1}, p_{t,j}\right)$ individuals with at least j leading 1s. For the

$\mu - C_{t,j-1} > 0$ remaining individuals (among the μ fittest individuals), the overall fitness (or the fitness ranking) of these individuals have been already decided by the $j-1$ first bits, and what is sampled at (or after) bit position j (also called free-riders [40]) will not have any impact on the ranking of these individuals. In other words, there is no bias in bit j among these (remaining) individuals, which also means that the number of 1s sampled here follows a binomial distribution with $\mu - C_{t,j-1}$ trials and success probability $p_{t,j}$, i.e., $\text{Bin}\left(\mu - C_{t,j-1}, p_{t,j}\right)$. Thus, we get:

$$\begin{aligned} X_{t,j} &\sim C_{t,j} + \text{Bin}\left(\mu - C_{t,j-1}, p_{t,j}\right) \\ &\sim \text{Bin}\left(C_{t,j-1}, p_{t,j}\right) + \text{Bin}\left(\mu - C_{t,j-1}, p_{t,j}\right) \\ &\sim \text{Bin}\left(\mu, p_{t,j}\right). \end{aligned}$$

Because the distribution of $X_{t,j}$ depends only on $p_{t,j}$, the same line of arguments can be repeated for each of the remaining bit positions $Z_t + 3, \ldots, n$. The proof is now complete. ■

We now show that the value of the random variable Z_t does not decrease during phase 1 with high probability.

Lemma 12. $\Pr\left(\forall t \in [1,\tau] : Z_t \geq Z_{t-1}\right) \geq 1 - \tau \cdot 2^{-\Omega(\mu)}$.

Proof. It suffices to show that with probability at most $\tau \cdot 2^{-\Omega(\mu)}$ there exists an iteration $t \in [1,\tau]$ such that $Z_t < Z_{t-1}$. We first note that the value of the random variable Z_t drops in iteration $t+1$ only if the number of individuals with at least Z_t leading 1s in the next iteration is less than μ. Recall that $Z_t < \alpha$ for any $t < \tau$. The number of individuals with at least Z_t leading 1s, sampled in iteration $t+1$, follows a binomial distribution with λ trials and a success probability $(1-1/n)^{Z_t}$. Thus, in expectation the number of such individuals is

$$\lambda \left(1 - \frac{1}{n}\right)^{Z_t} \geq \lambda \left(1 - \frac{1}{n}\right)^{\alpha} = \frac{\mu}{1-\delta}.$$

By a Chernoff bound (see Theorem 11), the probability of sampling at most $(1-\delta)\cdot\mu/(1-\delta) = \mu$ such individuals is at most $e^{-(\delta^2/2)\cdot\mu/(1-\delta)} = 2^{-\Omega(\mu)}$ for any constant $\delta \in (0,1)$. By the union bound, this rare event happens at least once during the τ first iterations with probability at most $\tau \cdot 2^{-\Omega(\mu)}$, and the complementary event takes place with probability at least $1 - \tau \cdot 2^{-\Omega(\mu)}$, which completes the proof. ■

Phase 2: after the fitness of the μ-th individual has hit value α for the first time

By (5.1), the Z_t first marginals are set to the upper border $1-1/n$ in iteration $t \in \mathbb{N}$. Recall that the random variable $X_{t,i}$ denotes the number of 1s at bit position $i \in [n]$ among the μ fittest individuals, which is used to update the probabilistic model of the UMDA.

The preceding section shows that the random variable Z_t is non-decreasing during phase 1 with probability $1-\tau \cdot 2^{-\Omega(\mu)}$. The following lemma also shows that its value stays above α afterwards with high probability.

Lemma 13. *Recall α in (5.7). It holds for any constant $k \in [0,1)$ that*

$$\Pr(\forall t \in [\tau, \tau + 2^{\Omega(\mu^k)}] : Z_t \geq \alpha) \geq 1 - 2^{\Omega(\mu^k)} \cdot 2^{-\Omega(\mu)}.$$

Proof. Consider the worst-case scenario in which $Z_t = \alpha$ for some $t \in [\tau, \tau + 2^{\Omega(\mu^k)}]$. We also note that the value of the random variable Z_t drops below α in iteration $t+1$ if and only if the number of individuals with at least Z_t leading 1s sampled in the next iteration is less than μ. An offspring with at least α leading 1s is still sampled with probability $(1-1/n)^\alpha = \mu/(\lambda(1-\delta))$ for some constant $\delta \in (0,1)$, and by a Chernoff bound, there are at most μ such individuals sampled in the next iteration with probability at most $2^{-\Omega(\mu)}$. By the union bound, this happens at least once in the interval $[\tau, \tau + 2^{\Omega(\mu^k)}]$ with probability at most $2^{\Omega(\mu^k)} \cdot 2^{-\Omega(\mu)}$. The complementary event then occurs with probability of at least $1 - 2^{\Omega(\mu^k)} \cdot 2^{-\Omega(\mu)}$, which completes the proof. ■

The following lemma further shows that there is also an upper bound on the random variable Z_t. The proof uses the fact that the random variable Z_t exceeds the threshold β if and only if the number of individuals with more than β leading 1s sampled in the next iteration is no less than μ.

Lemma 14. *Recall β in (5.8). It holds for any constant $k \in [0,1)$ that*

$$\Pr(\forall t \in [0, 2^{\Omega(\mu^k)}] : Z_t \leq \beta) \geq 1 - 2^{\Omega(\mu^k)} \cdot 2^{-\Omega(\mu)}.$$

Proof. It suffices to show that with probability at most $2^{\Omega(\mu^k)} \cdot 2^{-\Omega(\mu)}$ there exists an iteration $t \in [0, 2^{\Omega(\mu^k)}]$ (for any small constant $\varepsilon \in (0,1)$) such that $Z_t > \beta$. Consider an arbitrary

iteration $t \in [0, 2^{\Omega(\mu^k)}]$. An individual with at least β leading 1s is sampled with probability

$$\prod_{i=1}^{\beta} p_{t,i} \leq \left(1 - \frac{1}{n}\right)^{\beta} = \frac{\mu}{\lambda(1+\delta)}$$

for some constant $\delta \in (0,1)$. Thus, the number of such individuals sampled in the next iteration will be stochastically dominated by $\mathrm{Bin}(\lambda, \mu/(\lambda(1+\delta)))$, and by Lemma 2, their expected number is at most $\mu/(1+\delta)$. By a Chernoff bound, the probability of sampling at least $(1+\delta)\cdot\mu/(1+\delta) = \mu$ such individuals in the next iteration is at most $2^{-\Omega(\mu)}$. By the union bound, this rare event happens at least once in the interval $[0, 2^{\Omega(\mu^k)}]$ with probability at most $2^{\Omega(\mu^k)} \cdot 2^{-\Omega(\mu)}$. Thus, the complementary event occurs with probability at least $1 - 2^{\Omega(\mu^k)} \cdot 2^{-\Omega(\mu)}$. The proof is then complete by noting that the value of the random variable Z_t exceeds β if and only if the number of individuals with more than β leading 1s is at least μ. ■

Lemma 13 and Lemma 14 together give essential insights about the behaviour of the algorithm. The random variable Z_t will stay well below the threshold β for $2^{\Omega(\mu^k)}$ iterations with probability $1 - 2^{-\Omega(\mu)} \cdot 2^{\Omega(\mu^k)}$ for a sufficiently large parent population size μ. More precisely, the random variable Z_t will move back and forth around an equilibrium value $\log_{1-1/n}(\mu/\lambda)$. This is because when $Z_t = \log_{1-1/n}(\mu/\lambda)$, in expectation there are exactly $\lambda(1-1/n)^{\log_{1-1/n}(\mu/\lambda)} = \lambda \cdot (\mu/\lambda) = \mu$ individuals having at least $\log_{1-1/n}(\mu/\lambda)$ leading 1s.

An exponential lower bound on the runtime is obtained if we can also show that the probability of sampling the $n-\beta$ last bits correctly is exponentially small. We now choose the selection rate μ/λ such that $n - \beta \geq \varepsilon n$ for any constant $\varepsilon \in (0,1)$, that is equivalent to $\beta \leq n(1-\varepsilon)$. By (5.8) and solving for μ/λ, we then obtain

$$\frac{\mu}{(1+\delta)\lambda} \geq \left(1 - \frac{1}{n}\right)^{n(1-\varepsilon)}.$$

The right-hand side is at most $1/e^{1-\varepsilon}$ as $(1-1/n)^n \leq 1/e$ for all $n > 0$ [96], so the above inequality always holds if the selection rate satisfies $\mu/\lambda \geq (1+\delta)/e^{1-\varepsilon}$ for any constants $\delta \in (0,1)$ and $\varepsilon \in (0,1)$.

The remainder of this section shows that the $n - (\beta+1) = \Omega(n)$ last bits cannot be sampled correctly in any polynomial number of iterations with high probability. We first

show that the sampling process among the $\Omega(n)$ last bits are mutually independent. To ease the analysis, we further define $Y_{t,1}, Y_{t,2}, \dots, Y_{t,n}$ to be n Bernoulli random variables representing an offspring sampled from the product distribution as in (2.3).

Lemma 15. *It holds for any $t \in \mathbb{N}$ that $(Y_{t,i} : i \in \{Z_t + 2, Z_t + 3, \dots, n\})$ are pairwise independent.*

Proof. We obtain by Lemma 11 that $X_{t,j} \sim \text{Bin}\left(\mu, p_{t,j}\right)$ for any $j \geq Z_t + 2$. In other words, the number of ones sampled at a bit position $j \geq Z_t + 2$ among the μ fittest individuals depends only on the marginal $p_{t,j}$. Thus, for any two distinct bit positions $j_1, j_2 \in \{Z_t + 2, \dots, n\}$, sampling a one at bit position j_1 is independent of sampling a one at bit position j_2. ■

Now consider an arbitrary bit position $i \geq \beta + 1$. We always get $\mathbb{E}[Y_{t,i} \mid \mathscr{F}_{t-1}] = p_{t,i}$, and by the tower property of conditional expectation we also obtain

$$\mathbb{E}[Y_{t,i}] = \mathbb{E}[\mathbb{E}[Y_{t,i} \mid \mathscr{F}_{t-1}]] = \mathbb{E}[p_{t,i}].$$

For the UMDA without borders, the stochastic process $(p_{t,i} : t \in \mathbb{N})$ is a martingale [47], which results in $\mathbb{E}[p_{t,i}] = p_{0,i} = 1/2$. We will show in the following lemma that for the UMDA with borders the expected value of a marginal at an arbitrary bit position $i \geq \beta + 2$ also stays at $1/2$ for any $t \in 2^{\Omega(\mu^k)}$ for some constant $k \in [0, 1)$.

Lemma 16. *Let $k \in [0, 1)$ be a constant. If there exists a constant $m < n$ such that $Z_t \leq m$ for any $t \leq 2^{\Omega(\mu^k)}$, then it holds for any $i \geq m + 2$ that $\mathbb{E}[p_{t,i}] = 1/2$.*

Proof. For readability, we omit the index i throughout the proof. Recall from Algorithm 3 that $p_t = \max\{1/n, \min\{1 - 1/n, X_{t-1}/\mu\}\}$. By the definition of expectation [95, Definition 2.3], we get

$$\mathbb{E}[p_t] = \frac{1}{n} \cdot \Pr\left(X_{t-1} = 0\right) + \left(1 - \frac{1}{n}\right) \cdot \Pr\left(X_{t-1} = \mu\right) + \sum_{k=1}^{\mu-1} \frac{k}{\mu} \cdot \Pr\left(X_{t-1} = k\right). \tag{5.9}$$

We note further that

$$\mathbb{E}[X_{t-1}] = \sum_{k=0}^{\mu} k \Pr(X_{t-1} = k) = \mu \Pr(X_{t-1} = \mu) + \sum_{k=1}^{\mu-1} k \Pr(X_{t-1} = k),$$

from which we then obtain

$$\sum_{k=1}^{\mu-1} k \cdot \Pr(X_{t-1} = k) = \mathbb{E}[X_{t-1}] - \mu \cdot \Pr(X_{t-1} = \mu). \tag{5.10}$$

Substituting (5.10) into (5.9) yields

$$\mathbb{E}[p_t] = \frac{1}{\mu} \cdot \mathbb{E}[X_{t-1}] + \frac{1}{n} \cdot (\Pr(X_{t-1} = 0) - \Pr(X_{t-1} = \mu)). \tag{5.11}$$

We are left to calculate the two probabilities that $X_{t-1} = 0$ and $X_{t-1} = \mu$. Since these are unconditional probabilities, we shall make no assumption (even on p_{t-1}) when calculating them. All we know are that $p_0 = 1/2$ and, by Lemma 11, X_{t-1} is binomially distributed with μ trials and a success probability p_{t-1}, which means that there is no bias towards any border in the stochastic process $(X_t : t \in \mathbb{N})$. Due to this symmetry, we get

$$\Pr(X_{t-1} = \mu) = \Pr(X_{t-1} = 0). \tag{5.12}$$

Furthermore, by the tower rule we also have

$$\mathbb{E}[X_{t-1}] = \mathbb{E}[\mathbb{E}[X_{t-1} \mid p_{t-1}]] = \mathbb{E}[\mu \cdot p_{t-1}] = \mu \cdot \mathbb{E}[p_{t-1}] \tag{5.13}$$

Substituting (5.12) and (5.13) into (5.11) yields $\mathbb{E}[p_t] = \mathbb{E}[p_{t-1}]$. Then by induction on time, we obtain

$$\mathbb{E}[p_t] = \mathbb{E}[p_{t-1}] = \mathbb{E}[p_{t-2}] = \ldots = \mathbb{E}[p_0] = 1/2,$$

which completes the proof. ■

Lemma 16 gives us insights into the expected values of the marginals at any time $t \in 2^{\Omega(\mu^k)}$. One should not confuse the expectation with the actual value of the marginals. Friedrich et al. [47] showed a similar result for the UMDA without border that even when the expectation stays at $1/2$, the actual value of the marginal in iteration t can be close to the trivial lower or upper border due to its large variance. Very recently, Doerr and Zheng [36] obtained a tight bound of $\Theta(\mu)$ on the first hitting time of any trivial border for these marginals.

Lemma 17. *Let $k \in [0,1)$ be some constant. Let $\mu/\lambda \geq (1+\delta)/e^{1-\varepsilon}$ for any constants $\delta \in (0,1)$ and $\varepsilon \in (0,1)$. Then, the $n-(\beta+1) = \Omega(n)$ last bit positions cannot be sampled as all 1s during the $2^{\Omega(\mu^k)}$ first iterations with probability $1-2^{\Omega(\mu^k)} \cdot 2^{-\Omega(n)}$.*

Proof. Given the selection rate $\mu/\lambda \geq (1+\delta)/e^{1-\varepsilon}$, by Lemma 14 we get $Z_t \leq \beta \leq n(1-\varepsilon)$ for all $t \leq 2^{\Omega(\mu^k)}$ with probability at least $1-2^{\Omega(\mu^k)} \cdot 2^{-\Omega(\mu)}$. We shall prove the lemma by looking at the $n-(\beta+2) \geq n-n(1-\varepsilon)-2 = \varepsilon n-2 = \Omega(n)$ last bit positions. Let us now consider the total number of zeros sampled at these bit positions in an iteration $t \leq 2^{\Omega(\mu^k)}$. We know by Lemma 16 (for $m = \beta+2$) that their marginals stay at $1/2$ in expectation, and we also know by Lemma 15 that the samples at these bit positions are mutually independent. Therefore, by linearity of expectation [95, Theorem 2.1], the expected number of zeros sampled there is

$$(n-(\beta+2))\left(1-\frac{1}{2}\right) = \Omega(n).$$

This means that to sample all ones at these bit positions there are still at least $\Omega(n)$ zeros to correct. In other words, we need to deviate a distance of $\Omega(n)$ below the expected value, and by a Chernoff-Hoeffding bound (see Theorem 12) such an event happens with probability at most

$$2 \cdot \exp\left\{-\frac{2(\Omega(n))^2}{n}\right\} = 2^{-\Omega(n)}.$$

By the union bound, this event happens at least once during the $2^{\Omega(\mu^k)}$ first iterations with probability still at most $2^{\Omega(\mu^k)} \cdot 2^{-\Omega(n)}$. The proof is now complete. ■

We are ready to show our main result of the UMDA on the LEADINGONES function.

Theorem 14. *Let $k \in [0,1)$ be some constant. The* UMDA *with a parent population size $\mu = o(n^{1/k})$ and an offspring population size λ such that $\mu \leq \lambda \leq \frac{e^{1-\varepsilon}}{1+\delta}\mu$ for any constants $\varepsilon \in (0,1)$ and $0 < \delta \leq e^{1-\varepsilon}-1$ has a runtime of $2^{\Omega(\mu^k)}$ on the* LEADINGONES *function with probability $1-2^{-\Omega(\mu)}-2^{-\Omega(n)}$ and also in expectation.*

Proof. We now consider the two phases as introduced above. Phase 1 ends after the stochastic process Z_t has hit the value of $\alpha < \beta$ for the first time. Since $\beta \leq n(1-\varepsilon)$ due to the given constraint on μ/λ, we know by Lemma 17 that the $n-(\beta+1) = \Omega(n)$ last bit positions cannot be sampled as all 1s during this phase with probability at least $1-\tau \cdot 2^{-\Omega(n)}$. We assume that the phase lasts for $\tau \leq 2^{\Omega(\mu^k)}$ iterations; otherwise, we are done, and the theorem holds trivially.

During both phases, we have observed by Lemma 14 that the random variable Z_t exceeds $\beta \leq n(1-\varepsilon)$ at least once during the $2^{\Omega(\mu^k)}$ first iterations with probability at most $2^{-\Omega(\mu^k)} \cdot 2^{-\Omega(\mu)} = 2^{-\Omega(\mu)}$ (since $k < 1$), while in the same period the $\Omega(n)$ last bits are sampled as all ones at least once with probability at most $2^{\Omega(\mu^k)} \cdot 2^{-\Omega(n)} = 2^{-\Omega(n)}$ due to Lemma 17 and $\mu = o(n^{1/k})$. Thus, by the union bound the all-ones bitstring can be sampled at least once during the $2^{\Omega(\mu^k)}$ first iterations with probability at most $2^{-\Omega(\mu)} + 2^{-\Omega(n)}$. The algorithm then takes at least $2^{\Omega(\mu^k)}$ iterations to optimise the function with probability at least $1 - 2^{-\Omega(\mu)} - 2^{-\Omega(n)}$.

By the law of total expectation, the expected runtime is at least $2^{\Omega(\mu^k)}(1 - 2^{-\Omega(\mu)} - 2^{-\Omega(n)}) = 2^{\Omega(\mu^k)}$, which completes the proof. ■

5.3.2 A runtime of $\Omega(n\lambda/\log\lambda)$ under high selection rates

When the selection rate is set too high such that the value of α, defined in (5.7), exceeds the problem instance size n, phase 1 will end when the μ fittest individuals are all-ones bitstrings. By (5.7), this case occurs when $\alpha \geq n$, equivalent to

$$\frac{\mu}{(1-\delta)\lambda} \leq \left(1 - \frac{1}{n}\right)^n$$

for any constant $\delta \in (0,1)$. The right-hand side is at least $(1-\delta)/e$ for any $n \geq (1+\delta)/\delta$ [80], and the above inequality always holds if we choose the selection rate $\mu/\lambda \leq (1-\delta)^2/e$, We now recall the best known upper bound on the expected runtime for the UMDA on the LEADINGONES function.

Theorem 15 (Theorem 4 [23])**.** *The* UMDA *with an offspring population size* $\lambda \geq c \log n$ *for some sufficiently large constant* $c > 0$ *and a parent population size* μ *such that* $\mu \leq \frac{\lambda}{e(1+\delta)}$ *for any constant* $\delta > 0$ *has an expected runtime of* $O\left(n\lambda \log\lambda + n^2\right)$ *on the* LEADINGONES *function.*

Until now, we are still missing a lower bound on the expected runtime for the UMDA on the LEADINGONES function, and in this section, we aim at deriving such a lower bound. Recall that the random variable Z_t, defined in (5.1), denotes the number of marginals, counting from position one, are set to the upper border $1 - 1/n$ in iteration t, and the random variable Z_t^*, defined in (5.2), denotes the fitness value of the fittest individual. The following lemma shows the expected difference between these two random variables in an arbitrary iteration

$t \in \mathbb{N}$. We pessimistically assume that the Z_t first marginals are all set to one since we are only interested in a lower bound and this will speed up the optimisation process.

