Level-based analysis of the univariate marginal distribution algorithm

Dang, Duc-Cuong ; Lehre, Per Kristian; Nguyen, Phan Trung Hai

DOI:
10.1007/s00453-018-0507-5

Citation for published version (Harvard):
Dang, D-C, Lehre, PK & Nguyen, PTH 2019, 'Level-based analysis of the univariate marginal distribution algorithm', *Algorithmica*, vol. 81, no. 2, pp. 668-702. https://doi.org/10.1007/s00453-018-0507-5

Algorithmica
https://doi.org/10.1007/s00453-018-0507-5

Level-Based Analysis of the Univariate Marginal Distribution Algorithm

Duc-Cuong Dang[1] · Per Kristian Lehre[2] · Phan Trung Hai Nguyen[2]

Received: 11 September 2017 / Accepted: 16 August 2018

Abstract

Estimation of Distribution Algorithms (EDAs) are stochastic heuristics that search for optimal solutions by learning and sampling from probabilistic models. Despite their popularity in real-world applications, there is little rigorous understanding of their performance. Even for the Univariate Marginal Distribution Algorithm (UMDA)—a simple population-based EDA assuming independence between decision variables—the optimisation time on the linear problem OneMax was until recently undetermined. The incomplete theoretical understanding of EDAs is mainly due to the lack of appropriate analytical tools. We show that the recently developed *level-based theorem* for non-elitist populations combined with anti-concentration results yield upper bounds on the expected optimisation time of the UMDA. This approach results in the bound $\mathcal{O}\left(n\lambda \log \lambda + n^2\right)$ on the LeadingOnes and BinVal problems for population sizes $\lambda > \mu = \Omega(\log n)$, where μ and λ are parameters of the algorithm. We also prove that the UMDA with population sizes $\mu \in \mathcal{O}\left(\sqrt{n}\right) \cap \Omega(\log n)$ optimises OneMax in expected time $\mathcal{O}(\lambda n)$, and for larger population sizes $\mu = \Omega(\sqrt{n} \log n)$, in expected time $\mathcal{O}\left(\lambda \sqrt{n}\right)$. The facility and generality of our arguments suggest that this is a promising approach to derive bounds on the expected optimisation time of EDAs.

Keywords Estimation of distribution algorithms · Runtime analysis · Level-based analysis · Anti-concentration

Preliminary versions of this work appeared in the Proceedings of the 2015 and 2017 Genetic and Evolutionary Computation Conference (GECCO 2015 and 2017).

✉ Per Kristian Lehre
p.k.lehre@cs.bham.ac.uk

Duc-Cuong Dang
duc-cuong.dang@hds.utc.fr

Phan Trung Hai Nguyen
p.nguyen@cs.bham.ac.uk

[1] Hanoi, Vietnam

[2] School of Computer Science, University of Birmingham, Birmingham B15 2TT, UK

Published online: 08 October 2018

1 Introduction

Estimation of Distribution Algorithms (EDAs) are a class of randomised search heuristics with many practical applications [15,20,24,47,48]. Unlike traditional Evolutionary Algorithms (EAs) which search for optimal solutions using genetic operators such as mutation or crossover, EDAs build and maintain a probability distribution of the current population over the search space, from which the next generation of individuals is sampled. Several EDAs have been developed over the last decades. The algorithms differ in how they capture interactions among decision variables, as well as in how they build and update their probabilistic models. EDAs are often classified as either *univariate* or *multivariate*; the former treats each variable independently, while the latter also considers variable dependencies [40]. Well-known univariate EDAs include the compact Genetic Algorithm (cGA [21]), the Population-Based Incremental Learning Algorithm (PBIL [4]), and the Univariate Marginal Distribution Algorithm (UMDA [37]). Given a problem instance of size n, univariate EDAs represent probabilistic models as an n-vector, where each vector component is called a *marginal*. Some Ant Colony Optimisation (ACO) algorithms and even certain single-individual EAs can be cast in the same framework as univariate EDAs (or n-Bernoulli-λ-EDA, see, e.g., [18,22,25,42]). Multivariate EDAs, such as the Bayesian Optimisation Algorithm, which builds a Bayesian network with nodes and edges representing variables and conditional dependencies respectively, attempt to learn relationships between decision variables [22]. The surveys [1,22,39] describe further variants and applications of EDAs.