Lemma 18. *It holds for any $t \in \mathbb{N}$ that $\mathbb{E}[Z_t^* - Z_t] \leq 2 + \log \mu$.*

Proof. Let $\delta_t := Z_t^* - Z_t$. Consider the bit positions $Z_t + 2, Z_t + 3, \ldots, n$ among the μ fittest individuals. We shall view this as an abstract population of μ individuals, each of length $n - (Z_t + 1)$, and also let $\delta_t' := Z_t^* - (Z_t + 1) = \delta_t - 1$. In other words, δ_t' is a random variable describing the number of leading 1s of the fittest individual in this abstract population. We first note that if $X_{t,Z_t+1} = 0$, then $Z_t^* = Z_t$ and $\delta_t = 0$. By the law of total expectation, we get

$$\begin{aligned}
\mathbb{E}[\delta_t \mid Z_t] &= \overbrace{\mathbb{E}\left[\delta_t \cdot \mathbb{1}_{\{X_{t,Z_t+1}=0\}} \mid Z_t\right]}^{=0} + \mathbb{E}\left[\delta_t \cdot \mathbb{1}_{\{X_{t,Z_t+1}>0\}} \mid Z_t\right] \\
&= \mathbb{E}\left[(1+\delta_t') \cdot \mathbb{1}_{\{X_{t,Z_t+1}>0\}} \mid Z_t\right] \\
&\leq 1 + \mathbb{E}\left[\delta_t' \cdot \mathbb{1}_{\{X_{t,Z_t+1}>0\}} \mid Z_t\right].
\end{aligned}$$

We are left to calculate the last conditional expectation. Consider again the abstract population introduced above. The probability of sampling at most k leading 1s in this population is $1 - \prod_{i=Z_t+2}^{(Z_t+2)+k} p_{t,i}$, and the probability that all μ individuals in the abstract population have more than k leading 1s is

$$1 - \left(1 - \prod_{i=Z_t+2}^{(Z_t+2)+k} p_{t,i}\right)^{\mu}.$$

Because δ_t' is an integer-valued random variable, we then get

$$\mathbb{E}\left[\delta_t' \cdot \mathbb{1}_{\{X_{t,Z_t+1}>0\}} \mid Z_t, p_{t,Z_t+2}, \ldots, p_{t,n}\right] \leq \sum_{k=0}^{\infty} \left(1 - \left(1 - \prod_{i=Z_t+2}^{(Z_t+2)+k} p_{t,i}\right)^{\mu}\right).$$

We know, by Lemma 16, that the values of the marginals $p_{t,i}$ for each $i \geq Z_t + 2$ stay at $1/2$ in expectation and also, by Lemma 15, that the samples at these bit positions are pairwise

independent. Thus,

$$
\begin{aligned}
&\mathbb{E}\left[\delta'_t \cdot \mathbb{1}_{\{X_{t,Z_t+1}>0\}} \mid Z_t\right] \\
&= \mathbb{E}\left[\mathbb{E}\left[\delta'_t \cdot \mathbb{1}_{\{X_{t,Z_t+1}>0\}} \mid Z_t, p_{t,Z_t+2}, \ldots, p_{t,n}\right] \mid Z_t\right] && \text{(by the tower rule)} \\
&\leq \sum_{k=0}^{\infty}\left(1-\mathbb{E}\left[\left(1-\prod_{i=Z_t+2}^{(Z_t+2+k)} p_{t,i}\right)^{\mu} \mid Z_t\right]\right) && \text{(by linearity of expectation)} \\
&\leq \sum_{k=0}^{\infty}\left(1-\left(1-\mathbb{E}\left[\prod_{i=Z_t+2}^{(Z_t+2+k)} p_{t,i} \mid Z_t\right]\right)^{\mu}\right) && \text{(see below)} \\
&\leq \sum_{k=0}^{\infty}\left(1-\left(1-\prod_{i=Z_t+2}^{(Z_t+2+k)} \mathbb{E}\left[p_{t,i} \mid Z_t\right]\right)^{\mu}\right) && \text{(by independence, Lemma 15)} \\
&= \sum_{k=0}^{\infty}(1-(1-2^{-(k+1)})^{\mu}) && \text{(by Lemma 16)} \\
&\leq \log\mu + 2. && \text{(by the proof of Lemma 10)}
\end{aligned}
$$

The second inequality follows from the fact that $x \mapsto (1-x)^{\mu}$ is a convex function for any $x \in [0,1]$, so by Jensen's inequality [95, Theorem 2.4] we get $\mathbb{E}[(1-x)^{\mu}] \geq (1-\mathbb{E}[x])^{\mu}$. By the tower rule, we then obtain $\mathbb{E}[Z_t^* - Z_t] = \mathbb{E}[\mathbb{E}[Z_t^* - Z_t \mid Z_t]] \leq \mathbb{E}[\log\mu + 2] = \log\mu + 2$, which completes the proof. ■

Lemma 18 tells us that the two random variables Z_t and Z_t^* only differ by a logarithmic additive term at any point in time in expectation. The global optimum is found when the random variable Z_t^* reaches the value of n. We can therefore alternatively analyse the random variable Z_t instead of Z_t^*. In other words, the random variable Z_t, starting from an initial value Z_0 given in Lemma 10, has to travel an expected distance of $n-(\log\mu-2)-Z_0$ (at bit positions) before the global optimum is found. We shall apply the additive drift theorem (for a lower bound, see Theorem 1) for a potential function $\phi(x) = n-x$ on the stochastic process $(Z_t : t \in \mathbb{N})$. The single-step change is

$$\Delta_t := \phi(Z_t) - \phi(Z_{t+1}) = Z_{t+1} - Z_t.$$

Theorem 16. *The* UMDA *with a parent population size μ and an offspring population size λ such that $\lambda \geq \frac{e}{(1-\delta)^2}\mu$ for any constant $\delta \in (0,1)$ has an expected runtime of $\Omega\left(\frac{n\lambda}{\log\lambda}\right)$ on the* LEADINGONES *function.*

Proof. Let $i := Z_t + 1$. By the definition of Z_t, $Z_{t+1} = Z_t$ and $\Delta_t = 0$ if there are less than μ individuals with at least i leading 1s sampled in the next iteration (i.e., $C_{t+1,i} < \mu$). Thus, the drift is maximised when $C_{t+1,i} \geq \mu$. By the law of total expectation, we then get

$$\begin{aligned}\mathbb{E}[\Delta_t \mid Z_t] &= \overbrace{\mathbb{E}\left[\Delta_t \cdot \mathbb{1}_{\{C_{t+1,i}<\mu\}} \mid Z_t\right]}^{=0} + \mathbb{E}\left[\Delta_t \cdot \mathbb{1}_{\{C_{t+1,i}\geq\mu\}} \mid Z_t\right] \\ &= \mathbb{E}[\Delta_t \mid C_{t+1,i} \geq \mu, Z_t] \cdot \Pr(C_{t+1,i} \geq \mu \mid Z_t) \\ &\leq \mathbb{E}[\Delta_t \mid C_{t+1,i} \geq \mu, Z_t] \\ &= \mathbb{E}[Z_{t+1} \mid C_{t+1,i} \geq \mu, Z_t] - Z_t.\end{aligned}$$

We are left to bound the expectation. Given Z_t, we know by Lemma 16 that the marginals of bit positions from $Z_t + 2$ to n stay at $1/2$ in expectation, and also by Lemma 11 the samples at these bit positions are pairwise independent. By following the proof of Lemma 18, we can quickly upper bound the required expectation as follows.

$$\mathbb{E}[Z_{t+1} \mid C_{t+1,i} \geq \mu, Z_t] \leq Z_t + 3 + \log\lambda.$$

Then, the drift is

$$\mathbb{E}[\Delta_t \mid Z_t] \leq \log\lambda + 3.$$

Because the random variable Z_t has to travel an expected distance of $n - (\log\mu + 2) - Z_0$ before the global optimum is found for the first time, by the additive drift theorem (see Theorem 1) the expected number of iterations, conditional on Z_0, is upper bounded by $(n - \log\mu - 2 - Z_0)/(\log\lambda + 3)$. Note that $\mathbb{E}[Z_0] \leq \log\lambda + 2$, there are λ function evaluations performed in each iteration, and by the tower rule, we then obtain an overall expected runtime of at least

$$\lambda \cdot \frac{n - (\log\mu + 2) - (\log\lambda + 2)}{\log\lambda + 3} = \Omega\left(\frac{n\lambda}{\log\lambda}\right),$$

which completes the proof. ■

How tight is the lower bound? We obtain an upper bound of $O\left(n^2\right)$ for $\lambda = \Omega(\log n) \cap O(n/\log n)$ and an upper bound of $O(n\lambda\log\lambda)$ for any $\lambda = \omega(n/\log n)$ in [23], while the lower bound above becomes $\Omega(n\lambda/\log\lambda)$. Thus, there is still a gap between the upper and the lower bound. For larger population sizes $\lambda \geq \mu = \Omega(n\log n)$, Doerr & Krejca [34] have very recently shown a tight bound of $\Theta(n\lambda/\log(\lambda/\mu))$ on the expected runtime of

the UMDA on the LEADINGONES function. The new bound improves the best known upper bound of $O(n\lambda \log \lambda)$ in [23] by removing the logarithmic factor of $\log \lambda$. Our lower bound above matches the bound in [34] (and thus is tight) for any population sizes $\mu = \Omega(n \log n)$ and $\lambda/\mu = \Omega(n^{\varepsilon})$ for any constant $\varepsilon > 0$. In this situation, the UMDA finds the global optimum of the LEADINGONES function using $\Theta(n\lambda/\log n)$ function evaluations in expectation.

5.4 Epistasis and Noise

We also consider a prior noise model and formally define the problem for any constant $0 < p < 1$ as follows.

$$F(x_1,\dots,x_n) = \begin{cases} f(x_1,\dots,x_n), & \text{with probability } 1-p, \text{ and} \\ f(\dots,1-x_i,\dots), & \text{with probability } p, \text{ where } i \sim \text{Unif}([n]). \end{cases}$$

We denote F as the noisy fitness and f as the actual fitness. For simplicity, we also denote P_t as the population prior to noise. The same noise model is studied in [48, 22] for population-based EAs on the ONEMAX and LEADINGONES functions.

We shall make use of the level-based theorem and first partition the search space $\mathscr{X}$ into $n+1$ disjoint subsets $A_0,\dots,A_n$, where each subset A_j contains bitstrings with j leading 1s.

$$A_j := \{x \in \mathscr{X} : \text{LEADINGONES}(x) = j\}. \tag{5.14}$$

Thus, there are $n+1$ levels, ranging from A_0 to A_n. Recall that $A_{\geq j} = \cup_{i=j}^{n} A_i$. We then need to verify three conditions (G1), (G2) and (G3) of the level-based theorem, where due to the presence of noise we choose the parameter $\gamma_0 = \mu/(\lambda(1-\delta)(1-p))$ for any constant $\delta \in (0,1)$ and the selection rate μ/λ to leverage the impact of noise in our analysis. The following lemma tells us the number of individuals in the population in iteration t which have fitness $F(x) = f(x) \geq j$.

Lemma 19. *Assume that $|P_t \cap A_{\geq j}| \geq \gamma_0 \lambda$, where $\gamma_0 = \mu/(\lambda(1-p)(1-\delta))$ for some constant $\delta \in (0,1)$, and μ/λ is assumed constant. Then*

(i) *there are at least* μ *individuals with the fitness* $F(x) = f(x) \geq j$ *in the noisy population with probability* $1 - 2^{-\Omega(\mu)}$*, and*

(ii) *there are at most* $\varepsilon\mu$ *individuals with actual fitness* $f(x) \leq j-1$ *and noisy fitness* $F(x) \geq j$ *for some small constant* $\varepsilon \in (0,1)$ *with probability* $1 - 2^{-\Omega(\mu)}$.

Proof. We first prove **(i)**. We take an alternative view on the sampling of the population and the application of noise. More specifically, we first sample the population, sort it in descending order according to the true fitness, and then noise occurs at any individual with probability p. Because noise does not occur at an individual with probability $1-p$, amongst the $\gamma_0\lambda$ individuals in levels $A_{\geq j}$, in expectation there are

$$(1-p)\gamma_0\lambda = \frac{(\mu/\lambda)\lambda}{1-\delta} = \frac{\mu}{1-\delta}$$

individuals unaffected by noise. Furthermore, by a Chernoff bound, there are at least $(1-\delta)\cdot\mu/(1-\delta) = \mu$ such individuals for some constant $0 < \delta < 1$ with probability at least $1 - e^{-(\delta^2/2)\cdot\mu/(1-\delta)} = 1 - 2^{-\Omega(\mu)}$, which proves the statement **(i)**.

For the statement **(ii)**, we only consider individuals with actual fitness $f(x) < j$ and noisy fitness $F(x) \geq j$ in the population. If such an individual is selected when updating the model, it will introduce a 0 to the total number of 0s among the μ fittest individuals for the j first bits. Let B denote the number of such individuals. There are at most $(1-\gamma_0)\lambda$ individuals with actual fitness $f(x) < j$, so the probability that their noisy fitness values are at least $F(x) \geq j$ is at most p/n because the leftmost zero must be flipped in the prior noise model. Hence the expected number of these individuals is upper bounded by

$$\mathbb{E}[B] \leq \frac{(1-\gamma_0)\lambda p}{n} < \frac{\lambda p}{n}. \tag{5.15}$$

We now show by a Chernoff bound that the event $B \geq \varepsilon\mu$ for a small constant $\varepsilon \in (0,1)$ occurs with probability at most $2^{-\Omega(\mu)}$. We shall rely on the fact that $\lambda p/n \leq \mu\varepsilon/2$ for sufficiently large n, which follows from the assumption $\mu/\lambda = \mathrm{O}(1)$. We use the parameter $\delta := \varepsilon\mu/\mathbb{E}[B] - 1$, which by (5.15) and the assumption $\lambda p/n \leq \varepsilon\mu/2$ satisfies $\delta \geq \varepsilon\mu n/(p\lambda) - 1 \geq 1$. We also have the lower bound

$$\delta\cdot\mathbb{E}[B] = \varepsilon\mu - \mathbb{E}[B] \geq \varepsilon\mu - \frac{\lambda p}{n} \geq \frac{\varepsilon\mu}{2}.$$

A Chernoff bound now gives the desired result

$$\Pr(B \geq \varepsilon\mu) = \Pr(B \geq (1+\delta)\mathbb{E}[B]) \leq e^{-\delta\mathbb{E}[B]/3} = e^{-\varepsilon\mu/6} = 2^{-\Omega(\mu)}, \tag{5.16}$$

which completes the proof. ■

We now derive upper bounds on the expected runtime of the UMDA on LEADINGONES in the noisy environment.

Theorem 17. *Consider a prior noise model with* $O(1) \ni p \in (0,1)$. *The* UMDA *with two population sizes* λ *and* μ *such that* $\lambda \geq \max\{c\log n, e\mu^2(1+\delta)\}$ *and* $\mu/\lambda = O(1)$ *for some sufficiently large constant* $c > 0$ *and any constant* $\delta \in (0,1)$ *has an expected runtime of* $O\left(n\lambda\log\lambda + n^2\right)$ *on the* LEADINGONES *function.*

Proof. We will apply the level-based theorem. Each level A_j for $j \in [n] \cup \{0\}$ is formally defined as in (5.14), and there are a total of $m = n+1$ levels.

Condition (G1)

Assume that $|P_t \cap A_{\geq j}| \geq \gamma_0\lambda$. We are required to show that the probability of sampling an offspring in levels $A_{\geq j+1}$ in iteration $t+1$ is lower bounded by a value z_j. For any constant $\delta \in (0,1)$, we choose $\gamma_0 = \mu/(\lambda(1-\delta)(1-p))$. Note that $\gamma_0 = \mathrm{O}(1)$ because $\mu/\lambda = \mathrm{O}(1)$ and $p = \mathrm{O}(1)$. For convenience, we also partition the noisy population into four groups:

1. Individuals with fitness $f(x) \geq j$ and $F(x) \geq j$.
2. Individuals with fitness $f(x) \geq j$ and $F(x) < j$.
3. Individuals with fitness $f(x) < j$ and $F(x) \geq j$.
4. Individuals with fitness $f(x) < j$ and $F(x) < j$.

By Lemma 19, there are at least μ individuals in group 1 with probability $1 - 2^{-\Omega(\mu)}$. The algorithm selects the μ fittest individuals according to the noisy fitness values to update the probabilistic model. Hence, unless the mentioned event does not happen, no individuals from group 2 or group 4 will be included when updating the model.

We are now going to analyse how individuals from group 3 impact the marginal probabilities. Let B denote the number of individuals in group 3. We pessimistically assume that

the algorithm uses all of the B individuals in group 3 and $\mu - B$ individuals are chosen from group 1 when updating the model. For all $i \in [j]$, let Q_i be the number of individuals in group 3 which have 1s at bit positions 1 through j, except for one position i where they have a 0. By definition, we then have $\sum_{i=1}^{j} Q_i = B$. The marginal probabilities after updating the model are

$$p_{t+1,i} = \begin{cases} 1 - Q_i/\mu, & \text{if } Q_i > 0, \\ 1 - Q_i/\mu - 1/n, & \text{if } Q_i = 0. \end{cases} \tag{5.17}$$

Again by Lemma 1, we can lower bound the probability of sampling an offspring x with actual fitness $f(x) \geq j$, by

$$\prod_{i=1}^{j} p_{t+1,i} \geq \prod_{i=1}^{j} q_i, \tag{5.18}$$

which holds for any vector $q := (q_1, \ldots, q_j)$ that majorises the vector $(p_{t+1,1}, \ldots, p_{t+1,j})$. By Definition 7, we construct such a vector q as follows.

$$q_i = \begin{cases} 1 - 1/n, & \text{if } i < j, \\ \sum_{k=1}^{j} p_{t+1,k} - (1-1/n)(j-1), & \text{if } i = j. \end{cases}$$

We now show that with high probability, the vector element q_j stays within the interval $[1 - 1/n - \varepsilon, 1 - 1/n]$, i.e., we show that q_j is indeed a probability. Since $p_{t+1,i} \leq 1 - 1/n$ for all $i \leq j$, we have the upper bound $q_j \leq (1-1/n)j - (1-1/n)(j-1) = 1 - 1/n$. For the lower bound, we note from (5.17) that $p_{t+1,i} \geq 1 - Q_i/\mu - 1/n$ for all $i \leq j$ and any $Q_i \geq 0$, so we also obtain

$$\begin{aligned} q_j &\geq \sum_{k=1}^{j} \left(1 - \frac{Q_i}{\mu} - \frac{1}{n}\right) - \left(1 - \frac{1}{n}\right)(j-1) \\ &= 1 - \frac{1}{n} - \sum_{k=1}^{j} \frac{Q_i}{\mu} \\ &= 1 - \frac{1}{n} - \frac{B}{\mu}. \end{aligned}$$

By Lemma 19, we have $B \leq \varepsilon\mu$ for some constant $\varepsilon \in (0,1)$ with probability $1 - 2^{-\Omega(\mu)}$. Assume that this high-probability event actually happens, we therefore have $q_j \geq 1 - 1/n - \varepsilon$. From this result, the definition of the vector q and (5.18), we can conclude that the probability

of sampling in iteration $t+1$ an offspring x with actual fitness $f(x) \geq j$ is

$$\prod_{i=1}^{j} p_{t+1,i} \geq \prod_{i=1}^{j} q_i \geq \left(1-\frac{1}{n}\right)^{j-1}\left(1-\frac{1}{n}-\varepsilon\right) \geq \frac{1-\varepsilon-o(1)}{e} = \Omega(1)$$

since $(1-1/n)^{j-1} \geq 1/e$ for any $n > 0$. Because we also have $p_{t+1,j+1} \geq 1/n$, the probability of sampling an offspring in levels $A_{\geq j+1}$ is at least $\Omega(1)\cdot(1/n) = \Omega(1/n)$. Thus, the condition (G1) holds with $z_j = \Omega(1/n)$ and $z_* = \min_{j\in[m-1]}\{z_j\} = \Omega(1/n)$.

Condition (G2)

Assume further that $|P_t \cap A_{\geq j+1}| \geq \gamma\lambda$ for all $\gamma \in (0,\gamma_0]$. We are also required to show that the probability of sampling an offspring in levels $A_{\geq j+1}$ is at least $(1+\delta)\gamma$ for some constant $\delta \in (0,1)$. Because the marginal $p_{t+1,j+1}$ can be lower bounded by $\gamma\lambda/\mu$, the above probability can be written as follows.

$$\prod_{i=1}^{j+1} p_{t+1,i} \geq p_{t+1,j+1}\cdot\prod_{i=1}^{j} p_{t+1,i} \geq \frac{\gamma\lambda}{\mu}\cdot\frac{1-\varepsilon-o(1)}{e} \geq (1+\delta)\gamma,$$

where by choosing

$$\frac{\mu}{\lambda} \leq \frac{1-\varepsilon-o(1)}{e(1+\delta)} = \frac{1-\varepsilon'}{e(1+\delta)}$$

for some constants $\delta,\varepsilon \in (0,1)$ and some other constant $\varepsilon' \in (0,1)$. Thus, the condition (G2) of the level-based theorem is verified.

Condition (G3)

We are required to choose an offspring population size such that

$$\lambda \geq \frac{4}{\gamma_0\delta^2}\ln\left(\frac{128m}{\delta^2\cdot\min_j\{z_j\}}\right) = \frac{4(1-\delta)(1-p)}{(\mu/\lambda)\delta^2}\ln\left(\frac{128m}{\delta^2\cdot z_*}\right).$$

Since $\mu/\lambda = \mathrm{O}(1)$, $p = \mathrm{O}(1), m = \mathrm{O}(n)$ and $1/z_* = \mathrm{O}(n)$, this condition can be satisfied by choosing $\lambda \geq c\log n$ for some sufficiently large constant $c > 0$.

Having verified the three conditions (G1), (G2) and (G3), and noting that $\ln(\delta\lambda/(4+\delta z_j)) < \ln(3\delta\lambda/2)$, the level-based theorem now guarantees an upper bound of

$$\mathrm{O}\left(n\lambda\log\lambda + n^2\right).$$

Note that, throughout the proof, we always assume the occurrence of the following two events in each iteration (see Lemma 19):

(i) the number of individuals in group 1 is at least μ with probability $1-2^{-\Omega(\mu)}$,

(ii) the number of individuals in group 3 is $B \leq \varepsilon\mu$ for some constant $\varepsilon \in (0,1)$ with probability $1-2^{-\Omega(\mu)}$.

By the union bound, either or both of these two events happen in an iteration $t \in \mathbb{N}$ with probability at most $2^{-\Omega(\mu)} + 2^{-\Omega(\mu)} = 2^{-\Omega(\mu)}$. The complementary event occurs with probability at least $1-2^{-\Omega(\mu)}$. We shall refer to these two events as the unlikely and likely events, respectively. In the latter event, we have shown an (conditional) expected runtime of $\mathrm{O}\left(n\lambda\log\lambda + n^2\right)$.

When the unlikely event occurs, under the assumption of the condition (G1) the j-th marginal q_j could be as small as the lower border $1/n$, which results in $\prod_{i=1}^{j} p_{t+1,i} \geq (1-1/n)^{j-1} \cdot (1/n) = \Omega(1/n)$, and the probability of sampling an offspring in the levels $A_{\geq j+1}$ is at least $z_j = \Omega(1/n^2)$. For the condition (G2), we could have in the worst case $q_j \geq \gamma\lambda/\mu \geq 1/\mu$ as $|P_t \cap A_{\geq j+1}| \geq \gamma\lambda \geq 1$, and an offspring in levels $A_{\geq j+1}$ is sampled in iteration $t+1$ with probability at least

$$\prod_{i=1}^{j+1} p_{t+1,i} \geq \frac{1}{e} \cdot \frac{1}{\mu} \cdot \frac{\gamma\lambda}{\mu} \geq \frac{\gamma\lambda}{e\mu^2} \geq \gamma(1+\delta),$$

assuming that $\lambda/e\mu^2 \geq 1+\delta$, equivalent to $\lambda \geq e\mu^2(1+\delta)$ for some constant $\delta \in (0,1)$. Putting things together, the (conditionally) expected runtime of the UMDA on the noisy LeadingOnes function in the unlikely event is

$$\mathrm{O}\left(n\lambda\log\lambda + \sum_{j=1}^{m-1}(1/z_j)\right) = \mathrm{O}\left(n\lambda\log\lambda + n^3\right).$$

To obtain the overall expected runtime $\mathbb{E}[T]$, we then follow a similar augmentation put forward in Theorem 10. By the law of total expectation and by choosing a sufficiently large constant $c > 0$, we get

$$\begin{aligned}\mathbb{E}[T] &= (1-2^{-\Omega(\mu)}) \cdot \mathrm{O}\left(n\lambda\log\lambda + n^2\right) + 2^{-\Omega(\mu)} \cdot \mathrm{O}\left(n\lambda\log\lambda + n^3\right) \\ &= \mathrm{O}\left(n\lambda\log\lambda + n^2\right),\end{aligned}$$

which completes the proof. ■

5.5 Extension to PBIL

In this section, we aim at improving the upper bound of $\mathrm{O}\left(n^{2+c}\right)$ in [133] for the PBIL on the LEADINGONES function. We shall apply the level-based theorem (see Theorem 4). To begin with, we first remark that Algorithm 5 assumes a mapping $\mathscr{D}^*$ from the space of populations $\mathscr{X}^\lambda$ to the space of probability distributions over the search space. The mapping $\mathscr{D}^*$ is often said to depend on the current population only [20]; however, it is not always necessary, especially for the PBIL with a sufficiently large offspring population size λ. The rationale behind this is that in each iteration the PBIL draws λ individuals from the product distribution, specified in (2.3). If the number of samples λ is sufficiently large, it is very unlikely that the many empirical frequencies deviate far from the true marginals. We will make this intuition more rigorous via the DKW inequality (see Theorem 13).

We use the level definition in (5.14) and seek to verify three conditions in Theorem 4. Recall that $A_{\geq j} = \cup_{i=j}^{n} A_i$. For conditions (G1) and (G2), we assume that there are at least $\gamma_0\lambda$ individuals in levels $A_{\geq j}$ in iteration t. Following [23], we choose γ_0 to be the selection rate, i.e., $\gamma_0 = \mu/\lambda$. This implies that the μ fittest individuals have at least j leading 1s. We define

$$\widehat{p}_{t,i} \coloneqq \frac{1}{\lambda}\sum_{j=1}^{\lambda} x_{t,i}^{(j)}$$

to be the frequency of ones at bit position i in the entire population of λ individuals. We now show under the assumption of the condition (G1) that if the population size is $\lambda = \Omega(\log n)$, the j first marginals cannot be too close to the lower border $1/n$ with high probability.

Lemma 20. *Assume that* $|P_t \cap A_{\geq j}| \geq \gamma_0\lambda$ *and* $\lambda \geq c((1+1/\varepsilon)/\gamma_0)^2 \ln(n)$ *for any constants* c, $\varepsilon > 0$ *and* $\gamma_0 \in (0,1)$, *then*

(i) $p_{t,i} \geq \gamma_0/(1+\varepsilon)$ *for an arbitrary bit* $i \in [j]$ *with probability at least* $1-2n^{-2c}$,

(ii) $\prod_{i=1}^{j} p_{t,i} \geq \gamma_0/(1+\varepsilon)$ *with probability at least* $1-2n^{-2c}$, *and*

(iii) $p_{t,i} \geq \gamma_0/(1+\varepsilon)$ *for all* $i \in [n]$ *with probability at least* $1-2n^{-2c+1}$.

Proof. We first show the statement **(i)**. Consider an arbitrary bit $i \in [j]$. Let Q_i be the number of ones sampled at bit position i in the entire population P_t, and the corresponding empirical cumulative distribution function of the number of zeros is

$$\widehat{F}_\lambda(0) = \frac{1}{\lambda}\sum_{j=1}^{\lambda} \mathbb{1}_{\{x_{t,i}^{(j)}=0\}} = \frac{\lambda - Q_i}{\lambda} = 1 - \widehat{p}_{t,i}.$$

Since the true cumulative distribution function is $F(0) = 1 - p_{t,i}$, we get by the DKW inequality (see Theorem 13) for all $\phi > 0$ that

$$\Pr\left(\widehat{p}_{t,i} - p_{t,i} > \phi\right) \leq \Pr\left(|\widehat{p}_{t,i} - p_{t,i}| > \phi\right) \leq 2e^{-2\lambda\phi^2}.$$

This means that, with probability at least $1 - 2e^{-2\lambda\phi^2}$, we obtain $\widehat{p}_{t,i} - p_{t,i} \leq \phi$, which is equivalent to $p_{t,i} \geq \widehat{p}_{t,i} - \phi \geq \gamma_0 - \phi$ since $\widehat{p}_{t,i} \geq \gamma_0\lambda/\lambda = \gamma_0$. We now choose $\phi \leq \varepsilon\gamma_0/(1+\varepsilon)$ for some constant $\varepsilon > 0$ and $\lambda \geq c((1+1/\varepsilon)/\gamma_0)^2 \ln(n)$. Putting everything together, we obtain that $p_{t,i} \geq \gamma_0(1 - \varepsilon/(1+\varepsilon)) = \gamma_0/(1+\varepsilon)$ with probability at least $1 - 2n^{-2c}$, which proves the statement **(i)**.

The statement **(ii)** is proved by using a similar line of argumentation, where Q_i now denotes the number of 1s sampled at a bit position $i \in [j]$ in the entire population of λ individuals. Finally, the statement **(iii)** follows the first statement via a union bound. ■

Lemma 20 tells us that all marginals $p_{t,1}, p_{t,2}, \ldots, p_{t,n}$ are at least $\gamma_0/(1+\varepsilon)$ in an iteration $t \in \mathbb{N}$ with probability polynomially close to one. To show an upper bound on the expected runtime for the PBIL on the LEADINGONES function, we first apply the level-based theorem to obtain an upper bound assuming that all marginals are at least $\gamma_0/(1+\varepsilon)$ during the whole run.

We are ready to establish an improved upper bound on the expected runtime of the PBIL the LEADINGONES function. Surprisingly, the proof is straightforward and not very technically demanding compared to the proof in [133, Theorem 2].

Theorem 18. *Assume that all marginals $p_{t,1}, p_{t,2}, \ldots, p_{t,n}$ are at least $\mu/(\lambda(1+\varepsilon))$ for some constant $\varepsilon \in (0,1)$ during the course of the optimisation. The* PBIL *with an offspring population size $\lambda \geq c \log n$ for a sufficiently large constant $c > 0$, a smoothing parameter*

$\rho \in (1/e, 1]$ *and a parent population size* μ *such that*

$$\frac{\mu}{\lambda} \leq \left(\frac{\rho^{2+\ln(1+\varepsilon)}}{e(1+\varepsilon)} \right)^{1/\ln(e\rho)} \tag{5.19}$$

has an expected runtime of $O\left(n\lambda \log \lambda + n^2\right)$ *on the* LEADINGONES *function.*

Proof. We use the canonical partition, defined in (5.14), in which each subset A_j contains individuals with exactly j leading 1s. There are $m = n+1$ levels ranging from A_0 to A_n. Let $\gamma_0 = \mu/\lambda = \mathrm{O}(1)$. All marginals $p_{t,1}, p_{t,2}, \ldots, p_{t,n}$ are also assumed to be at least $\gamma_0/(1+\varepsilon) = \Omega(1)$ for some constant $\varepsilon \in (0,1)$ during the course of the optimisation.

For two conditions (G1) and (G2), assuming that $|P_t \cap A_{\geq j}| \geq \gamma_0 \lambda = \mu$, we are required to show that the probability of sampling an offspring in levels $A_{\geq j+1}$ in iteration $t+1$ is lower bounded by $(1+\delta)\gamma$ for some constant $\delta > 0$. We note by Lemma 1 that this probability can be bounded from below as follows:

$$\prod_{i=1}^{j+1} p_{t+1,i} \geq \left(\prod_{i=1}^{j} q_i \right) \cdot p_{t+1,j+1},$$

for any vector $q := (q_1, \ldots, q_j)$, which majorises the vector $p^*_{t+1} := (p_{t+1,i}, \ldots, p_{t+1,j})$. In the remainder of this proof, we shall construct such a vector q from vector p^*_{t+1}.

In order to construct vector q, we will shift the weight $\sum_{i=1}^{j} p_{t+1,i}$ as far as possible to the marginals with smaller indices. The trivial upper bound on each component q_i is the upper border $1 - 1/n$. For the lower bound, we note from the assumption $|P_t \cap A_{\geq j}| \geq \mu$ that the μ fittest individuals have at least j leading 1s, meaning that when updating the model we always have $p_{t+1,i} = (1-\rho)p_{t,i} + \rho \geq \rho$ for each $i \in [j]$. Therefore, a trivial lower bound on each component q_i is the smoothing parameter ρ. We define a vector $q = (q_1, \ldots, q_j)$ as follows:

$$q_i = \begin{cases} 1 - 1/n, & \text{if } 0 \leq i \leq m, \\ \rho, & \text{if } m+2 \leq i \leq j, \\ \sum_{i=1}^{j} p_{t+1,i} - m(1-1/n) - (j-m-1)\rho, & \text{if } i = m+1, \end{cases} \tag{5.20}$$

for an integer $m = \lfloor g(j) \rfloor$, where

$$g(j) = \frac{\sum_{i=1}^{j} p_{t+1,i} - j\rho}{1 - 1/n - \rho} \geq \frac{\sum_{i=1}^{j} p_{t+1,i} - j\rho}{1 - \rho}. \tag{5.21}$$

Because of the floor function, we always get $g(j) - 1 < m \leq g(j)$, and thus $\rho \leq q_{m+1} \leq 1 - 1/n$, meaning that the defined value of the component q_{m+1} is indeed a probability. By the definition of the vector q in (5.20), we have for any $k \in [j-1]$ that

$$\sum_{i=1}^{k} q_i \geq \sum_{i=1}^{k} p_{t+1,i}$$

and

$$\sum_{i=1}^{j} q_i = \sum_{i=1}^{j} p_{t+1,i}.$$

Therefore, according to Definition 7 vector q majorises vector p^*_{t+1}. By Lemma 1, the probability of sampling the j first bits correctly is

$$\prod_{i=1}^{j} p_{t+1,i} \geq \prod_{i=1}^{j} q_i \geq \left(1 - \frac{1}{n}\right)^m \cdot \rho^{j-m} \geq \frac{\rho^{j-m}}{e}, \tag{5.22}$$

which holds because $(1-1/n)^m \geq 1/e$ for any integer $m < n$. Recall that we aim at showing that the above probability is at least a constant, so we are done if we can show that $j - m = \mathrm{O}(1)$. We are going to show that this is indeed the case.

Let $p_0 := \gamma_0/(1+\varepsilon)$. By applying Lemma 20 for the first j bits and Lemma 8, we get

$$\sum_{i=1}^{j} p_{t,i} \geq j \cdot \left(\prod_{i=1}^{j} p_{t,i}\right)^{1/j} \geq j \cdot p_0^{1/j}; \tag{5.23}$$

thus,

$$\sum_{i=1}^{j} p_{t+1,i} = (1-\rho) \sum_{i=1}^{j} p_{t,i} + j\rho \geq (1-\rho) j p_0^{1/j} + \rho j.$$

We also have

$$\begin{aligned}
j-m &< j-(g(j)-1) && \text{(since } m > g(j)-1\text{)}\\
&\leq j+1-\frac{(1-\rho)jp_0^{1/j}+\rho j-j\rho}{1-\rho} && \text{(by (5.21))}\\
&= 1+j(1-p_0^{1/j})\\
&\leq 1+j(-\ln(p_0)/j) && \text{(by Lemma 9)}\\
&= 1-\ln(p_0).
\end{aligned}$$

Since the parameter γ_0 is assumed constant, so is the value $p_0 = \gamma_0/(1+\varepsilon)$ for any constant $\varepsilon > 0$; thus,

$$j-m < 1-\ln(p_0) = \mathrm{O}(1), \tag{5.24}$$

which also means that the probability of sampling the j first bits correctly in iteration $t+1$ is at least a constant. In the remainder of the proof, we will use this result to verify conditions (G1) and (G2) of the level-based theorem.

Condition (G1)

We are interested in a lower bound z_j on the probability of sampling an offspring in levels $A_{\geq j+1}$ in iteration $t+1$. Because the marginal $p_{t+1,j+1} \geq 1/n$, this probability is

$$\begin{aligned}
\left(\prod_{i=1}^{j} p_{t+1,i}\right)\cdot p_{t+1,j+1} &\geq \frac{\rho^{j-m}}{e}\cdot\frac{1}{n} && \text{(by (5.22))}\\
&\geq \frac{\rho^{\mathrm{O}(1)}}{en} && \text{(by (5.24))}\\
&= \Omega\left(\frac{1}{n}\right).
\end{aligned}$$

Thus, condition (G1) is satisfied with $z_j = \Omega(1/n)$ and $z_* = \min_{j\in[m-1]}\{z_j\} = \Omega(1/n)$.

Condition (G2)

Assume further that $|P_t \cap A_{\geq j+1}| \geq \gamma\lambda$ for all $\gamma \in (0,\gamma_0]$, meaning that the marginal at bit position $j+1$ will be set to $p_{t+1,j+1} \geq (1-\rho)p_{t,j+1} + \rho(\gamma\lambda)/\mu \geq \rho\gamma/\gamma_0$. In this case, the

probability of sampling the $j+1$ first bits correctly is

$$\left(\prod_{i=1}^{j} p_{t+1,i}\right) \cdot p_{t+1,j+1} \geq \frac{\rho^{1-\ln p_0}}{e} \cdot \frac{\rho\gamma}{\gamma_0} \geq \frac{\rho^{2-\ln(p_0)}\gamma}{e\gamma_0} = \frac{\rho^2\gamma}{\rho^{\ln(p_0)}\gamma_0 e}. \tag{5.25}$$

For $\rho = 1$, the lower bound in (5.25) becomes $\gamma/(e\gamma_0)$, and the condition (G2) can be easily confirmed by setting $\gamma_0 \leq 1/(e(1+\varepsilon))$ for any constant $\varepsilon > 0$. We note that this is already obtained in Theorem 15 for the UMDA on the LEADINGONES function. Otherwise, if the smoothing parameter $\rho < 1$, we can rewrite

$$\rho^{\ln(p_0)} = p_0^{\ln\rho} = \frac{\gamma_0^{\ln\rho}}{(1+\varepsilon)^{\ln\rho}}$$

for any constant $\varepsilon > 0$, then (5.25) is equivalent to

$$\geq \frac{\rho^2\gamma}{\gamma_0 e} \cdot \frac{(1+\varepsilon)^{\ln\rho}}{\gamma_0^{\ln\rho}} = \frac{\rho^2\gamma(1+\varepsilon)^{\ln\rho}}{e\gamma_0^{1+\ln\rho}} \geq (1+\varepsilon)\gamma,$$

which always holds if we choose the value γ_0 such that

$$\gamma_0^{1+\ln\rho} \leq \frac{\rho^2}{e(1+\varepsilon)^{1-\ln\rho}} = \frac{\rho^{2+\ln(1+\varepsilon)}}{e(1+\varepsilon)}. \tag{5.26}$$

For any $\rho \in (0,1]$, we know that $1-\ln\rho \geq 1$, and the right-hand side of (5.26) is always less than one, so in the left-hand side we require $1+\ln\rho > 0$, which is equivalent to $\rho > 1/e$. We then obtain the bound on the selection rate as follows.

$$\gamma_0 = \frac{\mu}{\lambda} \leq \left(\frac{\rho^{2+\ln(1+\varepsilon)}}{e(1+\varepsilon)}\right)^{1/\ln(e\rho)} = \mathrm{O}(1). \tag{5.27}$$

The smoothing parameter ρ is assumed a constant (from the statement of the theorem), so is the upper bound on the selection rate γ_0. In the end, condition (G2) of Theorem 4 is verified.

Condition (G3)

To satisfy this condition, we are required to choose an offspring population size

$$\lambda \geq \left(\frac{4}{\gamma_0\delta^2}\right)\ln\left(\frac{128m}{z_*\delta^2}\right).$$

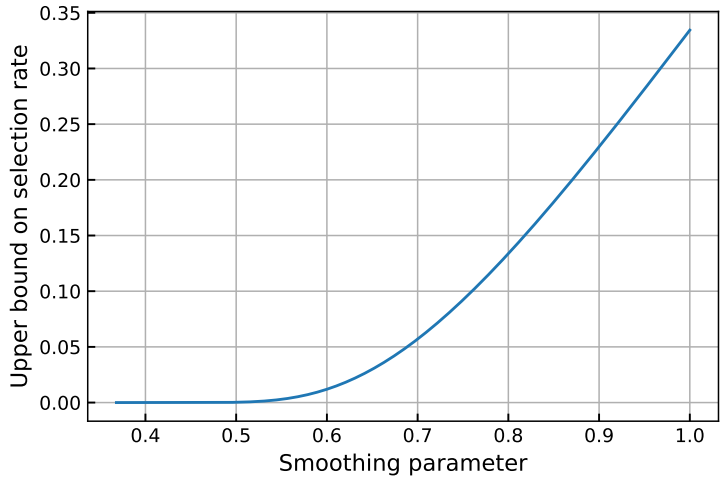

Fig. 5.1 Threshold on the selection rate for the PBIL with $\rho \in (1/e, 1]$ on the LEADINGONES function in (5.27) with $\varepsilon = 0.01$.

Since $m = \mathrm{O}(n)$, $1/z_* = \mathrm{O}(n)$, and $\gamma_0 = \mathrm{O}(1)$, the condition can be satisfied by choosing a sufficiently large constant $c > 0$ such that $\lambda \geq c \log n$.

Having verified the three conditions (G1), (G2) and (G3), and noting that $\ln\left(\delta\lambda/(4+\delta\lambda z_j)\right) < \ln(3\delta\lambda/2)$, Theorem 4 now guarantees an upper bound

$$\left(\frac{8}{\delta^2}\right)\sum_{j=0}^{n-1}\left[\lambda \ln\left(\frac{3\delta\lambda}{2}\right)+\frac{1}{z_j}\right] = \mathrm{O}\left(n\lambda\log\lambda + n^2\right),$$

which completes the proof. ■

We note from (5.27) that the threshold on the selection rate is a function of the smoothing parameter $\rho \in (1/e, 1]$, denoted by $h(\rho)$. When $\rho \to 1$, that is, the PBIL converges to the UMDA, $h(\rho) \to 1/(e(1+\varepsilon))$, which matches the selection rate considered in Theorem 15. Also, $h(\rho)$ is an increasing function and has a very small value when ρ gets closer to $1/e \approx 0.3679$ (see Fig. 5.1). In other words, we need to pick an extremely high selection rate when the smoothing parameter ρ approaches $1/e$ (from above) for the algorithm to optimise the function efficiently.

We are also interested in the runtime of the PBIL algorithm on the LEADINGONES function under a prior noise model.

Theorem 19. *Assume that all marginals $p_{t,1}, p_{t,2}, \dots, p_{t,n}$ are at least $\mu/(\lambda(1+\varepsilon))$ for some constant $\varepsilon \in (0,1)$ during the course of the optimisation. Consider the prior noise model with $O(1) \ni p \in (0,1)$. The* PBIL *with an offspring population size $\lambda \geq c \log n$ for some sufficiently large constant $c > 0$, a smoothing parameter $\rho \in (1/e, 1]$, and also a parent population size such that*

$$\frac{\mu}{\lambda} \leq \left(\frac{\rho^{2+\frac{\rho\varepsilon}{1-\rho}+\ln((1-p)(1+\varepsilon)^2)}}{e(1+\varepsilon)} \right)^{1/\ln(e\rho)} \tag{5.28}$$

has an expected runtime of $O\left(n\lambda \log \lambda + n^2\right)$ on the LEADINGONES *function.*

Proof. We define the levels as in (5.14), and there are $m = n+1$ levels. For any constant $\delta \in (0,1)$, we define $\gamma_0 = \mu/(\lambda(1-\delta)(1-p))$. Note that $\gamma_0 = \mathrm{O}(1)$ since $\mu/\lambda = \mathrm{O}(1)$ and $p = \mathrm{O}(1)$. We also partition the noisy population into four groups as in the proof of Theorem 17 and pessimistically assume that the PBIL uses all of the B individuals in group 3 and $\mu - B$ individuals are chosen from group 1 when updating the model. For all $i \in [j]$, let Q_i be the number of individuals in group 3 which has 1s at bit positions 1 through j, except for one position i where it has a 0. By definition, we then have

$$\sum_{i=1}^{j} Q_i = B. \tag{5.29}$$

Condition (G1)

Similar to the proof of Theorem 18, we shall show that the probability of sampling the j first bits correctly is at least a constant using a majorisation argument. Because noise only has an impact on the weight $\sum_{i=1}^{j} p_{t+1,i}$, we still define the vector q as in (5.20) and $m = \lfloor g(j) \rfloor$ as in (5.21). We are left to show a constant upper bound on the difference $j-m$ used in (5.22). We notice that in this case the weight becomes

$$\begin{aligned}
\sum_{i=1}^{j} p_{t+1,i} &= (1-\rho)\sum_{i=1}^{j} p_{t,i} + \frac{\rho}{\mu}\sum_{i=1}^{j} X_{i,t} \\
&\geq (1-\rho)(j p_0^{1/j}) + \frac{\rho}{\mu}\sum_{i=1}^{j} X_{i,t}, \qquad \text{(by (5.23))}
\end{aligned}$$

which by noting that $\sum_{i=1}^{j} X_{i,t} = j\mu - \sum_{i=1}^{j} Q_i = j\mu - B$ satisfies

$$\geq (1-\rho) j p_0^{1/j} + \frac{\rho}{\mu}(j\mu - B) = (1-\rho) j p_0^{1/j} + \rho j - \rho \frac{B}{\mu}. \tag{5.30}$$

Putting everything together, we then obtain

$$\begin{aligned}
j - m &< j + 1 - \frac{(1-\rho) j p_0^{1/j} - \rho B/\mu}{1-\rho} && \text{(by (5.21) \& (5.30))} \\
&= 1 + \frac{\rho B}{(1-\rho)\mu} + j - j p_0^{1/j} \\
&= 1 + \frac{\rho B}{(1-\rho)\mu} + j(1 - p_0^{1/j}) \\
&\leq 1 + \frac{\rho B}{(1-\rho)\mu} + j(-\ln(p_0)/j) && \text{(by Lemma 9)} \\
&= 1 + \frac{\rho B}{(1-\rho)\mu} - \ln(p_0),
\end{aligned}$$

which by (5.16), i.e., that $B \leq \varepsilon\mu$ for some small constant $\varepsilon > 0$, with probability $1 - 2^{-\Omega(\mu)}$ satisfies

$$\leq 1 + \frac{\rho\varepsilon\mu}{(1-\rho)\mu} - \ln(p_0) = 1 + \frac{\rho\varepsilon}{1-\rho} - \ln(p_0) = \mathrm{O}(1). \tag{5.31}$$

Thus, the probability of sampling an offspring in levels $A_{\geq j}$ is at least a constant, which immediately results in a lower bound of $\Omega(1/n)$ on the probability of sampling the $j+1$ first bits correctly. Thus, the condition (G1) of the level-based theorem is satisfied with the parameter $z_j = \Omega(1/n)$ and $z_* = \min_{j \in [m-1]\{z_j\}} = \Omega(1/n)$.

Condition (G2)

According to line 10 in Algorithm 4 and by the assumption that $|P_t \cap A_{\geq j+1}| \geq \gamma\lambda$, we know that $p_{t+1,j+1} \geq (1-\rho) p_{t,j+1} + \rho(\gamma\lambda)/\mu \geq \rho\gamma/\gamma_0 = \rho\gamma\lambda/\mu$ for all $\gamma \in (0, \gamma_0]$. Then,

the probability of sampling an offspring in levels $A_{\geq j+1}$ is

$$\begin{aligned}\left(\prod_{i=1}^{j} p_{t+1,i}\right) \cdot p_{t+1,j+1} &\geq \frac{\gamma}{e\mu/\lambda} \cdot \rho^{2+\frac{\rho\varepsilon}{1-\rho}-\ln(p_0)} && \text{(by (5.25) \& (5.31))}\\ &\geq \frac{\gamma\rho^{2+\frac{\rho\varepsilon}{1-\rho}}}{e\mu/\lambda} \cdot \frac{(1+\varepsilon)^{\ln\rho}}{\gamma_0^{\ln\rho}}\\ &\geq \frac{\gamma\rho^{2+\frac{\rho\varepsilon}{1-\rho}}}{e} \cdot \frac{(1-p)^{\ln\rho}(1+\varepsilon)^{2\ln\rho}}{(\mu/\lambda)^{1+\ln\rho}}\\ &\geq (1+\varepsilon)\gamma,\end{aligned}$$

which always holds if we choose the selection rate μ/λ such that

$$\frac{\mu}{\lambda} \leq \left(\frac{\rho^{2+\frac{\rho\varepsilon}{1-\rho}}(1-p)^{\ln\rho}}{e(1+\varepsilon)^{1-2\ln\rho}}\right)^{1/(1+\ln\rho)} = \left(\frac{\rho^{2+\frac{\rho e}{1-\rho}+\ln((1-p)(1+\varepsilon)^2)}}{e(1+\varepsilon)}\right)^{1/\ln(e\rho)}.$$

Similarly to (5.26), we also require $\rho \in (1/e, 1]$. The condition (G2) is now verified.

Condition (G3)

Since $\gamma_0 = \mathrm{O}(1), m = \mathrm{O}(n), 1/z_* = \mathrm{O}(n)$, to satisfy this condition, it suffices to have an offspring population size of $\lambda \geq c \log n$ for some sufficiently large constant $c > 0$.

Having verified the three conditions (G1), (G2) and (G3), Theorem 4 now guarantees an upper bound of $\mathrm{O}\left(n\lambda \log \lambda + n^2\right)$, assuming that all marginals are always lower bounded by the constant $(\mu/\lambda)/(1+\varepsilon)$ during the whole run. Note that we further assume the occurrence of the following two events in each iteration $t \in \mathbb{N}$:

(i) the number of individuals in group 1 (with actual fitness $f(x) \geq j$ and noisy fitness $F(x) \geq j$) is at least μ with probability $1 - 2^{-\Omega(\mu)}$ (see Lemma 19),

(ii) the number of individuals in group 3 is $B \leq \varepsilon\mu$ for some small constant $\varepsilon > 0$ with probability $1 - 2^{-\Omega(\mu)}$.