Recently EDAs have drawn a growing attention from the theory community of evolutionary computation [10,13,18,26–28,32,44–46]. The aim of the theoretical analyses of EDAs in general is to gain insights into the behaviour of the algorithms when optimising an objective function, especially in terms of the optimisation time, that is the number of function evaluations, required by the algorithm until an optimal solution has been found for the first time. Droste [14] provided the first rigorous runtime analysis of an EDA, specifically the cGA. Introduced in [21], the cGA samples two individuals in each generation and updates the probabilistic model according to the fittest of these individuals. A quantity of $\pm 1/K$ is added to the marginals for each bit position where the two individuals differ. The reciprocal K of this quantity is often referred to as the abstract *population size* of a genetic algorithm that the cGA is supposed to model. Droste showed a lower bound $\Omega(K\sqrt{n})$ on the expected optimisation time of the cGA for any pseudo-Boolean function [14]. He also proved the upper bound $\mathcal{O}(nK)$ for any linear function, where $K = n^{1/2+\varepsilon}$ for any small constant $\varepsilon > 0$. Note that each marginal of the cGA considered in [14] is allowed to reach the extreme values zero and one. Such an algorithm is referred to as an EDA *without margins*, since in contrast it is possible to reinforce some margins (also called *borders*) on the range of values for each marginal to keep it away from the extreme probabilities, often within the interval $[1/n, 1-1/n]$. An EDA without margins can prematurely converge to suboptimal solutions; thus, the runtime bounds of [14] were in fact conditioned on the event that early convergence never happens. Very recently, Witt [45] studied an effect called *domino convergence* on EDAs, where bits with heavy weights tend to be optimised before bits with light weights. By deriving a lower bound of $\Omega(n^2)$ on the expected optimisation time of the cGA on BinVal for any value of $K > 0$, Witt confirmed the claim made earlier by Droste [14] that BinVal is a harder problem for the cGA than the OneMax problem is. Moreover, Lengler et al. [32] considered $K = \mathcal{O}\left(\sqrt{n}/\log^2 n\right)$, which was not covered by Droste in [14], and obtained a lower bound $\Omega(K^{1/3}n + n\log n)$ on the expected optimisation time of the cGA on OneMax. Note that if $K = \Theta(\sqrt{n}/\log^2 n)$, the above lower bound will be $\Omega(n^{7/6}/\log^2 n)$, which further tightens the bounds on the expected optimisation time of the cGA.

An algorithm closely related to the cGA with (reinforced) margins is the 2-Max Min Ant System with iteration best (2-MMAS$_{\text{ib}}$). The two algorithms differ only in the update procedure of the model, and 2-MMAS$_{\text{ib}}$ is parameterised by an evaporation factor $\rho \in (0, 1)$. Sudholt and Witt [42] proved the lower bounds $\Omega(K\sqrt{n} + n\log n)$ and $\Omega(\sqrt{n}/\rho + n\log n)$ for the two algorithms on OneMax under any setting, and upper bounds $\mathcal{O}(K\sqrt{n})$ and $\mathcal{O}(\sqrt{n}/\rho)$ when K and ρ are in $\Omega(\sqrt{n}\log n)$. Thus, the optimal expected optimisation time $\Theta(n\log n)$ of the cGA and the 2-MMAS$_{\text{ib}}$ on OneMax is achieved by setting these parameters to $\Theta(\sqrt{n}\log n)$. The analyses revealed that choosing lower parameter values results in strong fluctuations that may cause many marginals (or *pheromones* in the context of ACO) to fix early at the lower margin, which then need to be repaired later. On the other hand, choosing higher parameter values resolves the issue but may slow down the learning process.