By the union bound, either or both of these events happen in an iteration $t \in \mathbb{N}$ with probability at most $2^{-\Omega(\mu)} + 2^{-\Omega(\mu)} = 2^{-\Omega(\mu)}$. The complementary event occurs with probability at least $1 - 2^{-\Omega(\mu)}$. We also refer to them as the unlikely and the likely events, respectively. In the latter case, we have shown above an expected runtime of $\mathrm{O}\left(n\lambda \log \lambda + n^2\right)$. For the

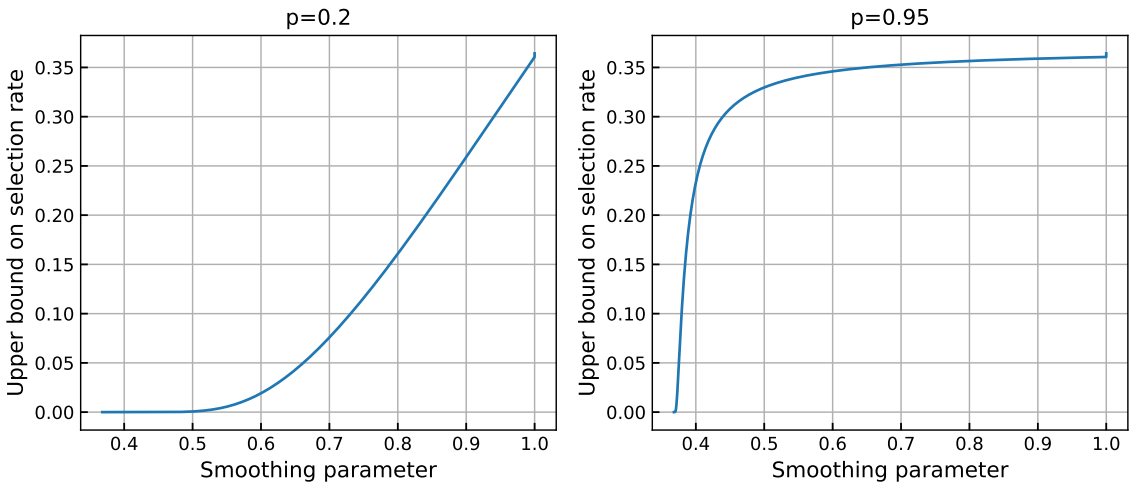

Fig. 5.2 Threshold on the selection rate for the PBIL with $\rho \in (1/e, 1]$ on the noisy LEADINGONES function in (5.28) with $\varepsilon = 0.01$ for two noise probabilities $p = 0.2$ (left) and $p = 0.95$ (right).

unlikely event, by following the same argument as in Theorem 17 we then obtain an expected runtime of $\mathrm{O}\left(n\lambda \log \lambda + n^3\right)$. By the law of total expectation, the expected runtime is

$$(1 - 2^{-\Omega(\mu)}) \cdot \mathrm{O}\left(n\lambda \log \lambda + n^2\right) + 2^{-\Omega(\mu)} \cdot \mathrm{O}\left(n\lambda \log \lambda + n^3\right) = \mathrm{O}\left(n\lambda \log \lambda + n^2\right)$$

for a sufficiently large constant $c > 0$ in $\lambda \geq c \log n$, assuming that all marginals are always lower bounded by the threshold $(\mu/\lambda)/(1+\varepsilon) = \Omega(1)$ during the whole run. ■

Fig. 5.2 plots the threshold on the selection rate in (5.28) for two noise probabilities $p = 0.2$ and $p = 0.95$. Surprisingly, when the noise probability p increases, the PBIL with a smoothing parameter ρ close to the threshold $1/e$ and a low selection rate μ/λ can still take an $\mathrm{O}\left(n^2\right)$ expected runtime to optimise the noisy LEADINGONES function. We also observe that in the context of LEADINGONES the smoothing parameter ρ should be set to some value close to one.

As a final remark, the results derived in Theorem 17 and Theorem 19 are limited in the sense that they are not the overall expected runtimes but the expected runtimes on a sub-probability space due to the assumption that all marginals are lower bounded in each iteration during the optimisation by the constant $(\mu/\lambda)/(1+\varepsilon)$ for some constant $\varepsilon \in (0,1)$. Though we know by Lemma 20 that this requirement is fulfilled in each iteration with probability at least $1 - 2n^{-2c+1}$. By the union bound, this happens during the first $\mathrm{O}\left(n \log \lambda + n^2/\lambda\right)$ iterations with probability at least $1 - 2n^{-2c+1} \cdot \mathrm{O}\left(n \log \lambda + n^2/\lambda\right) = 1 - \mathrm{O}(n^{-c'})$ for some

other arbitrarily large constant $c' > 0$ as long as the constant $c > 0$ (in $\lambda \geq c \log n$) is chosen sufficiently large. We call the associated event and its complementary the likely and the unlikely events, respectively. Although the constant $c > 0$ can be chosen arbitrarily large such that the likely event occurs with probability arbitrarily polynomially close to one, the overall expected runtime is still difficult to derive. This is because we do not know the algorithms' expected runtime on the function corresponding to the unlikely event.

To make this point more rigorous, we call an iteration a success if none of the marginals in that iteration drops below the constant threshold; otherwise, the iteration is considered a failure. We then let T_f denote the first hitting of a failure. Recall from Section 3.3.1 that $\tilde{T} = T/\lambda$ denotes the number of iterations until the global optimum has been found for the first time. By the law of total expectation, we have

$$\mathbb{E}[T] = \mathbb{E}\left[T \cdot \mathbb{1}_{\{\tilde{T} < T_f\}}\right] + \mathbb{E}\left[T \cdot \mathbb{1}_{\{\tilde{T} \geq T_f\}}\right].$$

We have shown above that $\mathbb{E}\left[T \cdot \mathbb{1}_{\{\tilde{T} < T_f\}}\right] = \mathrm{O}\left(n\lambda \log \lambda + n^2\right)$ and are missing an upper bound on the expectation

$$\mathbb{E}\left[T \cdot \mathbb{1}_{\{\tilde{T} \geq T_f\}}\right] = \mathbb{E}\left[T \mid \tilde{T} \geq T_f\right] \cdot \Pr\left(\tilde{T} \geq T_f\right)$$

Here, an iteration is a failure with probability at most $2n^{-2c+1}$; therefore, the hitting time T_f follows a geometric distribution with a success probability at most $2n^{-2c+1}$, and the expectation is $\mathbb{E}\left[T_f\right] \geq 1/(2n^{-2c+1}) = n^{2c-1}/2$, which will easily dominate $\mathrm{O}\left(n\lambda \log \lambda + n^2\right)$ for c large enough. Although we were not able to obtain the overall expected runtimes for the PBIL on the LeadingOnes function and its noisy variant at the time of submission, we consider the extension of these preliminary results as a potential future work since there had been few runtime results for the PBIL on test functions in the literature.

5.6 Conclusion

In this chapter, we have derived several runtime results for the UMDA and the PBIL on the LeadingOnes function. For the UMDA, we have found that the algorithm under a low selection rate requires an exponential runtime. More specifically, the algorithm takes an expected runtime of $2^{\Omega(\mu^k)}$ when $\mu = o(n^{1/k})$ for some constant $k \in [0,1)$ and

$\mu/\lambda \geq (1+\delta)/e^{1-\varepsilon}$ for any constant $\delta \in (0,1)$ and $\varepsilon \in (0,1)$. The analyses revealed the limitations of the probabilistic model based on probability vectors as the algorithm hardly stays at promising states for long enough to make progress. This led the algorithm into a non-optimal equilibrium state from which the sampling of the global optimum was exponentially unlikely. On the other hand, when the selection rate is high we obtain a lower bound of $\Omega(n\lambda/\log\lambda)$ on the expected runtime for the algorithm. We then moved on to consider the PBIL on the LEADINGONES function. The algorithm is shown to optimise the function within an expected runtime of $\mathrm{O}\left(n^2\right)$ for the optimal parameter settings, assuming that all marginals are always bounded from below by some constant.

Furthermore, we for the first time study the performances of the UMDA and the PBIL on the LEADINGONES function under a prior noise model, where a uniformly chosen bit is flipped with a constant probability $p \in (0,1)$ before invoking the fitness function. We show that an $\mathrm{O}\left(n^2\right)$ expected runtime still holds in this case for both algorithms under an offspring population size $\lambda = \mathrm{O}(n/\log n)$. Despite the simplicity of the noise model, this can be viewed as the first step towards broadening our understanding of the two algorithms in a noisy environment.

Chapter 6

Mild deception is enough to fool UMDA with moderate population sizes

This chapter is based on the joint work [78] with Per Kristian Lehre. We extend the exponential lower bound on the expected runtime of the UMDA on the DLB function in [78, Theorem 4.9] for a broader range of population sizes and close the gap left on the selection rate μ/λ in the conference version.

6.1 Open problems

The fact that the UMDA assumes a pairwise independence between decision variables leads some to conjecture [57] that the algorithm will not perform well when epistasis and deception are present in the environment. However, results from the preceding chapter have shown that univariate EDAs could still cope very well with epistasis. Regarding deception, Hauschild & Pelikan [57] presented an example in which the UMDA was shown empirically to have difficulty optimising the so-called TRAP-5 function. For evaluation, the function first partitions the bitstring of length $n \in \mathbb{N}$ into consecutive blocks of length five. Afterwards, the fitness of each block is obtained by applying the well-known TRAP function, which is then summed over all blocks to yield the overall fitness of the individual [57]. Here, we assume that the problem instance size n is a multiple of five. Recall the definition of the

TRAP function [57] as follows.

$$\textsc{Trap}(x) = \begin{cases} n & \text{if } x = 1^n, \\ \textsc{ZeroMax}(x) - 1 & \text{otherwise,} \end{cases}$$

where $\textsc{ZeroMax}(x) = n - \textsc{OneMax}(x)$ counts the number of 0s in the bitstring [108]. In each block, the algorithm climbs up the ZEROMAX slope and consequently gets further and further away from the all-ones bitstring, i.e., the global optimum [57]. In other words, marginals in each block are deceived into reaching the extreme value of zero and as a result the global optimum cannot be sampled. Fig. 6.1 illustrates how the marginals change over iterations (or generations) in one run of the UMDA without borders with $\lambda = 100$ and $\mu = 50$ on two problems, ONEMAX and TRAP-5, with the same problem instance size $n = 50$ [57, Fig. 2 & Fig. 4].

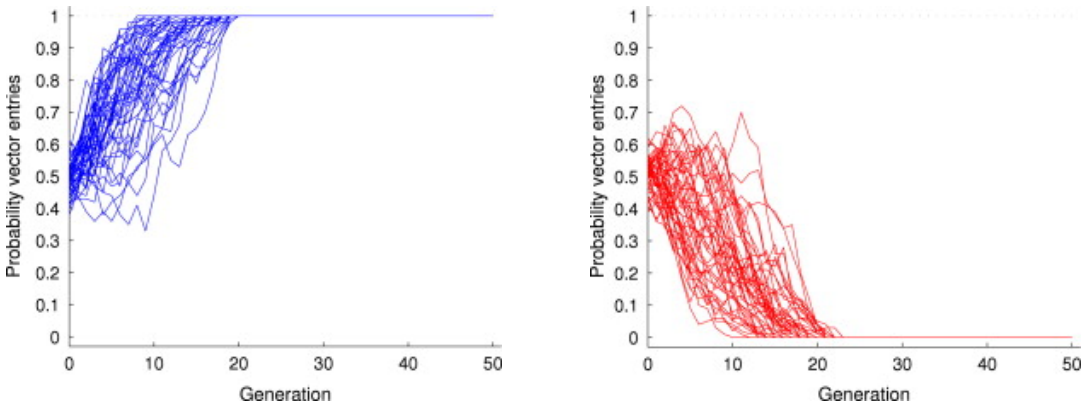

Fig. 6.1 Marginals in the probabilistic model in one run of the UMDA (without margins) with $\lambda = 100$ and $\mu = 50$. Left: ONEMAX with $n = 50$. Right: TRAP-5 with $n = 50$.

This excellent example demonstrates the limitations of the univariate model as statistics of low order (i.e., the mean value) is misleading [57]. However, the TRAP-5 function might be as difficult for the UMDA as it is for EA*s* [52]. We believe not only will the UMDA fail on this function, but it will also fail on some other problem with a milder degree of deception. To this end, a function where the UMDA takes an exponential expected runtime, while simple EAs have a polynomial expected runtime is still missing. On the other hand, a function where the UMDA outperforms simple EAs does exist in [14], where the SUBSTRING function was proposed to illustrate the advantage of the probability vector-based model. For this particular function, the $(1+1)$ EA takes a runtime of $2^{\Omega(n)}$ with probability $1 - 2^{-\Omega(n)}$, whereas the UMDA with an offspring population size $\lambda = \Omega(n^{2+\varepsilon})$, for any constant $\varepsilon > 0$, and a parent

population size $\mu = \lambda/2$ optimises the function in polynomial runtime with probability $1 - 2^{-\Omega(n)}$. We note that this result is limited in many senses that the population size is large, the selection rate μ/λ is fixed to $1/2$ and the considered UMDA does not have borders.

Motivated by this, we introduce a new benchmark problem named Deceiving Leading Blocks (DLB, see Definition 4), defined over the search space $\mathscr{X} = \{0,1\}^n$, which has an epistasis level of $n-1$ and is mildly deceptive. The fitness depends on the number of leading 11s (similar to the function LOB_2 [64, Definition 13]), and reaching a unique global optimum requires overcoming many mild traps. Generally speaking, this function is more difficult to optimise than the LEADINGONES function is. However, it is still much easier compared to the TRAP-5 function. The global optimum is the all-ones bitstring. The problem's difficulty is determined by the width w of each block that can be altered to increase the deception level. Here, we consider the smallest width $w = 2$ and assume that n is a multiple of two, as this suffices to prove an exponential gap between the runtimes of the UMDA and some simple EAs.

By studying this problem, we first show that simple EAs, including elitist and non-elitist variants such as $(1+\lambda)$ EA, $(\mu+1)$ EA and (μ,λ) EA, can optimise the DLB function within an expected runtime of $\mathrm{O}\left(n^3\right)$. We then show that the UMDA with a parent population size $\mu = o(n/\log n)$ and an offspring population size $\mu \leq \lambda = \mathrm{O}\left(\mu^{2-k}\right)$ for any constant $k \in (0,1]$ requires an expected runtime of $2^{\Omega(\mu^k)}$ to optimise the DLB function. Note that without deception (i.e., the LEADINGONES function) the exponential runtime holds for $\lambda \leq \frac{e^{1-\varepsilon}}{1+\delta}\mu < 2.72\mu$ only (see Theorem 14), which is asymptotically significantly smaller than the bound $\mathrm{O}\left(\mu^{2-k}\right)$. By introducing a very mild degree of deception into the fitness function, the UMDA requires an exponential expected runtime for a broad range of population sizes. In essence, our analysis here reveals the challenge faced by the class of univariate EDAs when dealing with deception. In this case, the UMDA is said to be fooled by deception. On the other hand, for much larger population sizes $\lambda = \Omega(\mu^2)$ we obtain an upper bound of $\mathrm{O}\left(n^3 + n\lambda\log\lambda\right)$ on the expected runtime of the UMDA on the DLB function. In this case, the selection rate is $\mu/\lambda = \mathrm{O}\left(1/\mu\right)$, and the UMDA selects few fittest individuals to update the probabilistic model. Intuitively speaking, we believe that the UMDA degenerates into the $(1,\lambda)$ EA, where only the fittest offspring is selected for the next iteration.

Very recently, Doerr & Krejca [33] investigated the runtime behaviour of the UMDA with large population sizes $\mu = \Omega(n\log n)$ and $\lambda = \Omega(\mu)$ on the DLB function. They

showed that the algorithm optimises the function using at most $O(n\lambda)$ fitness evaluations in expectation. When $\mu = \Omega(n \log n)$, the so-called genetic drift was shown not to happen with high probability, and thus all marginals were well bounded by some constant lower bound. The characteristic of the UMDA under this regime had already been shown in [134]. Here, we are more interested in the smaller population regime because for population sizes $\mu = \Omega(n \log n)$ we get $O(n\lambda) = O\left(n^2 \log n\right)$, which is by a logarithmic factor larger than the best-known expected runtime of the UMDA on the LEADINGONES function. Furthermore, the DLB instance that was studied here and in [134] (extended journal version [37]) is the simplest with a minimum width of $w = 2$. For this instance, the UMDA requires an exponential expected runtime for the setting $\lambda = O\left(\mu^2\right)$. We conjecture that the algorithm would have difficulty locating the global optimum for other instances of large widths $w > 2$ even under much larger population sizes, and consequently, the limitation of univariate EDAs to deception might be substantial as well as inherent.

6.2 EAs handle mild deception efficiently

We start by showing that simple EAs optimise the DLB function in polynomial expected runtime. We consider both elitist and non-elitist EAs, namely the $(1+\lambda)$ EA [63, Algorithm 1], the $(\mu+1)$ EA [126, Definition 1] and the (μ,λ) EA [80]. Although these EAs are simple, we analyse them here to emphasise their efficiency in dealing with epistasis and mild deception. Recall the Definition 6 for canonical fitness-based partitions. We will apply the level-based theorem and the so-called fitness-level method [122] as follows.

Theorem 20. *Consider a partition of the search space into non-empty sets $A_1, \ldots, A_m$ such that $A_1 <_f A_2 <_f \ldots <_f A_m$ and A_m only contains global optima. For a mutation-based* EA *$\mathscr{A}$ we say that $\mathscr{A}$ is in A_i or on level i if the best individual created so far is in A_i. Consider some elitist* EA *$\mathscr{A}$ and let s_i be a lower bound on the probability of creating a new offspring in $A_{i+1} \cup A_{i+2} \cup \ldots \cup A_m$, provided $\mathscr{A}$ is in A_i. Then the expected runtime of $\mathscr{A}$ on f (without the cost of initialisation) is bounded by*

$$\sum_{i=1}^{m-1} \Pr\left(\mathscr{A} \text{ starts in } A_i\right) \sum_{j=1}^{m-1} \frac{1}{s_i} \leq \sum_{i=1}^{m-1} \frac{1}{s_i}.$$

For the DLB function, the search space $\mathscr{X}$ is first partitioned into non-empty disjoint subsets $A_0, A_1, \ldots, A_m$ (called levels, where $m := n/2$) such that

$$A_i = \{x \in \mathscr{X} : \phi(x) = i\}, \tag{6.1}$$

where $\phi(x)$ is defined in (3.1), and A_m contains the all-ones bitstring. We now give runtime bounds on the DLB function for the EAs; the proofs are straightforward.

Theorem 21. *The expected runtime of the* $(1+\lambda)$ EA *on the* DLB *function is* $O\left(\lambda n + n^3\right)$.

Proof. Levels are defined as in (6.1). The probability of leaving the current level of $i < m$ is lower bounded by $(1-1/n)^{n-2}(1/n)^2 \geq 1/en^2$, and thus not leaving it happens with probability at most $1-1/en^2$. In each iteration, the $(1+\lambda)$ EA samples λ individuals by mutating the current bitstring. At least one among λ individuals leave the current level with probability of at least $1-(1-1/en^2)^\lambda \geq 1-e^{-\lambda/en^2}$. Note that if $\lambda \geq en^2$, then this probability is at least $1-1/e$; otherwise, it is at least $\lambda/2en^2$. Putting everything together, the expected runtime guaranteed by the fitness-level method is

$$\lambda \sum_{i=0}^{n/2-1} \left(\mathrm{O}(1) + \frac{2en^2}{\lambda}\right) = \mathrm{O}\left(n\lambda + n^3\right). \qquad \blacksquare$$

Theorem 22. *The expected runtime of the* $(\mu+1)$ EA *on the* DLB *function is* $O\left(\mu n \log n + n^3\right)$.

Proof. Levels are defined as in (6.1). It suffices to correct block $i+1$ to leave the current level. Following [126], we define a fraction $\nu(i) := n/\log n$. Given j copies of the best individual, another one is created with probability $(j/\mu)(1-1/n)^n \geq j/2e\mu$. Thus, the expected time for a fraction $\nu(i)$ of the population to be in level i is given by

$$T_0 \leq 2e\mu \sum_{j=1}^{n/\log n} (1/j) \leq 2e\mu \log n.$$

Now given $\nu(i)$ individuals in level i, the event of leaving this level occurs with probability $s_i \geq (\nu(i)/\mu)(1-1/n)^{2i}(1/n)^2 \geq (\nu(i)/\mu)1/en^2$. This probability is at least

$$s_i = \begin{cases} 1/en^2, & \text{if } \mu \leq \nu(i) \\ 1/(e\mu \log n), & \text{if } \mu > \nu(i). \end{cases}$$

The expected runtime of the algorithm on DLB is

$$\sum_{i=0}^{n/2-1}\left(T_0+\frac{1}{s_i}\right)=\mathrm{O}\left(\mu n\log n+n^3\right). \qquad \blacksquare$$

Theorem 23. *The expected runtime of the* (μ,λ) EA *with* $\lambda \geq c\log n$ *for some sufficiently large constant* $c>0$ *and* $\mu \leq \lambda e^{-2\nu}/(1+\delta)$ *for any constant* $\delta>0$ *and a mutation rate* ν/n *for some constant* $\nu \in (0,2]$ *on the* DLB *function is* $O\left(n\lambda\log\lambda+n^3\right)$.

Proof. Since (μ,λ) EA is non-elitist, Theorem 4 guarantees an upper bound on the expected runtime as long as the three conditions (G1), (G2) and (G3) are verified. Choose $\gamma_0 := \mu/\lambda$. The levels are defined as in (6.1). Moreover, A_j is assumed to be the current level.

Condition (G1)

We require a lower bound on the probability of sampling an offspring in $A_{\geq j+1}$, where $A_{\geq j+1} := \cup_{k=j+1}^{m} A_k$, given $|P_t \cap A_{\geq j}| \geq \gamma_0\lambda = \mu$. During the selection step, if we choose an individual in $A_{\geq j}$, then the step is successful if the mutation operator correctly flips two of the bits in the active block while keeping others unchanged. Thus, the probability of successful sampling is at least $(1-\nu/n)^{n-2}(\nu/n)^2$. By [30, Corollary 1.4.4.], we have $1-\nu/n \geq e^{(-\nu/n)/(1-\nu/n)} = e^{-\nu/(n-\nu)}$; therefore, $(1-\nu/n)^{n-2} \geq e^{-\nu(n-2)/(n-\nu)} \geq e^{-\nu}$ as $\nu \in (0,2]$ and $(n-2)/(n-\nu) \leq (n-2)/(n-2) = 1$.

Condition (G2)

Assume that at least $1 \leq \gamma\lambda < \mu$ individuals have at least $j+1$ leading 11s. It suffices to pick one of the $\gamma\lambda$ fittest individuals and flip none of the bits; the probability is at least

$$(\gamma\lambda/\mu)(1-\nu/n)^n \geq (\gamma/\gamma_0)e^{-2\nu} \geq (1+\delta)\gamma$$

if $\gamma_0 \leq e^{-2\nu}/(1+\delta)$ for any constant $\delta>0$.

Condition (G3)

To satisfy this condition, it suffices to choose an offspring population size $\lambda \geq c\log(n)$ for a sufficiently large constant $c>0$.

Having verified three conditions, the expected runtime of the (μ,λ) EA on DLB is

$$\mathrm{O}\left(\sum_{i=0}^{n/2-1}\left(\lambda\log\lambda+n^2\right)\right)=\mathrm{O}\left(n\lambda\log\lambda+n^3\right). \qquad \blacksquare$$

6.3 UMDA suffers with mild deception

Before we get to analysing the UMDA on the DLB function, we introduce some notation. Due to the similarity between the DLB and the LEADINGONES functions, we shall follow an approach similar to that in Chapter 5. Recall from Definition 4 that there are $m := n/2$ blocks, where n is an even positive integer. We then let $C_{t,i}$ for each $i \in [m]$ denote the number of individuals having at least i leading 11s in iteration $t \in \mathbb{N}$, and $D_{t,i}$ denote the number of individuals having $i-1$ leading 11s, followed by a 00 at the i-th block. For the special case $i = 1$, $D_{t,1}$ consists of those with the first block being a 00. We also let $E_{t,i}$ denote the number of individuals having $i-1$ leading 11s, followed by a 10 at the i-th block, and again $E_{t,1}$ consists of those having the first block being a 10.

Once the population has been sampled, the algorithm invokes truncation selection to select the μ fittest individuals to update the probability vector. We take this μ-cutoff into account by defining a random variable

$$Z_t := \max\{i \in \{0\} \cup [m] : C_{t,i} \geq \mu\}, \tag{6.2}$$

which tells us how many consecutive marginals, counting from position one, are set to the upper border $1-1/n$ in iteration t. We also define another random variable

$$Z_t^* := \max\{i \in \{0\} \cup [m] : C_{t,i} > 0\} \tag{6.3}$$

to be the number of leading 11s of the fittest individual(s). For readability, we often leave out the indices of random variables like when we write C_t instead of $C_{t,i}$, if values of the indices are clear from the context.

Similarly to Chapter 5, we also apply the principle of deferred decisions [95, p. 9] and imagine that the algorithm first samples the values of the first block for λ individuals. Once this is finished, it moves on to the second block and is repeated until the whole population is obtained.

We note that the selection mechanism prefers individuals with the first block being a 11 to those with a 00, which in turn is more preferred to those with a 10 or 01 (due to deceptive fitness). The number of 11s in the first block follows a binomial distribution with parameters λ and $p_{t,1}p_{t,2}$, that is, $C_{t,1} \sim \text{Bin}(\lambda, p_{t,1}p_{t,2})$. Having sampled $C_{t,1}$ 11s, there are $\lambda - C_{t,1}$ other individuals in block 1 in the current population. $D_{t,1}$ is also binomially distributed with parameters $\lambda - C_{t,1}$ and $(1-p_{t,1})(1-p_{t,2})/(1-p_{t,1}p_{t,2})$ by the definition of conditional probability since the event of sampling a 11 is excluded. Similarly having sampled 11s and 00s, $E_{t,1}$ is binomially distributed with $\lambda - C_{t,1} - D_{t,1}$ trials and success probability $(p_{t,1}(1-p_{t,2}))/(1-p_{t,1}p_{t,2}-(1-p_{t,1})(1-p_{t,2}))$ since again the event of sampling either a 11 or a 00 is excluded. Finally, the number of 01s is $\lambda - C_{t,1} - D_{t,1} - E_{t,1}$.

Having sampled the first block for λ individuals, and note that the bias due to selection in the second block comes into play only if the first block is a 11. Among the $C_{t,1}$ fittest individuals, those with a 11 in the second block will be ranked first, followed by those with a 00, and finally with a 10 or 01. Conditioned on the first block being a 11, the number of 11s in the second block is binomially distributed with parameters $C_{t,1}$ and $p_{t,3}p_{t,4}$, i.e., $C_{t,2} \sim \text{Bin}(C_{t,1}, p_{t,3}p_{t,4})$, and the number of 00s also follows a binomial distribution with $C_{t,1} - C_{t,2}$ trials and success probability $(1-p_{t,3})(1-p_{t,4})/(1-p_{t,3}p_{t,4})$. Similarly, $E_{t,2}$ is binomially distributed with parameters $C_{t,1} - C_{t,2} - D_{t,2}$ and $p_{t,3}(1-p_{t,4})/(1-p_{t,3}p_{t,4} - (1-p_{t,3})(1-p_{t,4}))$, and finally the number of 01s equals $C_{t,1} - C_{t,2} - D_{t,2} - E_{t,2}$. Unlike the first block, we also have $\lambda - C_{t,1}$ remaining individuals, and since there is no bias in the second block among these individuals, the numbers of 1s sampled at the two bit positions are binomially distributed with $\lambda - C_{t,1}$ trials and success probabilities $p_{t,3}$ and $p_{t,4}$, respectively.

We now consider an arbitrary block $i \in [m]$. By induction, we observe that the number of individuals having at least i leading 11s follows a binomial distribution with parameters $C_{t,i-1}$ and $p_{t,2i-1}p_{t,2i}$, that is,

$$C_{t,i} \sim \text{Bin}(C_{t,i-1}, p_{t,2i-1}p_{t,2i}). \tag{6.4}$$

Similarly,

$$D_{t,i} \sim \text{Bin}\left(C_{t,i-1} - C_{t,i}, \frac{(1-p_{t,2i-1})(1-p_{t,2i})}{1-p_{t,2i-1}p_{t,2i}}\right), \tag{6.5}$$

and

$$E_{i,t} \sim \text{Bin}\left(C_{t,i-1} - C_{t,i} - D_{t,i}, \frac{p_{t,2i-1}(1-p_{t,2i})}{p_{t,2i-1} + p_{t,2i} - 2p_{t,2i-1}p_{t,2i}}\right). \tag{6.6}$$

Finally, the number of individuals with $i-1$ leading 11s followed by a 01 in the block i is $C_{t,i-1} - C_{t,i} - D_{t,i} - E_{t,i}$. For the $\lambda - C_{t,i-1}$ remaining individuals, the numbers of 1s sampled in the bit positions $2i-1$ and $2i$ follow the binomial distributions $\mathrm{Bin}(\lambda - C_{t,i-1}, p_{t,2i-1})$ and $\mathrm{Bin}(\lambda - C_{t,i-1}, p_{t,2i})$, respectively. We note in particular that by the end of this alternative view on the sampling process, we obtain the population of λ individuals sorted in descending order according to fitness, where ties are broken uniformly at random. Let $(\mathscr{F}_t : t \in \mathbb{N})$ be a filtration induced from the population $(P_t : t \in \mathbb{N})$. The following lemma provides the expectations of these random variables.

Lemma 21. *For all $t \in \mathbb{N}$, if $i = 1$ then*

$$\begin{aligned}
\mathbb{E}[C_{t,i} \mid \mathscr{F}_{t-1}] &= \lambda p_{t,2i-1} p_{t,2i}, \\
\mathbb{E}[D_{t,i} \mid \mathscr{F}_{t-1}] &= \lambda (1 - p_{t,2i-1})(1 - p_{t,2i}), \\
\mathbb{E}[E_{t,i} \mid \mathscr{F}_{t-1}] &= \lambda p_{t,2i-1}(1 - p_{t,2i}).
\end{aligned}$$

Otherwise, if $i \in [m] \setminus \{1\}$, then

$$\mathbb{E}[C_{t,i} \mid \mathscr{F}_{t-1}] = \mathbb{E}[C_{t,i-1} \mid \mathscr{F}_{t-1}] p_{t,2i-1} p_{t,2i}, \tag{6.7}$$

$$\mathbb{E}[D_{t,i} \mid \mathscr{F}_{t-1}] = \mathbb{E}[C_{t,i-1} \mid \mathscr{F}_{t-1}](1 - p_{t,2i-1})(1 - p_{t,2i}), \tag{6.8}$$

$$\mathbb{E}[E_{t,i} \mid \mathscr{F}_{t-1}] = \mathbb{E}[C_{t,i-1} \mid \mathscr{F}_{t-1}] p_{t,2i-1}(1 - p_{t,2i}). \tag{6.9}$$

Proof. For the special case $i = 1$, the expectations are trivial since the random variables $C_{t,i}, D_{t,i}$ and $E_{t,i}$ are binomially distributed as in (6.4), (6.5) and (6.6) all with λ trials. In the remainder of the proof, we consider the case $i \neq 1$ and get

$$\begin{aligned}
\mathbb{E}[C_{t,i} \mid \mathscr{F}_{t-1}] &= \mathbb{E}[\mathbb{E}[C_{t,i} \mid C_{t,i-1}, \mathscr{F}_{t-1}] \mid \mathscr{F}_{t-1}] && \text{(by the tower rule)} \\
&= \mathbb{E}[C_{t,i-1} p_{t,2i-1} p_{t,2i} \mid \mathscr{F}_{t-1}] && \text{(by (6.4))} \\
&= \mathbb{E}[C_{t,i-1} \mid \mathscr{F}_{t-1}] p_{t,2i-1} p_{t,2i}. && (p_t \text{ is } \mathscr{F}_{t-1}\text{-measurable})
\end{aligned}$$

We also get

$$\begin{aligned}
\mathbb{E}[D_{t,i} \mid \mathscr{F}_{t-1}] &= \mathbb{E}[\mathbb{E}[D_{t,i} \mid C_{t,i-1}, C_{t,i}, \mathscr{F}_{t-1}] \mid \mathscr{F}_{t-1}] && \text{(by the tower rule)} \\
&= \mathbb{E}\left[(C_{t,i-1} - C_{t,i}) \frac{(1-p_{t,2i-1})(1-p_{t,2i})}{1-p_{t,2i-1}p_{t,2i}} \mid \mathscr{F}_{t-1}\right] && \text{(by (6.5))} \\
&= \mathbb{E}[C_{t,i-1} \mid \mathscr{F}_{t-1}](1-p_{t,2i-1}p_{t,2i}) \frac{(1-p_{t,2i-1})(1-p_{t,2i})}{1-p_{t,2i-1}p_{t,2i}} && \text{(by (6.7))} \\
&= \mathbb{E}[C_{t,i-1} \mid \mathscr{F}_{t-1}](1-p_{t,2i-1})(1-p_{t,2i}),
\end{aligned}$$

and similarly by (6.6), (6.7) and (6.8), we finally obtain

$$\mathbb{E}[E_{t,i} \mid \mathscr{F}_{t-1}] = \mathbb{E}[C_{t,i-1} \mid \mathscr{F}_{t-1}] p_{t,2i-1}(1-p_{t,2i}). \qquad \blacksquare$$

An initial observation is that the all-ones bitstring cannot be sampled in the initial population P_0 with overwhelming probability. The following lemma, similar to the results of Lemma 10, gives upper bounds on the expectations of the random variables Z_t^* and Z_t (defined in (6.2) and (6.3), respectively) in iteration $t = 0$.

Lemma 22. $\mathbb{E}[Z_0] \leq \mathbb{E}[Z_0^*] = O(\log \lambda)$.

By the definition of the random variable Z_t, the $2Z_t$ first marginals are set to the upper border $1 - 1/n$ in iteration $t \in \mathbb{N}$. Recall that the random variable $X_{t,i}$ denotes the number of 1s in bit position $i \in [n]$ among the μ fittest individuals, which is used to update the probabilistic model of the UMDA. We also define another random variable $Y_{t,j}$ to be the number of 11s sampled in any block position $j \in [m]$, also among the μ fittest individuals in an iteration $t \in \mathbb{N}$.

Lemma 23. *It holds for any $t \in \mathbb{N}$ that*

(a) $Y_{t,j} \sim \mathrm{Bin}\left(\mu, p_{t,2j-1}p_{t,2j}\right)$ *for all* $j \geq Z_t + 2$, *and*

(b) $X_{t,i} \sim \mathrm{Bin}\left(\mu, p_{t,i}\right)$ *for all* $i \geq 2Z_t + 3$.

Proof. We first prove **(a)**. By the definition of the random variable Z_t, we know that $C_{t,Z_t} \geq \mu$ and $C_{t,Z_t+1} < \mu$. Consider the block $j := Z_t + 2$. We then obtain from (6.4) that $C_{t,j} \sim \mathrm{Bin}\left(C_{t,j-1}, p_{t,2j-1}p_{t,2j}\right)$. The $\mu - C_{t,j-1} > 0$ remaining individuals (among the μ fittest individuals) have the block $j-1$ set to $\{00, 10, 01\}$. Therefore, their fitness values (or

rankings) have been already decided by the $j-1$ first blocks. What is sampled in the block j has no impact on the ranking of these individuals. There is no bias in block j among these individuals, which means that the number of 11s sampled here follows a binomial distribution with $\mu - C_{t,j-1}$ trials and success probability $p_{t,2j-1}p_{t,2j}$, i.e., $\text{Bin}\left(\mu - C_{t,j-1}, p_{t,2j-1}p_{t,2j}\right)$. Putting things together, the total number of 11s sampled in the block j among the μ fittest individuals equals

$$\begin{aligned} Y_{t,j} &\sim C_{t,j} + \text{Bin}\left(\mu - C_{t,j-1}, p_{t,2j-1}p_{t,2j}\right) \\ &= \text{Bin}\left(C_{t,j-1}, p_{t,2j-1}p_{t,2j}\right) + \text{Bin}\left(\mu - C_{t,j-1}, p_{t,2j-1}p_{t,2j}\right) \\ &= \text{Bin}\left(\mu, p_{t,2j-1}p_{t,2j}\right). \end{aligned}$$

We note that this result holds for any block $j \geq Z_t + 2$, which proves the first statement.

For statement **(b)**, we consider bit position $i = 2j-1$ in block $j = Z_t + 2$. We note that the number of 1s sampled in bit position i can be written as the sum of three terms:

(1) the number of individuals with at least j leading 11s (i.e., $C_{t,j}$),

(2) the number of individuals with $j-1$ leading 11s, followed by a 10 block in block j (i.e., $E_{t,j}$), and

(3) the number of 1s sampled in bit position i among all the μ fittest individuals except the top $C_{t,j-1}$ individuals. There is no bias in favour of any bit at bit position i, so the total number of 1s sampled here among these individuals is binomially distributed with parameters $\mu - C_{t,j-1}$ and $p_{t,i}$.

We note further that the sum of $C_{t,j} + E_{t,j}$ equals the number of 1s sampled in bit position i among the $C_{t,j-1}$ fittest individuals. Thus, we get:

$$\begin{aligned} X_{t,i} &\sim C_{t,j} + E_{t,j} + \text{Bin}\left(\mu - C_{t,j-1}, p_{t,i}\right) \\ &= \text{Bin}\left(C_{t,j-1}, p_{t,i}\right) + \text{Bin}\left(\mu - C_{t,j-1}, p_{t,i}\right) \\ &= \text{Bin}\left(\mu, p_{t,i}\right). \end{aligned}$$

By the same line of argumentation, we can show that the number of 1s sampled in bit position $i+1$ is $X_{t,i+1} \sim \text{Bin}\left(\mu, p_{t,i+1}\right)$, and similarly for other bit positions from $i+3$ to n. ■

We now consider block $i = Z_t + 1$, where $C_{t,i} < \mu$ by the definition of Z_t in (6.2). The following lemma (which is probably the most important lemma in the chapter) shows that if the value of the random variable $C_{t,i}$ is below a threshold, then in iteration $t+1$ the number of individuals with at least i leading 1s sampled decreases, while the number of individuals with exactly $i-1$ leading 1s followed by a 00 increases in expectation and also with high probability. We consider two different regimes of the selection rate, i.e., $\mu/\lambda < 1/(2e)$ and $\mu/\lambda \geq 1/(2e)$.

Lemma 24. *Consider the block $i = Z_t + 1$ in an arbitrary iteration $t \in \mathbb{N}$, and assume further that $C_{t,i} + D_{t,i} \geq \mu$.*

(A) *If $\mu/\lambda < 1/(2e)$ and there exists a constant $\varepsilon \in (0,1)$ such that $C_{t,i} \leq (\mu^2/\lambda)(1-\varepsilon) < (\mu/2e)(1-\varepsilon)$, then*

- **A.1)** $\mathbb{E}[C_{t+1,i} \mid \mathscr{F}_t] < C_{t,i}(1-\varepsilon)$,
- **A.2)** $\mathbb{E}[C_{t+1,i} + D_{t+1,i} \mid \mathscr{F}_t] > \mu(1+\varepsilon^2)$,
- **A.3)** $\Pr(C_{t+1,i} + D_{t+1,i} \leq \mu) = 2^{-\Omega(\mu)}$, *and*
- **A.4)** $\Pr(C_{t+1,i} \geq (\mu^2/\lambda)(1-\varepsilon)) = 2^{-\Omega(\mu^2/\lambda)}$.

(B) *Otherwise, if $\mu/\lambda \geq 1/(2e)$ and there exists a constant $\varepsilon \in (0,1)$ such that $C_{t,i} \leq (\mu/2e)(1-\sqrt{\alpha})$, where $\alpha := 2e(1+\varepsilon)(\mu/\lambda) - 1 \geq \varepsilon$, then*

- **B.1)** $\mathbb{E}[C_{t+1,i} \mid \mathscr{F}_t] < C_{t,i}(1-\sqrt{\varepsilon})$,
- **B.2)** $\mathbb{E}[C_{t+1,i} + D_{t+1,i} \mid \mathscr{F}_t] > \mu(1+\varepsilon)$,
- **B.3)** $\Pr(C_{t+1,i} + D_{t+1,i} \leq \mu) = 2^{-\Omega(\mu)}$, *and*
- **B.4)** $\Pr(C_{t+1,i} \geq \mu(1-\sqrt{\alpha})/2e) = 2^{-\Omega(\lambda)}$.

Proof. The assumption $C_{t,i} + D_{t,i} \geq \mu$ implies that the two marginals $p_{t,2i-1}$ and $p_{t,2i}$ in the block i will be set to $C_{t,i}/\mu$ when updating the model in iteration t. By the definition of Z_t, we know that $C_{t+1,i-1} = C_{t+1,Z_t} \sim \text{Bin}\left(\lambda, (1-1/n)^{2(i-1)}\right)$. Statement (A.1) is then trivial

since

$$\begin{aligned}
\mathbb{E}[C_{t+1,i} \mid \mathscr{F}_t] &= \mathbb{E}[C_{t+1,i-1} \mid \mathscr{F}_t] p_{t+1,2i-1} p_{t+1,2i} && \text{(by (6.7))} \\
&= \lambda(1-1/n)^{2(i-1)}(C_{t,i}/\mu)(C_{t,i}/\mu) \\
&< \lambda(\mu/\lambda)(1-\varepsilon)(C_{t,i}/\mu) && \text{(as } C_{t,i}/\mu \leq (\mu/\lambda)(1-\varepsilon)\text{)} \\
&= C_{t,i}(1-\varepsilon).
\end{aligned}$$

Noting also that $C_{t,i}/\mu \leq (\mu/\lambda)(1-\varepsilon) < (1-\varepsilon)/2e < (1-\varepsilon)/2$. The statement (A.2) can be shown as follows.

$$\begin{aligned}
\mathbb{E}[C_{t+1,i} + D_{t+1,i} \mid \mathscr{F}_t] &= \lambda(1-1/n)^{2(i-1)}((C_{t,i}/\mu)^2 + (1-C_{t,i}/\mu)^2) && \text{(by (6.7) \& (6.8))} \\
&\geq (\lambda/e)(1-2(C_{t,i}/\mu)(1-C_{t,i}/\mu)) \\
&\geq \lambda(1-(1-\varepsilon)(1-(1-\varepsilon)/2)) \\
&= (\lambda/2e)(1+\varepsilon^2) \\
&> \mu(1+\varepsilon^2).
\end{aligned}$$

For statement (A.3), we now associate each of the λ individuals in the population with an indicator random variable, which is set to one only if the individual has $i-1$ leading 11s, followed by either a 11 or a 00. There are λ such indicators, and we are interested in their sum, which is identical to the sum of $D_{t+1,i} + C_{t+1,i}$. By statement (A.2), the expectation of the sum is at least $\mu(1+\varepsilon^2) = \mu/(1-\delta)$ for some constants $\varepsilon \in (0,1)$ and $\delta := 1 - 1/(1+\varepsilon^2) \in (0,1)$. Then, by a Chernoff bound the probability that the sum is at most $(1-\delta)\mu/(1-\delta) = \mu$ is at most $e^{-(\delta^2/2)\mu/(1-\delta)} = 2^{-\Omega(\mu)}$.

For statement (A.4), we note that $C_{t+1,i}$ is stochastically dominated by another random variable $\tilde{C}$, which is binomially distributed with λ trials and success probability $(\mu/\lambda)^2(1-\varepsilon)^2$. Note that $\mathbb{E}[\tilde{C}] = (\mu^2/\lambda)(1-\varepsilon)^2 = \Omega(\mu^2/\lambda)$; thus, we can rewrite $(\mu^2/\lambda)(1-\varepsilon) = \mathbb{E}[\tilde{C}]/(1-\varepsilon) = (1+\varepsilon')\mathbb{E}[\tilde{C}]$ for some other constant $\varepsilon' = 1/(1-\varepsilon) - 1 > 0$. We then obtain

$$\begin{aligned}
\Pr(C_{t+1,i} \geq (\mu^2/\lambda)(1-\varepsilon)) &\leq \Pr(\tilde{C} \geq (\mu^2/\lambda)(1-\varepsilon)) && \text{(by Definition 8)} \\
&= \Pr(\tilde{C} \geq (1+\varepsilon')\mathbb{E}[\tilde{C}]) \\
&\leq e^{-(\varepsilon')^2\mathbb{E}[\tilde{C}]/3} && \text{(by a Chernoff bound)} \\
&= 2^{-\Omega(\mu^2/\lambda)},
\end{aligned}$$

which completes proof of statement (A.4).