Friedrich et al. [18] 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 uncommitted under a biasless fitness function. During the optimisation of LeadingOnes, when some bit positions are temporarily neutral, while the others are not, both properties appear useful to avoid commitment to wrong decisions. Unfortunately, many univariate EDAs without margins, including the cGA, the UMDA, the PBIL and some related algorithms are balanced but not stable [18]. A more stable version of the cGA—the so-called stable cGA (or scGA)—was then introduced in [18]. Under appropriate settings, it yields an expected optimisation time $\mathcal{O}(n\log n)$ on LeadingOnes with high probability. Furthermore, a recent study by Friedrich et al. [17] showed that cGA can cope with higher levels of noise more efficiently than mutation-only heuristics do.

Introduced by Baluja [4], the PBIL is another univariate EDA. Unlike the cGA that samples two solutions in each generation, the PBIL samples a population of λ individuals, from which the μ fittest individuals are selected to update the probabilistic model using a convex combination with a *smoothing parameter* $\rho \in (0, 1]$ of the current model and the frequencies of ones among all selected individuals at that bit position. The PBIL can be seen as a special case of the *cross-entropy method* [38] on the binary hypercube $\{0, 1\}^n$. Wu et al. [46] analysed the runtime of the PBIL on OneMax and LeadingOnes. The authors argued that due to the use of a sufficiently large population size, it is possible to prevent the marginals from reaching the lower border early even when a large smoothing parameter ρ is used. Runtime results were proved for the PBIL without margins on OneMax and the PBIL with margins on LeadingOnes, and were then compared to the runtime of some Ant System approaches. However, the required population size is large, i.e. $\lambda = \omega(n)$. Very recently, Lehre and Nguyen [28] obtained an upper bound $\mathcal{O}(n\lambda \log \lambda + n^2)$ on the expected optimisation time for the PBIL with margins on BinVal and LeadingOnes, which improves the previously known upper bound $\mathcal{O}(n^{2+\epsilon})$ in [46] by a factor of n^ε, where ε is some positive constant, for smaller population sizes $\lambda = \Omega(\log n)$.

The UMDA is a special case of the PBIL with the largest smoothing parameter $\rho = 1$, that is, the probabilistic model for the next generation depends solely on the selected individuals in the current population. The algorithm has a wide range of applications, not only in computer science, but also in other areas like population genetics and bioinformatics [20,48]. Moreover, the UMDA relates to the notion of *linkage equilibrium* [36,41], which is a popular model assumption in population genetics. Thus, studies of the UMDA can contribute to the understanding of population dynamics in population genetics.

Despite an increasing momentum in the runtime analysis of EDAs over the last few years, our understanding of the UMDA in terms of runtime is still limited. The algorithm was early analysed in a series of papers [5–8], where time-complexities of the UMDA on simple uni-modal functions were derived. These results showed that the UMDA with margins often outperforms the UMDA without margins, especially on functions like BVLeadingOnes, which is a uni-modal problem. The possible reason behind the failure of the UMDA without margins is due to fixation, causing no further progression for the corresponding decision variables. The UMDA with margins is able to avoid this by ensuring that each search point always has a positive chance to be sampled. Shapiro investigated the UMDA with a different selection mechanism than truncation selection [40]. In particular, this variant of the UMDA selects individuals whose fitnesses are no less than the mean fitness of all individuals in the current population when updating the probabilistic model. By representing the UMDA as a Markov chain, the paper showed that the population size has to be at least $\sqrt{n}$ for the UMDA to prevent the probabilistic model from quickly converging to the corners of the hypercube on the search space. This phenomenon is well-known as *genetic drift* [2]. A decade later, the first upper bound on the expected optimisation time of the UMDA on OneMax was revealed [10]. Working on the standard UMDA using truncation selection, Dang and Lehre [10] proved an upper bound $\mathcal{O}(n\lambda \log \lambda)$ on the expected optimisation time of the UMDA on OneMax, assuming a population size $\lambda = \Omega(\log n)$. If $\lambda = \Theta(\log n)$, then the upper bound is $\mathcal{O}(n \log n \log \log n)$. Inspired by the previous work of [42] on cGA/2-MMAS$_{\text{ib}}$, Krejca and Witt [26] obtained a lower bound $\Omega(\mu\sqrt{n} + n \log n)$ for the UMDA on OneMax via *drift analysis*, where $\lambda = (1 + \Theta(1))\mu$. Compared to [42], the analysis is much more involved since, unlike in cGA/2-MMAS$_{\text{ib}}$ where each change of marginals between consecutive generations is small and limited by to the smoothing parameter, large changes are always possible in the UMDA. From these results, we observe that the latest upper and lower bounds for the UMDA on OneMax still differ by $\Theta(\log \log n)$. This raises the question of whether this gap could be closed.