To prove statement (B.1), note that

$$\begin{aligned}
C_{t,i}/\mu &\le (1/(2e))(1-\sqrt{\alpha}) && \text{(by assumption)}\\
&= (1/(2e))(1-\sqrt{2e(1+\varepsilon)(\mu/\lambda)-1})\\
&\le (1/(2e))(1-\sqrt{2e(1+\varepsilon)(1/(2e))-1}) && \text{(as } \mu/\lambda \ge 1/(2e)\text{)}\\
&= (1-\sqrt{\varepsilon})/(2e)\\
&< (1-\sqrt{\varepsilon})/2.
\end{aligned}$$

Therefore, statement (B.1) can be shown as follows.

$$\begin{aligned}
\mathbb{E}[C_{t+1,i} \mid \mathscr{F}_t] &= \mathbb{E}[C_{t,i-1} \mid \mathscr{F}_t]\, p_{t,2i-1} p_{t,2i} && \text{(by (6.7))}\\
&\le \lambda (C_{t,i}/\mu)(C_{t,i}/\mu) && \text{(as } C_{t,i-1} \le \lambda\text{)}\\
&= (\lambda/\mu)(C_{t,i}/\mu) C_{t,i}\\
&\le (2e)((1-\sqrt{\varepsilon})/2e) C_{t,i}\\
&= C_{t,i}(1-\sqrt{\varepsilon}).
\end{aligned}$$

For statement (B.2), we note that $C_{t,i}/\mu \le (1-\sqrt{\alpha})/2e < (1-\sqrt{\alpha})/2$ and then obtain

$$\begin{aligned}
\mathbb{E}[C_{t+1,i} + D_{t+1,i} \mid \mathscr{F}_t] &\ge (\lambda/e)(1-2(C_{t,i}/\mu)(1-C_{t,i}/\mu)) && \text{(by (6.7) \& (6.8))}\\
&> (\lambda/e)(1-(1-\sqrt{\alpha})(1-(1-\sqrt{\alpha})/2)))\\
&= (\lambda/e)(1-(1-\sqrt{\alpha})+(1/2)(1-\sqrt{\alpha})^2)\\
&= (\lambda/e)(\sqrt{\alpha}+(1/2)(1-2\sqrt{\alpha}+\alpha))\\
&= (\lambda/2e)(1+\alpha)\\
&= (\lambda/2e)2e(1+\varepsilon)(\mu/\lambda)\\
&= \mu(1+\varepsilon).
\end{aligned}$$

The statement (B.3) follows similarly to the proof of statement (A.3). For the statement (B.4), we employ a similar approach used in (A.4), where we choose $\tilde{C} \sim \mathrm{Bin}\left(\lambda, ((1-\sqrt{\alpha})/2e)^2\right)$

and $\mathbb{E}[\tilde{C}] = \lambda((1-\sqrt{\alpha})/2e)^2 = \Omega(\lambda)$. We also note that

$$\begin{aligned}\mu(1-\sqrt{\alpha})/2e &= \mathbb{E}[\tilde{C}]/((\lambda/\mu)((1-\sqrt{\alpha})/2e)) \\ &\geq \mathbb{E}[\tilde{C}]/(1-\sqrt{\alpha}) \\ &\geq \mathbb{E}[\tilde{C}]/(1-\varepsilon) \\ &= (1+\varepsilon')\mathbb{E}[\tilde{C}]\end{aligned}$$

for some other constant $\varepsilon' = 1/(1-\varepsilon) - 1 > 0$ and $\alpha \geq \varepsilon$. Since $\tilde{C}$ stochastically dominates $C_{t+1,i}$, we get by a Chernoff bound that

$$\begin{aligned}\Pr(C_{t+1,i} \geq \mu(1-\sqrt{\alpha})/2e) &\leq \Pr(\tilde{C} \geq \mu(1-\sqrt{\alpha})/2e) && \text{(by Definition 8)} \\ &\leq \Pr(\tilde{C} \geq (1+\varepsilon')\mathbb{E}[\tilde{C}]) \\ &\leq e^{-(\varepsilon')^2\mathbb{E}[\tilde{C}]/3} && \text{(by a Chernoff bound)} \\ &\leq 2^{-\Omega(\lambda)},\end{aligned}$$

which completes the proof. ■

In essence, Lemma 24 tells us that when $\mu/\lambda < 1/(2e)$ and in an iteration $t \in \mathbb{N}$ the block $Z_t + 1$ consists of 11s and 00s only among the μ fittest individuals such that the number of 11s is at most $(\mu^2/\lambda)(1-\varepsilon) < (\mu/2e)(1-\varepsilon)$, then in the next iteration the number of 11s sampled among the μ fittest individuals becomes smaller in expectation (see the statement (A.1)). This block consists of only 11s and 00s among the μ fittest individuals in expectation (see the statement (A.2)). Moreover, these events also happen with probability exponentially close to one (see the statements (A.3) & (A.4)). Similar events also occur when $\mu/\lambda \geq 1/(2e)$ as shown in statement (B). Note in particular that the statement (B.4) has a probability of at most $2^{-\Omega(\lambda)}$, which is asymptotically significantly smaller than the probability of $2^{-\Omega(\mu^2/\lambda)}$ in the statement (A.4). This means that we only need to consider the case when $\mu/\lambda < 1/(2e)$ since the result will also hold for any selection rate $\mu/\lambda \geq 1/(2e)$. Hence, we shall consider the case $\mu/\lambda < 1/(2e)$ and form our arguments based on statement (A) for the rest of the chapter. By combining the two conditions in the statement (A), we get $C_{t,i} < \mu(1-\varepsilon)/(2e) < \mu/(2e)$ for some small constant $\varepsilon \in (0,1)$.

We can rewrite the statement (A.1) as follows

$$\mathbb{E}[C_{t,i} - C_{t+1,i} \mid \mathscr{F}_t] > \varepsilon C_{t,i}.$$

This means that the stochastic process $(C_{t,i} : t \in \mathbb{N})$ has a (multiplicative) drift towards the value of zero. By applying the multiplicative drift theorem (see Theorem 2) for a potential function $\phi(C_t) = C_t$ with an initial distance of $\phi(C_0) < \mu$, the random variable $C_{t,i}$ will hit the value of zero within $(1+\log\mu)/\varepsilon = \mathrm{O}(\log\mu)$ iterations in expectation. Once this has happened, we will show that the UMDA requires at least $2^{\Omega(\mu)}$ iterations in expectation to sample at least

$$\nu := (\mu^2/\lambda)(1-\varepsilon) = (\mu/\lambda)\mu(1-\varepsilon) < \mu/(2e) \tag{6.10}$$

11s to have a drift towards the upper border to escape the 'trap' in the block $Z_t + 1$ (another way of saying this is to repair the specified block).

Furthermore, we note so far Lemma 24 assumes that $C_{t,i} + D_{t,i} \geq \mu$, which means that there are only 11s and 00s among the μ fittest individuals in block $i = Z_t + 1$. In this situation, the algorithm updates the two corresponding marginals to $C_{t,i}/\mu$ (where we assume further that $C_{t,i} \leq \nu$). However, this can be relaxed to cover other situations because the UMDA updates each marginal using the total number of 1s sampled at the bit position independent of all other bit positions. More precisely, as long as the number of 1s among the μ fittest individuals at each bit position is still below the threshold ν, then all results in Lemma 24 still hold.

Recall that we aim at showing an $2^{\Omega(\mu)}$ lower bound on the runtime of the UMDA on the DLB function. This lower bound will be obtained if we can show that there exists a block $i = Z_t + 1 < m$ between Z_0 and $m = n/2$, where its marginals are deceived into reaching the lower bound $1/n$, and then the UMDA has to wait a long time (in terms of iterations) while the block is being repaired.

Lemma 25 (Second moment [Weisstein])**.** *Let $n \in \mathbb{N}$ and $p \in [0,1]$. Let $X \sim \mathrm{Bin}(n,p)$. Then,* $\mathbb{E}\left[X^2\right] = np((n-1)p+1)$.

Lemma 26. *Let $\delta \in (0,1]$ be some constant. Let $\mu \leq (\delta/8)n + 1$. If there exists a constant $k < m$ such that $Z_t \leq k-2$ for any time $t \in \mathbb{N}$, then it holds for any $j \in [2k-1, n]$ that*

(a) $\mathbb{E}\left[p_{t,j}\right] = 1/2$,

(b) $\mathbb{E}\left[X_{t,j}\right] = \mu/2$, *and*

(c) $\mathrm{Var}\left[X_{t,j}\right] \geq (\mu^2/4)(1-\delta)(1-(1-1/\mu)^t)$.

Proof. For readability, we omit the index j through out the proof. Statement (a) has been proven in Lemma 16. Statement (b) follows from (5.13) that

$$\mathbb{E}[X_t] = \mu \cdot \mathbb{E}[p_t] = \mu/2.$$

For statement (c), we obtain

$$\begin{aligned}
\mathbb{E}[X_t^2] &= \mathbb{E}[\mathbb{E}[X_t^2 \mid p_t]] && \text{(by the tower rule)} \\
&= \mathbb{E}[\mu p_t(p_t(\mu-1)+1)] && \text{(by Lemma 25)} \\
&= \mu(\mu-1)\mathbb{E}[p_t^2] + \mu\mathbb{E}[p_t] \\
&= \mu(\mu-1)\mathbb{E}[p_t^2] + \mu/2.
\end{aligned}$$

By the definition of expectation, we also have

$$\mathbb{E}[p_t^2] = (1/n)^2 \Pr(X_{t-1}=0) + (1-1/n)^2 \Pr(X_{t-1}=\mu) + \sum_{k=1}^{\mu-1} (k/\mu)^2 \Pr(X_{t-1}=k),$$

which by noting that

$$\mathbb{E}[X_{t-1}^2] = \mu^2 \Pr(X_{t-1}=\mu) + \sum_{k=1}^{\mu-1} k^2 \Pr(X_{t-1}=k)$$

satisfies

$$\begin{aligned}
\mathbb{E}[p_t^2] &= (1/n)^2 \Pr(X_{t-1}=0) + (1-1/n)^2 \Pr(X_{t-1}=\mu) \\
&\quad + (1/\mu^2)(\mathbb{E}[X_{t-1}^2] - \mu^2 \Pr(X_{t-1}=\mu)) \\
&= (1/\mu^2)\mathbb{E}[X_{t-1}^2] + (1/n)^2 \Pr(X_{t-1}=0) \\
&\quad + ((1-1/n)^2 - 1)\Pr(X_{t-1}=\mu) \\
&= (1/\mu^2)\mathbb{E}[X_{t-1}^2] - (2/n)(1-1/n)\Pr(X_{t-1}=\mu) \\
&\geq (1/\mu^2)\mathbb{E}[X_{t-1}^2] - (2/n)(1-1/n) && (\text{as } \Pr(X_{t-1}=\mu) \leq 1)
\end{aligned}$$

Thus,

$$\begin{aligned}
\mathbb{E}[X_t^2] &= \mu(\mu-1)\mathbb{E}[p_t^2] + \mu/2 \\
&\geq (1-1/\mu)\mathbb{E}[X_{t-1}^2] - 2(\mu/n)(\mu-1)(1-1/n) + \mu/2 \\
&= (1-1/\mu)\mathbb{E}[X_{t-1}^2] + (\mu/2 - 2(\mu/n)(\mu-1)(1-1/n)) \\
&= \alpha \cdot \mathbb{E}\left[X_{t-1}^2\right] + \beta,
\end{aligned}$$

where $\alpha := 1-1/\mu \in (0,1)$ and $\beta := \mu/2 - 2(\mu/n)(\mu-1)(1-1/n)$. We now obtain a recurrence relation for the expectation of X_t^2 on time t and by $\sum_{i=0}^{n} x^i = (1-x^{n+1})/(1-x)$ for any $x \neq 1$ [19, p. 1147], we then get

$$\begin{aligned}
\mathbb{E}[X_t^2] &\geq \alpha \cdot \mathbb{E}\left[X_{t-1}^2\right] + \beta \\
&\geq \alpha(\alpha\mathbb{E}\left[X_{t-2}^2\right] + \beta) + \beta \\
&= \alpha^2\mathbb{E}\left[X_{t-2}^2\right] + \beta(1+\alpha) \\
&\geq \alpha^t\mathbb{E}\left[X_0^2\right] + \beta\sum_{i=0}^{t-1}\alpha^i \\
&= \alpha^t\mathbb{E}\left[X_0^2\right] + \beta\frac{1-\alpha^t}{1-\alpha} \\
&= \alpha^t\mathbb{E}\left[X_0^2\right] + \beta\mu(1-\alpha^t) && \text{(as } 1-\alpha = 1/\mu\text{)} \\
&= \beta\mu - \alpha^t(\beta\mu - \mathbb{E}\left[X_0^2\right]).
\end{aligned}$$

By Lemma 23, $X_0 \sim \text{Bin}(\mu, 1/2)$, and by Lemma 25 we obtain $\mathbb{E}[X_0^2] = \mu(1/2)((1/2)(\mu-1)+1) = \mu(\mu+1)/4 > (\mu/2)^2$. We also get

$$\begin{aligned}
\text{Var}[X_t] &= \mathbb{E}[X_t^2] - \mathbb{E}[X_t]^2 \\
&\geq \beta\mu - \alpha^t(\beta\mu - \mathbb{E}\left[X_0^2\right]) - (\mu/2)^2 \\
&= (\beta\mu - (\mu/2)^2) - \alpha^t(\beta\mu - \mathbb{E}\left[X_0^2\right]) \\
&> (\beta\mu - (\mu/2)^2) - \alpha^t(\beta\mu - (\mu/2)^2) && \text{(as } \mathbb{E}\left[X_0^2\right] > (\mu/2)^2\text{)} \\
&= (\beta\mu - (\mu/2)^2)(1-\alpha^t).
\end{aligned}$$

We know that

$$\begin{aligned}
\beta\mu - (\mu/2)^2 &= \mu(\mu/2 - 2(\mu/n)(\mu-1)(1-1/n)) - \mu^2/4 \\
&= \mu^2/2 - 2(\mu^2/n)(\mu-1)(1-1/n) - \mu^2/4 \\
&= \mu^2/4 - 2(\mu^2/n)(\mu-1)(1-1/n) \\
&= (\mu^2/4)(1-(8/n)(\mu-1)(1-1/n)) \\
&> (\mu^2/4)(1-(8/n)(\mu-1)) && \text{(as } 1-1/n<1\text{)} \\
&\geq (\mu^2/4)(1-\delta). && \text{(as } \mu \leq (\delta/8)n+1\text{)}
\end{aligned}$$

Putting everything together, we get

$$\mathrm{Var}[X_t] \geq (\mu^2/4)(1-\delta)(1-(1-1/\mu)^t),$$

which completes the proof of statement (c). ■

Furthermore, in case of no borders, the variance of $X_{t,j}$ for any bit $j \in [2k-1,n]$, where k is defined in Lemma 26, can be expressed as a function of time as $(1-(1-1/\mu)^t)(\mu^2/4)$, which can be derived by applying the law of total variance on the martingale $X_{t,j} \sim \mathrm{Bin}\left(\mu, X_{t-1,j}/\mu\right)$. See [47, Lemma 7 & Corollary 9] for a similar derivation for the cGA without borders. Surprisingly, the statement (c) in Lemma 26 tells us that the variance of $X_{t,j}$ for the UMDA with borders grows asymptotically in the same order of magnitude as the variance for the UMDA without borders.

Recall that we shall show that there exists a block $i = Z_t + 1 < m$ between Z_0 and $m = n/2$. where its marginals are deceived into reaching the lower border $1/n$. The number of 1s sampled among the μ fittest individuals at either of the two bit positions forms a martingale $(X_t : t \in \mathbb{N})$ on the state space $\{0\} \cup [\mu]$. We now show that when the variance $\mathrm{Var}[X]$ becomes sufficiently large, the chance of sampling less than ν (defined in 6.10) 1s at either of the two bit positions is a constant.

Lemma 27. *Let $\mu \in \mathbb{N}$, $p \in [0,1]$ and $0 \leq \nu < \mu/2$. Let $(X_t : t \in \mathbb{N})$ be a martingale on the state space $\{0\} \cup [\mu]$ such that $\mathbb{E}[X_0] = \mu/2$ and $X_{t+1} \sim \mathrm{Bin}(\mu, X_t/\mu)$. If there exists some constant $\delta \in (0,1)$ such that $\mathrm{Var}[X_t] \geq (0.4689/\delta)^2\mu^2$ and*

$$\Pr\left(X_t = \frac{\mu}{2} - i\right) = \Pr\left(X_t = \frac{\mu}{2} + i\right) \tag{6.11}$$

for all $i \in [\mu/2]$*, then*

$$\Pr(X_t \leq \nu) \geq \frac{1-\delta}{2}.$$

Proof. Due to symmetry in (6.11), we get $\mathbb{E}[X_t] = \mu/2$ and $\Pr(X_t \leq \nu) = \Pr(X_t \geq \mu - \nu)$. Thus,

$$\begin{aligned}
\Pr(X_t \leq \nu) &= \frac{1}{2}(1 - \Pr(\nu < X_t < \mu - \nu)) \\
&\geq \frac{1}{2}\left(1 - \frac{(\mu - 2\nu)\eta}{\sqrt{\mathrm{Var}[X_t]}}\right) && \text{(by Theorem 8)} \\
&\geq \frac{1}{2}\left(1 - \frac{(\mu - 2\nu)\eta}{\sqrt{(0.4689\mu/\delta)^2}}\right) && \text{(as } \mathrm{Var}[X_t] \geq (0.4689\mu/\delta)^2\text{)} \\
&= \frac{1}{2}\left(1 - \frac{(1 - 2(\nu/\mu))\eta}{0.4689/\delta}\right) \\
&\geq \frac{1}{2}\left(1 - \frac{\delta\eta}{0.4689}\right) && \text{(as } 0 < 1 - 2(\nu/\mu) \leq 1\text{)} \\
&\geq \frac{1}{2}(1 - \delta), && \text{(as } \eta < 0.4689\text{, Theorem 8)}
\end{aligned}$$

which completes the proof. ■

Lemma 28. *Let* $\delta_1, \delta_2 \in (0,1)$ *be two constants. Let* $\mu \leq (\frac{\delta_1}{8})n + 1$*. If there exists a constant* $k < m$ *such that* $Z_t \leq k - 2$ *for any time* $t \in \mathbb{N}$*, then for any* $j \in [2k-1, n]$*, it holds that* $\mathrm{Var}\left[X_{t,j}\right] \geq (\frac{0.4689}{\delta_2})^2\mu^2$ *for any* $t \geq \mu(-\ln(1 - \frac{4}{1-\delta_1}(\frac{0.4689}{\delta_2})^2))$.

Proof. By Lemma 26, we know that $\mathrm{Var}\left[X_{t,j}\right] \geq (\mu^2/4)(1-\delta_1)(1-(1-1/\mu)^t)$. To satisfy the condition $\mathrm{Var}\left[X_{t,j}\right] \geq (0.4689/\delta_2)^2\mu^2$, we just need to find $t \in \mathbb{N}$ such that

$$\frac{1-\delta_1}{4}\left(1 - \left(1 - \frac{1}{\mu}\right)^t\right) \geq \left(\frac{0.4689}{\delta_2}\right)^2.$$

Solving for t yields

$$t \geq \frac{\ln(1 - \frac{4}{1-\delta_1}(\frac{0.4689}{\delta_2})^2)}{\ln(1 - \frac{1}{\mu})} = \frac{-\ln(1 - \frac{4}{1-\delta_1}(\frac{0.4689}{\delta_2})^2)}{\ln(\frac{\mu}{\mu-1})}.$$

Since $\ln(\frac{\mu}{\mu-1}) \geq 1 - \frac{1}{\mu/(\mu-1)} = 1 - \frac{\mu-1}{\mu} = \frac{1}{\mu}$ [log], the inequality above still holds for any $t \geq \mu(-\ln(1 - \frac{4}{1-\delta_1}(\frac{0.4689}{\delta_2})^2))$. ■

6.3.1 An exponential runtime under moderate selection rates

In this section, we will show that the UMDA requires an exponential expected runtime to optimise the DLB function. To proceed, we will analyse the situation that

1) from the beginning to the smallest iteration $t^* \in \mathbb{N}$ such that the number of 1s sampled in each bit position in the block $Z_{t^*}+1$ is at most $\nu = (\mu^2/\lambda)(1-\delta)$ for some constant $\delta \in (0,1)$, and

2) from the iteration t^* onward.

We will show that phase 1 will last for $\Omega(\mu)$ iterations. The following result, which follows the same proof of Lemma 18, shows the difference between the two random variables Z_t^* and Z_t, defined in (6.2) and (6.3), respectively.

Lemma 29. *It holds for any $t \in \mathbb{N}$ that $\mathbb{E}[Z_t^* - Z_t] = O(\log \mu)$.*

We now show that the all-ones bitstring cannot be sampled during the $\Omega(\mu)$ first iterations with overwhelming probability. The proof is inspired in part by the proof of [71, Lemma 9].

Lemma 30. *Consider the* UMDA *with $\mu = o(n/\log n)$ optimising the* DLB *function. The all-ones bitstring cannot be sampled during the $\lfloor 2.267\mu \rfloor$ first iterations with probability $1 - 2^{-\Omega(n)}$.*

Proof. We know by Lemma 29 that $\mathbb{E}[Z_t^* - Z_t] = \mathrm{O}(\log n)$ for any $t \in \mathbb{N}$. Therefore, the algorithm requires at least $\Omega(n/\log n)$ iterations in expectation to optimise the DLB function, assuming the best-case scenario in which the random variable Z_t^* gets increased by $\mathrm{O}(\log n)$ in each iteration. During the $o(n/\log n)$ first iterations, the maximum expected value of the random variable Z_t^* is at most $o(n/\log n) \cdot \mathrm{O}(\log n) = o(n)$. By definition, there exists a constant $c \in (0,1)$ such that $Z_t^* < cn/2$ during the $o(n/\log n)$ first iterations for n large enough. Furthermore, all results in Lemma 26 hold for at least $(1-c)n = \Omega(n)$ last bit positions during the same period.

We now consider an arbitrary bit position among the $\Omega(n)$ last bit positions and will upper bound the probability of such a marginal exceeding the threshold value of $99/100$ during the $\Omega(\mu)$ first iterations. For readability, we omit the index j and consider the potential $\phi_t := \phi(X_t) = X_t^2$. By Lemma 23, we can pessimistically assume that we are not at any border, which implies that $X_{t+1} \sim \text{Bin}(\mu, X_t/\mu)$, and the expected single-step change is

$$\mathbb{E}[\phi_{t+1} - \phi_t \mid \mathscr{F}_t] = \mu \frac{X_t}{\mu}\left(\frac{X_t}{\mu}(\mu-1)+1\right) - X_t^2 = X_t\left(1 - \frac{X_t}{\mu}\right) < \frac{\mu}{4}.$$

Let $\tilde{T} := \min\{t \in \mathbb{N} : X_t \geq 99\mu/100\}$, i.e., the first hitting time of the value of $99\mu/100$ in the stochastic process $(X_t : t \in \mathbb{N})$. We have a Markov chain $\phi_{\tilde{T}}$ with process $\phi_t = X_t^2$ starting at $(\mu/2)^2$ and then progressing by $\phi_{t+1} - \phi_t$ for $\tilde{T}$ iterations. We then get

$$\mathbb{E}[\phi_{\tilde{T}}] = \left(\frac{\mu}{2}\right)^2 + \sum_{t=0}^{\tilde{T}-1} \mathbb{E}[\mathbb{E}[\phi_{t+1} - \phi_t] \mid \mathscr{F}_t] < \left(\frac{\mu}{2}\right)^2 + \tilde{T} \cdot \frac{\mu}{4}.$$

Applying Markov's inequality to the random variable $\phi_{\tilde{T}}$ yields

$$\Pr\left(\phi_{\tilde{T}} \geq k\left(\frac{\mu^2}{4} + \tilde{T} \cdot \frac{\mu}{4}\right)\right) \leq \Pr\left(\phi_{\tilde{T}} \geq k\mathbb{E}\left[\phi_{\tilde{T}}\right]\right) \leq \frac{1}{k}.$$

We want that $(99\mu/100)^2 \geq k\left(\mu^2/4 + \tilde{T}\mu/4\right)$ since then

$$\Pr\left(\phi_{\tilde{T}} \geq \left(\frac{99\mu}{100}\right)^2\right) \leq \Pr\left(\phi_{\tilde{T}} \geq k\left(\frac{\mu^2}{4} + \tilde{T} \cdot \frac{\mu}{4}\right)\right) \leq \frac{1}{k}.$$

We get $\tilde{T} \leq \mu((4/k)(99/100)^2 - 1)$, which is positive as long as $k \in (1, 4(99/100)^2)$. Thus, we can choose $k = 1.2$ and then obtain $\tilde{T} \leq 2.267\mu = o(n/\log n)$. During the $\lfloor 2.267\mu \rfloor$ first iterations, the probability that an arbitrary marginal among the $\Omega(n)$ last bit positions exceeds $99/100$ is at most $1/k = 1/1.2 < 0.84$. The complementary event occurs with probability of at least $1 - 0.84 = 0.16$. In expectation, there are at least $0.16 \cdot \Omega(n)$ such bit positions among the $\Omega(n)$ last bit positions. By a Chernoff bound, there are at least $(1-\delta) \cdot 0.16 \cdot \Omega(n) = \Omega(n)$ such bit positions for some constant $\delta \in (0,1)$ with probability at least $1 - e^{-\delta^2 \cdot 0.16 \cdot \Omega(n)/2} = 1 - 2^{-\Omega(n)}$. Since the all-ones bitstring can be sampled only if the $\Omega(n)$ last bit positions are sampled as all ones, the probability of sampling the global optimum is bounded from above by $(99/100)^{\Omega(n)} = 2^{-\Omega(n)}$. ■

In the preceding proof, we know that the algorithm requires at least $\Omega(n/\log n)$ iterations in expectation to optimise the DLB function. Very recently, Doerr & Krejca [34, Lemma 5] show that this holds for the $\Omega(n/\log(\lambda/\mu))$ first iterations, which becomes $\Omega(n)$ if $\lambda = \Theta(\mu)$. We now recall the following result that was already proved in Lemma 15 and Lemma 17. It implies that the samplings at the $\Omega(n)$ last bit positions are mutually independent and the expected values of the marginals of these remaining bits will stay around the value of $1/2$.