This paper derives upper bounds on the expected optimisation time of the UMDA on the following problems: OneMax, BinVal, and LeadingOnes. The preliminary versions of this work appeared in [10] and [27]. Here we use the improved version of the *level-based analysis* technique [9]. The analyses for LeadingOnes and BinVal are straightforward and similar to each other, i.e. yielding the same runtime

$\mathcal{O}(n\lambda\log\lambda + n^2)$; hence, they will serve the purpose of introducing the technique in the context of EDAs. Particularly, we only require population sizes $\lambda = \Omega(\log n)$ for LeadingOnes which is much smaller than previously thought [6–8]. For OneMax, we give a more detailed analysis so that an expected optimisation time $\mathcal{O}(n\log n)$ is derived if the population size is chosen appropriately. This significantly improves the results in [9,10] and matches the recent lower bound in [26]. More specifically, we assume $\lambda \geq b\mu$ for a sufficiently large constant $b > 0$, and separate two regimes of small and large selected populations: the upper bound $\mathcal{O}(\lambda n)$ is derived for $\mu = \Omega(\log n) \cap \mathcal{O}(\sqrt{n})$, and the upper bound $\mathcal{O}(\lambda\sqrt{n})$ is shown for $\mu = \Omega(\sqrt{n}\log n)$. These results exhibit the applicability of the level-based technique in the runtime analysis of (univariate) EDAs. Table 1 summarises the latest results about the runtime analyses of univariate EDAs on simple benchmark problems; see [25] for a recent survey on the theory of EDAs.

Related independent work Witt [44] independently obtained the upper bounds $\mathcal{O}(\lambda n)$ and $\mathcal{O}(\lambda\sqrt{n})$ on the expected optimisation time 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 $\mu = \Omega(\sqrt{n}) \cap \mathcal{O}\left(\sqrt{n}\log n\right)$, 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 [44].

This paper is structured as follows. Section 2 introduces the notation used throughout the paper and the UMDA with margins. We also introduce the techniques used, including the level-based theorem, which is central in the paper, and an important sharp bound on the sum of Bernoulli random variables. Given all necessary tools, Sect. 3 presents upper bounds on the expected optimisation time of the UMDA on both LeadingOnes and BinVal, followed by the derivation of the upper bounds on the expected optimisation time of the UMDA on OneMax. The latter consists of two smaller subsections according to two different ranges of values of the parent population size. Section 5 presents a brief empirical analysis of the UMDA on LeadingOnes, BinVal and OneMax to support the theoretical findings in Sects. 3 and 4. Finally, our concluding remarks are given in Sect. 6.

2 Preliminaries

This section describes the three standard benchmark problems, the algorithm under investigation and the level-based theorem, which is a general method to derive upper bounds on the expected optimisation time of non-elitist population-based algorithms. Furthermore, a sharp upper bound on the sum of independent Bernoulli trials, which is essential in the runtime analysis of the UMDA on OneMax for a small population size, is presented, followed by Feige's inequality.

We use the following notation throughout the paper. The natural logarithm is denoted as $\ln(\cdot)$, and $\log(\cdot)$ denotes the logarithm with base 2. Let $[n]$ be the set $\{1, 2, \ldots, n\}$. The floor and ceiling functions are $\lfloor x \rfloor$ and $\lceil x \rceil$, respectively, for $x \in \mathbb{R}$. For two random variables X, Y, we use $X \preceq Y$ to indicate that Y stochastically dominates X, that is $\Pr(X \geq k) \leq \Pr(Y \geq k)$ for all $k \in \mathbb{R}$.

Table 1 Expected optimisation time (number of fitness evaluations) of univariate EDAs on the three problems OneMax, LeadingOnes and BinVal

Problem	Algorithm	Constraints	Runtime
OneMax	UMDA	$\lambda = \Theta(\mu), \lambda = \mathcal{O}(\text{poly(n)})$	$\Omega(\lambda\sqrt{n} + n\log n)$ [26]
		$\lambda = \Theta(\mu),\ \mu = \Omega(\log n) \cap o(n)$	$\mathcal{O}(\lambda n)$ [44]
		$\lambda = \Theta(\mu),\ \mu = \Omega(\sqrt{n}\log n)$	$\mathcal{O}\left(\lambda\sqrt{n}\right)$ [44]
		$\lambda = \Omega(\mu),\ \mu = \Omega(\log n) \cap \mathcal{O}\left(\sqrt{n}\right)$	$\mathcal{O}(\lambda n)$ [Theorem 8]
		$\lambda = \Omega(\mu),\ \mu = \Omega(\sqrt{n}\log n)$	$\mathcal{O}\left(\lambda\sqrt{n}\right)$ [Theorem 9]
	PBIL*	$\mu = \omega(n), \lambda = \omega(\mu)$	$\omega(n^{3/2})$ [46]
	cGA	$K = n^{1/2+\epsilon}$	$\Theta(K\sqrt{n})$ [14]
		$K = \mathcal{O}\left(\sqrt{n}/\log^2 n\right)$	$\Omega(K^{1/3}n + n\log n)$ [32]
	scGA	$\rho = \Omega(1/\log n), a = \Theta(\rho), c > 0$	$\Omega(\min\{2^{\Theta(n)}, 2^{c/\rho}\})$ [13]
LeadingOnes	UMDA	$\mu = \Omega(\log n), \lambda = \Omega(\mu)$	$\mathcal{O}\left(n\lambda\log\lambda + n^2\right)$ [Theorem 7]
	PBIL	$\lambda = n^{1+\epsilon}, \mu = \mathcal{O}\left(n^{\epsilon/2}\right), \epsilon \in (0,1)$	$\mathcal{O}\left(n^{2+\epsilon}\right)$ [46]
		$\lambda = \Omega(\mu), \mu = \Omega(\log n)$	$\mathcal{O}\left(n\lambda\log\lambda + n^2\right)$ [28]
	scGA	$\rho = \Theta(1/\log n), a = \mathcal{O}(\rho)$	$\mathcal{O}(n\log n)$ [18]
BinVal	UMDA	$\mu = \Omega(\log n), \lambda = \Omega(\mu)$	$\mathcal{O}\left(n\lambda\log\lambda + n^2\right)$ [Theorem 7]
	PBIL	$\lambda = \Omega(\mu), \mu = \Omega(\log n)$	$\mathcal{O}\left(n\lambda\log\lambda + n^2\right)$ [28]
	cGA	$K = n^{1/2+\epsilon}$	$\Theta(Kn)$ [14]
		any $K > 0$	$\Omega(n^2)$ [45]

*Without margins

We consider a partition of the *finite* search space $\mathcal{X} = \{0, 1\}^n$ into m ordered subsets $A_1, \ldots, A_m$ called *levels*, i.e. $A_i \cap A_j = \emptyset$ for any $i \neq j$ and $\cup_{i=1}^m A_i = \mathcal{X}$. The union of all levels above j inclusive is denoted $A_{\geq j} := \cup_{i=j}^m A_i$. An optimisation problem on $\mathcal{X}$ is assumed, without loss of generality, to be the maximisation of some function $f\colon \mathcal{X} \to \mathbb{R}$. A partition is called *fitness-based* (or f-based) if for any $j \in [m-1]$ and all $x \in A_j, y \in A_{j+1}\colon f(y) > f(x)$. An f-based partitioning is called *canonical* when $x, y \in A_j$ if and only if $f(x) = f(y)$.