Lemma 31. *Consider the situation of Lemma 26. Then, the events of sampling of 11s in any block from k to m are pairwise independent. Furthermore, the all-ones bitstring cannot be sampled with probability at least* $1 - 2^{-\Omega(n)}$.

Theorem 24. *The* UMDA *with a parent population size* $\mu = o(n/\log n)$*, and an offspring population size* $\mu \leq \lambda = O\left(\mu^{2-\varepsilon}\right)$ *for some constant* $\varepsilon \in (0,1]$ *has an expected runtime of* $2^{\Omega(\mu^{\varepsilon})}$ *on the* DLB *function.*

Proof. We start by noting that the all-ones bitstring cannot be sampled during the $\lfloor 2.267\mu \rfloor$ first iterations with probability $1 - 2^{-\Omega(n)}$, and at least $\Omega(n)$ last marginals are still below $99/100$ (see Lemma 30). For now, we assume that the opposite of this typical event does not happen. By choosing $\mu = o(n/\log n)$ and then applying Lemma 28 for two constants $\delta_1 = 0.001$ and $\delta_2 = 0.999$, we know that after at most

$$\mu\left(-\ln\left(1 - \frac{4}{1-0.001}\left(\frac{0.4689}{0.999}\right)^2\right)\right) < 2.14\mu$$

first iterations, the variance of a neutral bit is already at least $(\frac{0.4689}{\delta_2})^2\mu^2$, and thus by Lemma 27, the probability that X_t drops below the threshold value ν, defined in (6.10), is

$$\Pr(X_t \leq \nu) \geq \frac{1-\delta_2}{2} = \frac{1-0.999}{2} = \Omega(1).$$

Consider now $t \geq 2.14\mu$. Assume in iteration $t' = t+1$ that $Z_{t'} = Z_t + 1$ (i.e., Z_t gets increased by one in the next iteration), then we can find with probability $\Omega(1)$ that there are fewer than ν 1s sampled among the μ fittest individuals in any bit position in block $Z_{t'}+1$. Since then, by Lemma 24, there is a multiplicative drift towards the value of zero in the stochastic process $(C_{t'+\Delta t,i} : \Delta t \in \mathbb{N})$ in block $i := Z_{t'} + 1$. By the multiplicative drift theorem (see Theorem 2), the number of 11s sampled there will reduce to zero within $O(\log \mu)$ iterations in expectation.

Once the number of 1s sampled among the μ fittest individuals at each bit position in the i-th block has dropped below the threshold value ν, we say that the UMDA *gets stuck* if the following three events happen simultaneously:

(i) The number of 11s sampled among the μ fittest individuals at the i-th block stays below the threshold ν. By the statement (A.4) in Lemma 24, the complementary event occurs with probability at most $2^{-\Omega(\mu^2/\lambda)} = 2^{-\Omega(\mu/\mu^{1-\varepsilon})} = 2^{-\Omega(\mu^\varepsilon)}$ for some constant $\varepsilon \in (0,1]$.

(ii) The total number of 11s and 00s among the μ fittest individuals there is at least μ. By the statement (A.3) in Lemma 24, the complementary event happens with probability at most $2^{-\Omega(\mu)}$.

(iii) The $\Omega(n)$ last bit positions cannot be sampled as all 1s. The complementary event occurs with probability at most $2^{-\Omega(n)}$ by Lemma 17[1].

By the union bound, either or all events do not happen with probability at most

$$2^{-\Omega(n)} + 2^{-\Omega(\mu)} + 2^{-\Omega(\mu^\varepsilon)} = 2^{-\Omega(\mu^\varepsilon)},$$

since $\mu = \mathrm{O}(n)$ and $\varepsilon \in (0,1]$. Thus, when $t \geq \lfloor 2.14\mu \rfloor$, the UMDA requires at least $1/2^{-\Omega(\mu^\varepsilon)} = 2^{\Omega(\mu^\varepsilon)}$ iterations in expectation until the i-th block has been repaired. Once the block has been repaired, the algorithm will get stuck in some of the $\Omega(n)$ following blocks. Since the probability of sampling less than ν 1s at a bit position among the μ fittest individuals is constant, the algorithm in expectation gets stuck in at least $\Omega(1) \cdot \Omega(n) = \Omega(n)$ other blocks before the global optimum can be found. Therefore, after iteration $t \geq \lfloor 2.14\mu \rfloor$, the UMDA takes at least $\Omega(n) \cdot 2^{\Omega(\mu^\varepsilon)}$ expected iterations further before the global optimum of the DLB function can be found.

To obtain the overall expected runtime, we have to consider the $\lfloor 2.14\mu \rfloor$ first iterations. During this period, we know that the all-ones bitstring cannot be sampled with probability $1 - 2^{-\Omega(n)}$ (by Lemma 30). Noting further that the UMDA performs λ function evaluations

[1] We can re-use the result that was proven for the LEADINGONES function in Chapter 5 because of Statement **(b)** in Lemma 23.

in every iteration. Therefore, by the law of total expectation, we get

$$\begin{aligned}\mathbb{E}[T] &= \mathbb{E}\left[T \cdot \mathbb{1}_{\{T \leq 2.14\mu\}}\right] + \mathbb{E}\left[T \cdot \mathbb{1}_{\{T > 2.14\mu\}}\right] \\ &\geq \mathbb{E}[T \mid T \leq 2.14\mu] \Pr(T \leq 2.14\mu) + \lambda \cdot (2.14\mu + \Omega(n) \cdot 2^{\Omega(\mu^\varepsilon)})) \\ &= \mathbb{E}[T \mid T \leq 2.14\mu] \cdot 2^{-\Omega(n)} + 2.14\mu\lambda + \lambda \cdot \Omega(n) \cdot 2^{\Omega(\mu^\varepsilon)} \\ &= 2^{\Omega(\mu^\varepsilon)},\end{aligned}$$

which completes the proof. ■

We observe that when the population size μ is small, the threshold value ν is not too large, meaning that the UMDA only needs to sample a few 1s to escape the 'current trap' in an arbitrary iteration. Larger population sizes, such as $\mu = \Omega(n^{\varepsilon_1})$ for some constant $\varepsilon_1 \in (0,1)$ and $\lambda = \mathrm{O}\left(\mu^{2-\varepsilon}\right)$ for some constant $\varepsilon \in (0,1]$, will result in an exponential lower bound of $2^{\Omega(n^{\varepsilon\varepsilon_1})}$ on the expected runtime of the UMDA on the DLB function.

6.3.2 Extremely high selection rates may help

We now apply Theorem 4 to derive a polynomial upper bound on the expected runtime of the UMDA with a parent population size $\mu = \Omega(\log n)$ and an offspring population size $\lambda = \Omega(\mu^2)$ on the DLB function.

Theorem 25. *The* UMDA *with a parent population size $\mu \geq c \log n$ for some sufficiently large constant $c > 0$, and an offspring population size $\lambda \geq (1+\delta)e\mu^2$ for any constant $\delta > 0$, has an expected runtime of $O(n\lambda \log \lambda + n^3)$ on the* DLB *function.*

Proof. We use the same levels as defined in (6.1). There are $m = (n/2) + 1$ levels from A_0 to A_m, assuming that n is a multiple of two. We choose $\gamma_0 = \mu/\lambda$.

Condition (G2)

For any level $j \in \{0\} \cup [m-2]$ satisfying $|P_t \cap A_{\geq j}| \geq \gamma_0\lambda = \mu$ and $|P_t \cap A_{\geq j+1}| \geq \lambda\gamma \geq 1$ for all $\gamma \in (0, \gamma_0]$, we seek a lower bound of $(1+\delta)\gamma$ for $\Pr\left(y \in A_{\geq j+1}\right)$ where y is sampled from the model p_{t+1} (defined in 2.3). The given conditions on j imply that the μ fittest individuals of P_t have at least j leading 11s and among them at least $\lceil \gamma\lambda \rceil$ have at least $j+1$ leading 11s. Hence, $p_{t+1,i} = 1 - 1/n$ for all $i \in [2j]$, and for each $i \in \{2j+1, 2j+2\}$ that

$p_{t+1,i} \geq \max(\min(1-1/n, \gamma\lambda/\mu), 1/n) \geq \min(1-1/n, \gamma/\gamma_0)$, so

$$\begin{aligned}\Pr\left(y \in A_{\geq j+1}\right) &\geq \prod_{i=1}^{2(j+1)} p_{t+1,i} \geq \left(1-\frac{1}{n}\right)^{2j}\left(\frac{\gamma}{\gamma_0}\right)^2 \\ &\geq \frac{1}{e}\left(\frac{\gamma}{\gamma_0}\right)^2 = \frac{\lambda\gamma}{e\mu^2} \geq (1+\delta)\gamma\end{aligned}$$

due to $\gamma_0 \geq \gamma \geq 1/\lambda$, and $\lambda \geq (1+\delta)e\mu^2$ for any constant $\delta > 0$. Therefore, condition (G2) is now satisfied.

Condition (G1)

For any level $j \in \{0\} \cup [m-1]$ satisfying $|P_t \cap A_{\geq j}| \geq \gamma_0\lambda = \mu$ we seek a lower bound z_j on $\Pr\left(y \in A_{\geq j+1}\right)$. Again the condition on level j implies that $p_{t+1,i} = 1-1/n$ for all $i \in [2j]$. Due to the imposed lower margin, we can assume pessimistically that $p_{t+1,2j+1} = p_{t+1,2j+2} \geq 1/n$. Hence,

$$\Pr\left(y \in A_{\geq j+1}\right) \geq \left(1-\frac{1}{n}\right)^{2j}\left(\frac{1}{n}\right)^2 \geq \frac{1}{en^2} =: z_j,$$

and the condition (G1) is satisfied for $z_j := \Omega(1/n^2)$ and $z_* = \min_{j\in[m-1]}\{z_j\} = \Omega(1/n^2)$.

Condition (G3)

We are required to satisfy the following condition

$$\lambda \geq \left(\frac{4}{\gamma_0\delta^2}\right)\ln\left(\frac{128m}{\delta^2 z_*}\right).$$

Note that $1/z_* = \mathrm{O}(n)$ and $m = \mathrm{O}(n)$. The assumption that $\mu \geq c\log n$ for a sufficiently large constant $c > 0$ implies that

$$\lambda = \frac{\mu}{\mu/\lambda} \geq \left(\frac{c}{\gamma_0}\right)\log n \geq \left(\frac{4}{\gamma_0\delta^2}\right)\ln\left(\frac{128m}{\delta^2 z_*}\right).$$

Therefore, the condition is satisfied as long as the constant $c > 0$ is chosen large enough.

All conditions of Theorem 4 are satisfied, so the expected runtime of the UMDA on the DLB function is

$$\mathrm{O}\left(\sum_{j=1}^{n/2}\left(\lambda\ln\lambda + n^2\right)\right) = \mathrm{O}\left(n\lambda\log\lambda + n^3\right). \qquad \blacksquare$$

One might say that the failure of the UMDA to optimise the DLB problem is not necessarily coming from the pairwise deception, but from the low selection rate in the same way that it struggles on the LEADINGONES function [79]. However, as a final remark, we think that this claim is inaccurate. In fact, for the range of selection rates considered, it has been shown that the UMDA optimises other non-deceptive problems easily in polynomial expected runtime, including LEADINGONES [23, 79] and ONEMAX [23].

Non-elitist algorithms using (μ,λ)-selection are typically efficient when $\mu/\lambda < 1/e$, i.e., below the error threshold (see [75]). In contrast, to show that the UMDA optimises DLB efficiently, we need to decrease this ratio significantly so that we get $\mu/\lambda = \mathrm{O}(1/\mu)$, i.e., extremely high selection rate. We do not normally see such high selection rates in practical applications of the UMDA.

6.4 Conclusion

In this chapter, we have introduced the DLB function, in which bits are highly correlated (similar to LEADINGONES) and contains many small traps. Since this function was new, we first showed that simple EAs, namely the $(1+\lambda)$ EA, the $(\mu+1)$ EA and the non-elitist (μ,λ) EA, can optimise the function within an expected runtime of $\mathrm{O}\left(n^3\right)$. Next, we aimed at showing that due to correlation and deception, the DLB problem may be hard for the UMDA, which assumes independence between decision variables. Via the anti-concentration bound (see Theorem 8, applied in Lemma 27), we were able to show a lower bound of $2^{\Omega(\mu^k)}$ on the expected runtime of the UMDA with a parent population size $\mu = o(n/\log n)$ and an offspring population size $\mu \leq \lambda = \mathrm{O}\left(\mu^{2-k}\right) = o(\mu^2)$ for some constant $k \in (0,1]$ on the DLB function. In contrast, very large population sizes $\lambda = \Omega(\mu^2)$ will help as in this situation we showed via the level-based analysis that the UMDA will find the global optimum of the DLB function using at most $\mathrm{O}\left(n\lambda\log\lambda + n^3\right)$ function evaluations in expectation.

Chapter 7

Conclusion

Estimation of distribution algorithms (EDAs) have been successfully applied to solve many real-world optimisation problems. While the theory of EAs has been enriched significantly over the last decades, our understandings of EDAs in terms of runtime are still limited. Runtime results provide a guarantee of the performances of the algorithms for a wide range of algorithm-specific parameters. The past few years have seen some progress in this topic, showing competitive performance compared to other EAs on some simple test functions.

7.1 Summary of Contributions

The runtime of EAs depends on the relationship between the algorithmic parameters and the underlying fitness landscape structure. If we can find out something about these complex relationships, we might be better able to apply evolutionary algorithms reliably. In the case of UMDA, the algorithmic parameters are μ and λ, and the most reasonable fitness landscape structures to start considering are linearity, epistasis, and deception.

In this thesis, we started by investigating the ability of the UMDA as a hill climber. By considering the ONEMAX function, we showed that the UMDA with a parent population size $\Omega(\log n) \ni \mu \leq \sqrt{n(1-c)}$ for some constant $c \in (0,1)$ and an offspring population size $\lambda \geq 13e\mu/(1-c)$ optimises the ONEMAX function within an expected runtime of $O(n\lambda)$. For a larger population size $\mu = \Omega(\sqrt{n}\log n)$, we showed that the UMDA with an offspring population size $\lambda \geq 294(1+\delta)\mu$ for some constant $\delta > 0$ has an expected runtime of $O(\lambda\sqrt{n})$. Both results were derived via the level-based theorem. Under appropriate

parameter settings, the UMDA has in both cases an expected runtime of $O(n \log n)$ on the ONEMAX function. These results improve the previously best known upper bound of $O(n \log n \log \log n)$ for an offspring population size $\lambda = \Theta(\log n)$ in [23] by a factor of $\Theta(\log \log n)$. Recently, a lower bound of $\Omega(n \log n)$ for any population size $\lambda = \Theta(\mu)$ has been shown in [71]. Putting together, a tight bound of $\Theta(n \log n)$ is obtained, which shows that the UMDA has a similar asymptotic runtime as that of the simple $(1+1)$ EA on the ONEMAX function [40]. In other words, the UMDA optimises the ONEMAX function efficiently. This is unsurprising as the ONEMAX is a linear function and the UMDA also assumes variable independence.

We then considered objective functions with variable interactions (also called epistasis). We chose the LEADINGONES function, which has an epistasis level of $n-1$. The UMDA is known to require an expected runtime of $O\left(n\lambda \log \lambda + n^2\right)$ for an offspring population size $\lambda = \Omega(\log n)$ and a parent population size $\mu \leq \lambda/(e(1+\delta))$ for any constant $\delta > 0$ on the LEADINGONES function [23]. We showed that the UMDA optimises the LEADINGONES function within an expected runtime of $\Omega(n\lambda/\log \lambda)$ for a parent population size $\mu = \Omega(\log n)$ and an offspring population size $\lambda \geq (1+\delta)e\mu$ for any constant $\delta \in (0,1)$. To our knowledge, this is the first lower bound for the UMDA on the LEADINGONES function. We also showed that if the offspring population size $\mu \leq \lambda \leq e^{1-\varepsilon}\mu/(1+\delta)$ for some other constant $\varepsilon \in (0,1)$ and $\mu = o(n^{1/k})$ for some constant $k \in [0,1)$, the UMDA takes an exponential expected runtime of $2^{\Omega(\mu^k)}$ on the LEADINGONES function. Our analysis revealed that the value of $1/e$ might signal a phase transition in which the runtime switches between polynomial to exponential (a similar phase transition for EAs was shown in [75]). Our result revealed that the UMDA, which assumes variable independence, can optimise the LEADINGONES function with an epistasis level of $n-1$ efficiently. Although the LEADINGONES function is just one example from the set of many objective functions where variable interaction exists and further research is desired, our analysis here emphasises that the algorithm could be robust to epistasis to a certain degree.

Real-world applications of EAs often involve noisy objective functions. To understand how noise impacts the runtime, we therefore considered the noisy LeadingOnes function, where a single bit is flipped before evaluating the fitness with a constant probability $p \in (0,1)$ (also called prior noise). We showed that if the selection rate μ/λ is sufficiently high, the UMDA optimises the noisy function within an expected runtime of $O\left(n^2 + n\lambda \log \lambda\right)$. To the best of our knowledge, this is the first time that the two algorithms are rigorously studied

in a noisy environment, while the cGA is already considered in [46] under Gaussian posterior noise. Despite the simplicity of the noise model, the analysis can be viewed as the first step towards understanding the algorithms' runtime behaviours in a noisy environment. This is important because the level of noise has implications for choosing the parameter settings of the algorithms.

Finally, we studied the ability of the UMDA to cope with deception. If it is known that an objective function is linear, we do not need an EA to optimise it. Hence, real-world applications of EDAs will typically involve problems with epistasis and/or deception. It has been shown empirically in [57] that the UMDA gets stuck in optimising the TRAP-5 function. We hypothesised that the UMDA fail on this function, but it will also fail on some other problem with a milder degree of deception. To verify this, we introduced a new benchmark problem called DLB, which has an epistasis level of $n-1$ and is mildly deceptive. We first showed that simple EAs, including elitist and non-elitist variants, can optimise the DLB function within an expected runtime of $\mathrm{O}\left(n^3\right)$. We then showed that the UMDA with $\mu = o(n/\log n)$ and $\mu \leq \lambda = \mathrm{O}\left(\mu^{2-k}\right)$ for any constant $k \in (0,1]$ requires an expected runtime of $2^{\Omega(\mu^k)}$ on the function. Recall that, under a parent population size $\mu = o(n^{1/k})$ for any constant $k \in [0,1)$, the UMDA requires a runtime of $2^{\Omega(\mu^k)}$ to optimise the LEADINGONES function for an offspring population size $\mu \leq \lambda \leq e^{1-\varepsilon}\mu/(1+\delta) < 2.72\mu$ for some constants $\varepsilon, \delta \in (0,1)$. It is now clear that when the objective function is deceptive, the exponential runtime holds for a significantly extended offspring population size. Because this problem instance is the simplest in the class of DLB functions, we conjecture that when the deception degree gets increased (by increasing the width w), the exponential runtime will hold for an extensive range of the offspring population size λ. Our result confirms that the univariate EDAs due to the variable independence assumption have difficulty in coping with deceptive objective functions.

To sum up, the thesis presented a picture of the runtime behaviours of the UMDA when optimising objective functions with different characteristics. Overall, it demonstrated that while univariate EDAs may cope well with variable independence and epistasis in the environment, the algorithms could suffer even at a mild level of deception.

7.2 Future Work

In Chapter 4, we showed that the UMDA optimises the ONEMAX function within an expected runtime of $\Theta(n \log n)$ under appropriate parameter settings. Like ONEMAX, BINVAL is also a well-known test function in the class of linear functions. The two are often considered extreme variants of this class. Runtime bounds for univariate EDAs on the BINVAL function are very rare [39, 24, 128]. It remains unknown whether the UMDA could optimise the function using at most $O(n \log n)$ function evaluations. Due to the exponential scaling, Witt [128] showed that the bits with heavy weights would be optimised before those with light weights. Consequently, the UMDA could take a runtime of $\Omega(n^2)$ to optimise the BINVAL function. While an upper bound of $O(n^2)$ has been shown in [24] for an offspring population size $\lambda = \Omega(\log n) \cap O(n/\log n)$. We are still missing a lower bound for the UMDA on the BINVAL function to confirm this. If this were the case, then the UMDA would require different runtimes on the two linear functions class members. In the end, this could help to obtain runtime bounds for the univariate EDAs on the class of linear functions.

In Chapter 5, we considered the PBIL on the LEADINGONES function as unlike the UMDA the algorithm takes into account the current values of the marginals when updating the probabilistic model (also called incremental learning). Analysing the runtime of the PBIL on the LEADINGONES function on which the runtime of the UMDA is already known will shed light on whether incremental learning would bring any benefit. We obtained an expected runtime of $O(n\lambda \log \lambda + n^2)$ for the PBIL on the LEADINGONES function under population sizes $\lambda = \Omega(\log n)$ and $\lambda = \Theta(\mu)$, assuming that all marginals stay above the constant $(\mu/\lambda)/(1+\varepsilon)$ for some constant $\varepsilon \in (0,1)$ during the optimisation. Although we were not able to obtain the overall expected runtime at the time of submission, we take this as a direction for future research as there have been few runtime results for the PBIL on test functions. In the end, an expected runtime of $O(n^2)$ might be obtained for $\lambda = \Omega(\log n) \cap O(n/\log n)$, which could improve the previously best known upper bound of $O(n^{2+c})$ [133] by a significant factor of $\Theta(n^c)$ for some constant $c \in (0,1)$. Our bound could hold for any offspring population size $\lambda = \Omega(\log n)$ as opposed to large population sizes $\lambda = \Omega(n^{1+c})$ as in [133].

Furthermore, we introduced noise into the LEADINGONES function under a prior noise model. Despite the simplicity of the model, the UMDA and the PBIL were shown to handle noise well. A recent survey [35] pointed out that the runtime of EDAs in a noisy environment

is an open problem. In practice, noise is inevitable in many applications as a consequence of human/machine errors. Thus, another direction for future work could be to analyse the runtime of EDAs on posterior noise. Unlike the prior noise model which occurs before fitness evaluation, noise in the posterior model occurs during the fitness evaluations. This type of noise has already been studied in [46], in which the cGA was shown to take a runtime of $\mathrm{O}\left(\sigma^4 n \log^2 n\right)$ with high probability to optimise the OneMax function under the zero-mean Gaussian noise with a variance of $\sigma^2 > 0$.

In Chapter 6, we considered the UMDA on the DLB function, which is a modification of the LeadingOnes function to create a deceptive fitness function. Although the degree of deception is mild, the UMDA requires an extremely high selection rate $\mu/\lambda = \mathrm{O}(1/\mu)$ to optimise the function. We believe that the algorithm's trajectory is similar to that of the $(1,\lambda)$EA on the same function. To rigorously prove this, one could show that the two algorithms' trajectories match this situation. It is not entirely clear how this could be done. However, one might use measures from the information theory like the total variation distance or Kullback-Leibler divergence [72] to examine the distance between the two probability distributions implied by the algorithms at any point in time until convergence.

So far, we have discussed univariate EDAs only. This class of EDAs assume variable independence. Although we showed in Chapter 5 that the algorithms might cope well with epistasis, we still believe that the algorithms will face challenges when dealing with more complex problems. The multivariate EDAs allow an arbitrary probabilistic graphical model to be constructed. Despite the high cost of building such a model for many decision variables, the algorithms can learn higher-order variable dependencies, which is essential for successfully finding the global optima. Due to the complexity of these algorithms, we do not expect runtime bounds for multivariate EDAs like the Bayesian optimisation algorithm [110] on test functions to be proven soon. However, a runtime bound for the simplest variant in the class like the MIMIC algorithm [27] on the LeadingOnes function might be within our reach. We believe that some preliminary results towards this direction could show whether or not multivariate EDAs offer any benefit.

References

[log] Natural logarithm: Inequalities. https://functions.wolfram.com/ElementaryFunctions/Log/29/. Accessed: 2020-11-09.

[2] Ackley, D. H. (1987). An empirical study of bit vector function optimisation. *Genetic Algorithms and Simulated Annealing*, pages 170–204.

[3] Armañanzas, R., Inza, I., Santana, R., Saeys, Y., Flores, J. L., Lozano, J. A., Peer, Y. V. d., Blanco, R., Robles, V., Bielza, C., and Larrañaga, P. (2008). A review of estimation of distribution algorithms in bioinformatics. *BioData Mining*, 1:6.

[4] Asoh, H. and Mühlenbein, H. (1994). On the mean convergence time of evolutionary algorithms without selection and mutation. In *Proceedings of the Conference on Parallel Problem Solving from Nature*, PPSN '94, pages 88–97.

[5] Baillon, J.-B., Cominetti, R., and Vaisman, J. (2016). A sharp uniform bound for the distribution of sums of Bernoulli trials. *Combinatorics, Probability and Computing*, 25(3):352–361.

[6] Baluja, S. (1994). Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning. Technical Report CMU-CS-94-163, Carnegie Mellon University.

[7] Baluja, S. and Davies, S. (1997). Using optimal dependency-trees for combinatorial optimization: Learning the structure of the search space. Technical Report CMU-CS-97-107, Carnegie Mellon University.

[8] Baritompa, B. and Steel, M. (1996). Bounds on absorption times of directionally biased random sequences. *Random Structures & Algorithms*, 9(3):279–293.

[9] Bishop, C. M. (2013). Model-based machine learning. *Philosophical transactions. Series A, Mathematical, physical, and engineering sciences*, 371(1984).

[10] Bonyadi, M. R. and Michalewicz, Z. (2016). Evolutionary computation for real-world problems. In Matwin, S. and Mielniczuk, J., editors, *Challenges in Computational Statistics and Data Mining*, pages 1–24. Springer International Publishing.

[11] Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press.

[12] Braun, H. (1991). On solving travelling salesman problems by genetic algorithms. In *Proceedings of the Conference on Parallel Problem Solving from Nature*, PPSN '91, pages 129–133.

[13] Brochu, E., Cora, V. M., and de Freitas, N. (2010). A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *CoRR*, abs/1012.2599.

[14] Chen, T., Lehre, P. K., Tang, K., and Yao, X. (2009a). When is an estimation of distribution algorithm better than an evolutionary algorithm? In *Proceedings of the IEEE Congress on Evolutionary Computation*, CEC '09, pages 1470–1477.

[15] Chen, T., Tang, K., Chen, G., and Yao, X. (2007). On the analysis of average time complexity of estimation of distribution algorithms. In *Proceedings of the IEEE Congress on Evolutionary Computation*, CEC '07, pages 453–460.

[16] Chen, T., Tang, K., Chen, G., and Yao, X. (2009b). Rigorous time complexity analysis of univariate marginal distribution algorithm with margins. In *Proceedings of the IEEE Congress on Evolutionary Computation*, CEC '09, pages 2157–2164.

[17] Chen, T., Tang, K., Chen, G., and Yao, X. (2010). Analysis of computational time of simple estimation of distribution algorithms. *IEEE Transactions on Evolutionary Computation*, 14(1):1–22.

[18] Chen, Y., Meng, G., Zhang, Q., Xiang, S., Huang, C., Mu, L., and Wang, X. (2019). Reinforced evolutionary neural architecture search. *CoRR*, abs/1808.00193.

[19] Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2009). *Introduction to Algorithms*. MIT Press, 3rd edition.

[20] Corus, D., Dang, D., Eremeev, A. V., and Lehre, P. K. (2018). Level-based analysis of genetic algorithms and other search processes. *IEEE Transactions on Evolutionary Computation*, 22(5):707–719.

[21] Cui, X., Zhang, W., Tüske, Z., and Picheny, M. (2018). Evolutionary stochastic gradient descent for optimization of deep neural networks. In *Proceedings of the Conference on Neural Information Processing Systems*, NIPS '18, page 6051–6061.

[22] Dang, D. and Lehre, P. K. (2015a). Efficient optimisation of noisy fitness functions with population-based evolutionary algorithms. In *Proceedings of the Conference on Foundations of Genetic Algorithms*, FOGA '15, pages 62–68.

[23] Dang, D. and Lehre, P. K. (2015b). Simplified runtime analysis of estimation of distribution algorithms. In *Proceedings of the Conference on Genetic and Evolutionary Computation Conference*, GECCO '15, pages 513–518.

[24] Dang, D., Lehre, P. K., and Nguyen, P. T. H. (2019). Level-based analysis of the univariate marginal distribution algorithm. *Algorithmica*, 81(2):668–702.

[25] Davidor, Y. (1991). Epistasis variance: A viewpoint on GA-Hardness. In *Proceedings of the Conference on Foundations of Genetic Algorithms*, FOGA '91, pages 23–35.

[26] de Boer, P.-T., Kroese, D. P., Mannor, S., and Rubinstein, R. Y. (2005). A tutorial on the cross-entropy method. *Annals of Operations Research*, 134(1):19–67.

[27] De Bonet, J. S., Isbell Jr, C. L., and Viola, P. (1996). MIMIC: Finding optima by estimating probability densities. In *Proceedings of the Conference on Neural Information Processing Systems*, NIPS '96, pages 424–430.

[28] Deuflhard, P. (2011). *Newton Methods for Nonlinear Problems: Affine Invariance and Adaptive Algorithms*. Springer Publishing Company, Incorporated.

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

[30] Doerr, B. (2020). Probabilistic tools for the analysis of randomized optimization heuristics. In *Theory of Evolutionary Computation: Recent Developments in Discrete Optimization*, pages 1–87. Springer International Publishing.

[31] Doerr, B., Johannsen, D., and Winzen, C. (2012). Multiplicative drift analysis. *Algorithmica*, 64:673–697.

[32] Doerr, B. and Krejca, M. S. (2018). Significance-based estimation-of-distribution algorithms. In *Proceedings of the Conference on Genetic and Evolutionary Computation*, GECCO '18, pages 1483–1490.

[33] Doerr, B. and Krejca, M. S. (2020). The univariate marginal distribution algorithm copes well with deception and epistasis. In *Evolutionary Computation in Combinatorial Optimization*, pages 51–66.

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

[35] Doerr, B. and Neumann, F. (2020). A survey on recent progress in the theory of evolutionary algorithms for discrete optimization. *CoRR*, abs/2006.16709.

[36] Doerr, B. and Zheng, W. (2020). Sharp bounds for genetic drift in estimation of distribution algorithms. *IEEE Transactions on Evolutionary Computation*, 24(6):1140–1149.

[37] Doerr, B. and Zheng, W. (2020). Sharp bounds for genetic drift in estimation of distribution algorithms. *Trans. Evol. Comp*, 24(6):1140–1149.

[38] Dostál, M. (2013). Evolutionary music composition. In Zelinka, I., Snášel, V., and Abraham, A., editors, *Handbook of Optimization: From Classical to Modern Approach*, pages 935–964. Springer Berlin Heidelberg.

[39] Droste, S. (2006). A rigorous analysis of the compact genetic algorithm for linear functions. *Natural Computing*, 5(3):257–283.

[40] Droste, S., Jansen, T., and Wegener, I. (2002). On the analysis of the (1+ 1) evolutionary algorithm. *Theoretical Computer Science*, 276(1-2):51–81.

[41] Eiben, A. E. and Smith, J. E. (2003). *Introduction to Evolutionary Computing*. SpringerVerlag.

[42] Feige, U. (2006). On sums of independent random variables with unbounded variance and estimating the average degree in a graph. *SIAM Journal of Computing*, 35(4):964–984.

[43] Feller, W. (1968). *An introduction to probability theory and its applications*, volume 1. John Wiley & Sons, Inc., 3 edition.

[44] Feller, W. (2008). *An introduction to probability theory and its application*. Wiley, 2nd edition.

[45] Forrest, S. and Mitchell, M. (1993). What makes a problem hard for a genetic algorithm? some anomalous results and their explanation. *Machine Learning*, 13(2):285–319.

[46] Friedrich, T., Kötzing, T., Krejca, M., and Sutton, A. M. (2017). The compact genetic algorithm is efficient under extreme Gaussian noise. *IEEE Transactions on Evolutionary Computation*, 21(3):477–490.

[47] Friedrich, T., Kötzing, T., and Krejca, M. S. (2016). EDAs cannot be balanced and stable. In *Proceedings of the Conference on Genetic and Evolutionary Computation*, GECCO '16, pages 1139–1146.

[48] Gießen, C. and Kötzing, T. (2016). Robustness of populations in stochastic environments. *Algorithmica*, 75(3):462–489.

[49] Gleser, L. J. (1975). On the distribution of the number of successes in independent trials. *Annals of Probability*, 3(1):182–188.

[50] Goldberg, D. E., Deb, K., and Clark, J. H. (1991). Genetic algorithms, noise, and the sizing of populations. *Urbana*, 51:61801.

[51] Goldberg, D. E., Sastry, K., and Latoza, T. (2001). On the supply of building blocks. In *Proceedings of the Conference on Genetic and Evolutionary Computation*, GECCO '01, pages 336–342.

[52] Gras, R. (2008). How efficient are genetic algorithms to solve high epistasis deceptive problems? In *Proceedings of the IEEE Congress on Evolutionary Computation*, CEC '08, pages 242–249.

[53] Gutmann, H. M. (2001). A radial basis function method for global optimization. *Journal of Global Optimization*, 19:201–227.

[54] Hajek, B. (1982). Hitting-time and occupation-time bounds implied by drift analysis with applications. *Advances in Applied Probability*, 14(3):502–525.

[55] Harik, G. (1999). Linkage learning via probabilistic modeling in the ECGA. Technical Report 99010, University of Illinois at Urbana-Champaign.

[56] Harik, G. R., Lobo, F. G., and Goldberg, D. E. (1997). The compact genetic algorithm. Technical Report 97006, University of Illinois at Urbana-Champaign.

[57] Hauschild, M. and Pelikan, M. (2011). An introduction and survey of estimation of distribution algorithms. *Swarm and Evolutionary Computation*, 1(3):111–128.

[58] He, J. and Yao, X. (2003). Towards an analytic framework for analysing the computation time of evolutionary algorithms. *Artificial Intelligence*, 145(1):59–97.

[59] Holland, J. H. (1992). *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press.

[60] Holmström, K. (2008). An adaptive radial basis algorithm (ARBF) for expensive black-box global optimization. *Journal of Global Optimization*, 41:447–464.

[61] Ishibuchi, H., Akedo, N., and Nojima, Y. (2015). Behavior of multiobjective evolutionary algorithms on many-objective knapsack problems. *IEEE Transactions on Evolutionary Computation*, 19(2):264–283.

[62] Jaderberg, M., Dalibard, V., Osindero, S., Czarnecki, W. M., Donahue, J., Razavi, A., Vinyals, O., Green, T., Dunning, I., Simonyan, K., Fernando, C., and Kavukcuoglu, K. (2017). Population based training of neural networks. *CoRR*, abs/1711.09846.

[63] Jansen, T., De Jong, K. A., and Wegener, I. (2005). On the choice of the offspring population size in evolutionary algorithms. *Evolutionary Computation*, 13(4):413–440.

[64] Jansen, T. and Wiegand, P. R. (2004). The cooperative coevolutionary (1+1) EA. *Evolutionary Computation*, 12(4):405–434.

[65] Jogdeo, K. and Samuels, S. M. (1968). Monotone convergence of binomial probabilities and a generalization of Ramanujan's equation. *The Annals of Mathematical Statistics*, 39(4):1191–1195.

[66] Johannsen, D. (2010). *Random Combinatorial Structures and Randomized Search Heuristics*. PhD thesis, Universität des Saarlandes.

[67] Jones, D. R., Schonlau, M., and Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13:455–492.

[68] Jordan, M. I. (2004). Graphical models. *Statistical Science*, 19(1):140–155.

[69] Korb, K. B. and Nicholson, A. E. (2010). *Bayesian Artificial Intelligence, Second Edition*. CRC Press, Inc., USA, 2nd edition.

[70] Krejca, M. S. (2019). *Theoretical analyses of univariate estimation-of-distribution algorithms*. PhD thesis, Universität Potsdam.

[71] Krejca, M. S. and Witt, C. (2017). Lower bounds on the run time of the univariate marginal distribution algorithm on OneMax. In *Proceedings of the Conference on Foundation of Genetic Algorithms*, FOGA '17, pages 65–79.

[72] Kullback, S. and Leibler, R. A. (1951). On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86.

[73] Larrañaga, P., Karshenas, H., Bielza, C., and Santana, R. (2012). A review on probabilistic graphical models in evolutionary computation. *Journal of Heuristics*, 18(5):795–819.

[74] Lässig, J. and Sudholt, D. (2014). Analysis of speedups in parallel evolutionary algorithms and (1+λ) EAs for combinatorial optimization. *Theoretical Computer Science*, 551:66–83.

[75] Lehre, P. K. (2010). Negative drift in populations. In *Proceedings of the Conference on Parallel Problem Solving from Nature*, PPSN '10, pages 244–253.

[76] Lehre, P. K. and Nguyen, P. T. H. (2017). Improved runtime bounds for the univariate marginal distribution algorithm via anti-concentration. In *Proceedings of the Conference on Genetic and Evolutionary Computation*, GECCO '17, pages 1383–1390.

[77] Lehre, P. K. and Nguyen, P. T. H. (2018). Level-based analysis of the population-based incremental learning algorithm. In *Proceedings of the Conference on Parallel Problem Solving from Nature*, PPSN '18, pages 105–116.

[78] Lehre, P. K. and Nguyen, P. T. H. (2019a). On the limitations of the univariate marginal distribution algorithm to deception and where bivariate EDAs might help. In *Proceedings of the Conference on Foundations of Genetic Algorithms*, FOGA '19, page 154–168.

[79] Lehre, P. K. and Nguyen, P. T. H. (2019b). Runtime analysis of the univariate marginal distribution algorithm under low selective pressure and prior noise. In *Proceedings of the Conference on Genetic and Evolutionary Computation*, GECCO '19, pages 1497–1505.

[80] Lehre, P. K. and Oliveto, P. S. (2018). Theoretical analysis of stochastic search algorithms. In *Handbook of Heuristics*, pages 1–36. Springer International Publishing.

[81] Lehre, P. K. and Sudholt, D. (2020). Parallel black-box complexity with tail bounds. *IEEE Transactions on Evolutionary Computation*, 24(6):1010–1024.

[82] Lehre, P. K. and Witt, C. (2012). Black-Box Search by Unbiased Variation. *Algorithmica*, 64(4):623–642.

[83] Lehre, P. K. and Witt, C. (2020). Tail bounds on hitting times of randomized search heuristics using variable drift analysis. *Combinatorics, Probability and Computing*, page 1–20.

[84] Lemaréchal, C. (2012). Cauchy and the gradient method. https://www.math.uni-bielefeld.de/documenta/vol-ismp/40_lemarechal-claude.pdf. Accessed: 2020-11-19.

[85] Lengler, J. (2020). Drift analysis. In Doerr, B. and Neumann, F., editors, *Theory of Evolutionary Computation: Recent Developments in Discrete Optimization*, pages 89–131. Springer International Publishing.

[86] Lengler, J. and Steger, A. (2018). Drift analysis and evolutionary algorithms revisited. *Combinatorics, Probability and Computing*, 27(4):643–666.

[87] Lengler, J., Sudholt, D., and Witt, C. (2018). Medium step sizes are harmful for the compact genetic algorithm. In *Proceedings of the Conference on Genetic and Evolutionary Computation*, GECCO '18, pages 1499–1506.

[88] Lewis, M. (2008). Evolutionary visual art and design. In Romero, J. and Machado, P., editors, *The Art of Artificial Evolution: A Handbook on Evolutionary Art and Music*, pages 3–37. Springer Berlin Heidelberg.

[89] Li, A., Spyra, O., Perel, S., Dalibard, V., Jaderberg, M., Gu, C., Budden, D., Harley, T., and Gupta, P. (2019). A generalized framework for population based training. In *Proceedings of the Conference on Knowledge Discovery & Data Mining*, KDD '19, page 1791–1799.

[90] Marshall, A. W., Olkin, I., and Arnold, B. C. (2011). *Inequalities: Theory of Majorization and Its Applications*. Springer Series in Statistics. Springer-Verlag New York.

[91] Massart, P. (1990). The tight constant in the Dvoretzky-Kiefer-Wolfowitz inequality. *The Annals of Probability*, 18(3):1269–1283.

[92] Mitavskiy, B., Rowe, J., and Cannings, C. (2009). 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):243–284.

[93] Mitchell, M., Holland, J. H., and Forrest, S. (1992). The royal road for genetic algorithms: Fitness landscapes and GA performance. In *Proceedings of the First European Conference on Artificial Life*.

[94] Mitrinović, D. S. (1970). *Analytic inequalities*. Springer-Verlag.

[95] Mitzenmacher, M. and Upfal, E. (2005). *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press.

[96] Motwani, R. and Raghavan, P. (1995). *Randomised algorithms*. Cambridge University Press.

[97] Mühlenbein, H. (1992). How genetic algorithms really work i. mutation and hill climbing. In *Proceedings of the Conference on Parallel Problem Solving from Nature*, PPSN '92, pages 15–26.

[98] Mühlenbein, H. and Mahnig, T. (1999). Convergence theory and applications of the factorized distribution algorithm. *CIT. Journal of computing and information technology*, 7(1):19–32.

[99] Mühlenbein, H. and Paaß, G. (1996). From recombination of genes to the estimation of distributions i. binary parameters. In *Proceedings of the Conference on Parallel Problem Solving from Nature*, PPSN '96, pages 178–187.

[100] Mühlenbein, H. and Schlierkamp-Voosen, D. (1993). The science of breeding and its application to the breeder genetic algorithm (BGA). *Evolutionary computation*, 1(4):335–360.

[101] Murty, K. G. (1983). *Linear programming*. John Wiley & Sons.

[102] Nemirovsky, A. and Yudin, D. (1983). Problem complexity and method efficiency in optimization. *SIAM Review*, 27(2):264–265.

[103] Neshat, M., Alexander, B., and Wagner, M. (2020). A hybrid cooperative co-evolution algorithm framework for optimising power take off and placements of wave energy converters. *Information Sciences*, 534:218–244.

[104] Neumann, A., Alexander, B., and Neumann, F. (2020). Evolutionary image transition and painting using random walks. *CoRR*, abs/2003.01517.

[105] Neumann, F. and Wegener, I. (2007). Randomized local search, evolutionary algorithms, and the minimum spanning tree problem. *Theoretical Computer Science*, 378(1):32–40.

[106] Neumann, F. and Witt, C. (2010a). Analyzing stochastic search algorithms. In *Bioinspired Computation in Combinatorial Optimisation: Algorithms and Their Computational Complexity*, pages 33–48. Springer Berlin Heidelberg.

[107] Neumann, F. and Witt, C. (2010b). Stochastic search algorithms. In *Bioinspired Computation in Combinatorial Optimization: Algorithms and Their Computational Complexity*, pages 21–32. Springer Berlin Heidelberg.

[108] Oliveto, P. S. and Yao, X. (2011). Runtime analysis of evolutionary algorithms for discrete optimization. In *Theory of Randomized Search Heuristics*, pages 21–52. World Scientific.

[109] Pearl, J. (1988). Chapter 3-markov and bayesian networks: Two graphical representations of probabilistic knowledge. In Pearl, J., editor, *Probabilistic Reasoning in Intelligent Systems*, pages 77–141. Morgan Kaufmann.

[110] Pelikan, M., Goldberg, D. E., and Cantú-Paz, E. (1999). BOA: The bayesian optimization algorithm. In *Proceedings of the Conference on Genetic and Evolutionary Computation*, GECCO '99, pages 525–532.

[111] Pelikan, M. and Muehlenbein, H. (1999). The bivariate marginal distribution algorithm. *Advances in Soft Computing*, pages 521–535.

[112] Rechenberg, I. (1973). Evolutionsstrategie. stuttgart: Holzmann-froboog. Technical report, ISBN 3-7728-0373-3.

[113] Reeves, C. R. (2014). Fitness landscapes. In Burke, E. K. and Kendall, G., editors, *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*, pages 681–705. Springer US.

[114] Schwefel, H.-P. (1977). *Numerische optimierung von computer-modellen mittels der evolutionsstrategie*, volume 1. Birkhäuser, Basel Switzerland.

[115] Shakya, S. K. (2006). *DEUM: a framework for an estimation of distribution algorithm based on Markov random fields.* PhD thesis, Robert Gordon University.

[116] Shapiro, J. L. (2005). Drift and scaling in estimation of distribution algorithms. *Evolutionary Computation*, 13(1):99–123.

[117] Slowik, A. and Kwasnicka, H. (2020). Evolutionary algorithms and their applications to engineering problems. *Neural Computing and Applications*, 32:12363–12379.

[118] Sudholt, D. (2013). A new method for lower bounds on the running time of evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 17(3):418–435.

[119] Sudholt, D. and Witt, C. (2016). Update strength in EDAs and ACO: How to avoid genetic drift. In *Proceedings of the Conference on Genetic and Evolutionary Computation*, GECCO '16, pages 61–68.

[120] Suykens, J. A. K. (2001). Nonlinear modelling and support vector machines. In *Proceedings of the IEEE Conference on Instrumentation and Measurement Technology. Rediscovering Measurement in the Age of Informatics (Cat. No.01CH 37188)*, volume 1 of *IMTC '01*, pages 287–294.

[121] Tran, R., Wu, J., Denison, C., Ackling, T., Wagner, M., and Neumann, F. (2013). Fast and effective multi-objective optimisation of wind turbine placement. In *Proceedings of the Conference on Genetic and Evolutionary Computation*, GECCO '13, page 1381–1388.

[122] Wegener, I. (2002). Methods for the analysis of evolutionary algorithms on pseudo-boolean functions. In *Evolutionary Optimization*, pages 349–369. Springer US.

[Weisstein] Weisstein, E. W. Binomial distribution. https://mathworld.wolfram.com/BinomialDistribution.html. Accessed: 2020-11-12.

[124] Whitley, L. D. (1991). Fundamental principles of deception in genetic search. In *Proceedings of the Conference on Foundations of Genetic Algorithms*, FOGA '91, pages 221–241.

[125] Williams, D. (1991). *Probability with Martingales*. Cambridge University Press.

[126] Witt, C. (2006). Runtime analysis of the (μ+1) EA on simple pseudo-boolean functions. *Evolutionary Computation*, 14(1):65–86.

[127] Witt, C. (2017). Upper bounds on the runtime of the univariate marginal distribution algorithm on OneMax. In *Proceedings of the Conference on Genetic and Evolutionary Computation*, GECCO '17, pages 1415–1422.

[128] Witt, C. (2018a). Domino convergence: why one should hill-climb on linear functions. In *Proceedings of the Conference on Genetic and Evolutionary Computation*, GECCO '18, pages 1539–1546.

[129] Witt, C. (2018b). Theory of estimation-of-distribution algorithms. In *Proceedings of the Conference Companion on Genetic and Evolutionary Computation*, GECCO '18, pages 1170–1197.

[130] Witt, C. (2019). Upper bounds on the running time of the univariate marginal distribution algorithm on OneMax. *Algorithmica*, 81(2):632–667.

[131] Wolpert, D. and Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82.

[132] Wright, S. (1932). The roles of mutation, inbreeding, crossbreeding, and selection in evolution. In *Proceedings of the Sixth International Congress on Genetics*, pages 356–366.

[133] Wu, Z., Kolonko, M., and Möhring, R. H. (2017). Stochastic runtime analysis of the cross-entropy algorithm. *IEEE Transactions on Evolutionary Computation*, 21(4):616–628.

[134] Zheng, W., Yang, G., and Doerr, B. (2018). Working principles of binary differential evolution. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '18, pages 1103–1110.