Given the search space $\mathcal{X}$, each $x \in \mathcal{X}$ is called a *search point* (or *individual*), and a *population* is a vector of search points, i.e. $P \in \mathcal{X}^\lambda$. For a finite population $P = \left(x^{(1)}, \ldots, x^{(\lambda)}\right)$, we define $|P \cap A_j| := |\{i \in [\lambda] \mid x^{(i)} \in A_j\}|$, i.e. the number of individuals in population P which are in level A_j. *Truncation selection*, denoted as (μ, λ)*-selection* for some $\mu < \lambda$, applied to population P transforms it into a vector P' (called *selected* population) with $|P'| = \mu$ by discarding the $\lambda - \mu$ worst search points of P with respect to some fitness function f, where ties are broken uniformly at random.

2.1 Three Problems

We consider the three pseudo-Boolean functions: OneMax, LeadingOnes and BinVal, which are defined over the finite binary search space $\mathcal{X} = \{0, 1\}^n$ and widely used as theoretical benchmark problems in runtime analyses of EDAs [10,14,26,28,44,46]. Note in particular that these problems are only required to describe and compare the behaviour of the EDAs on problems with well-understood structures. The first problem, as its name may suggest, simply counts the number of ones in the bitstring and is widely used to test the performance of EDAs as a hill climber [25]. While the bits in OneMax have the same contributions to the overall fitness, BinVal, which aims at maximising the binary value of the bitstring, has exponentially scaled weights relative to bit positions. In contrast, LeadingOnes counts the number of leading ones in the bitstring. Since bits in this particular problem are highly correlated, it is often used to study the ability of EDAs to cope with dependencies among decision variables [25].

The global optimum for all functions is the all-ones bitstring, i.e. 1^n. For any bitstring $x = (x_1, \ldots, x_n) \in \mathcal{X}$, these functions is defined as follows:

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

Definition 2 $\textsc{LeadingOnes}(x) := \sum_{i=1}^{n} \prod_{j=1}^{i} x_j.$

Definition 3 $\textsc{BinVal}(x) := \sum_{i=1}^{n} 2^{n-i} x_i.$

2.2 Univariate Marginal Distribution Algorithm

Introduced by Mühlenbein and Paaß [37], the Univariate Marginal Distribution Algorithm (UMDA; see Algorithm 1) is one of the simplest EDAs, which assume independence between decision variables. To optimise a pseudo-Boolean function $f\colon \{0,1\}^n \to \mathbb{R}$, the algorithm follows an iterative process: *sample* independently and identically a population of λ offspring from the current probabilistic model and *update* the model using the μ fittest individuals. Each sample-and-update cycle is called a *generation* (or *iteration*). The probabilistic model in generation $t \in \mathbb{N}$ is represented as a vector $p_t = (p_t(1), \ldots, p_t(n)) \in [0,1]^n$, where each component (or *marginal*) $p_t(i) \in [0,1]$ for $i \in [n]$ and $t \in \mathbb{N}$ is the probability of sampling a one at the i-th bit position of an offspring in generation t. Each individual $x = (x_1, \ldots, x_n) \in \{0,1\}^n$ is therefore sampled from the joint probability distribution

$$\Pr(x \mid p_t) = \prod_{i=1}^{n} p_t(i)^{x_i}\,(1 - p_t(i))^{(1-x_i)}. \tag{1}$$

Note that the probabilistic model is initialised as $p_0(i) := 1/2$ for each $i \in [n]$. Let $x_t^{(1)}, \ldots, x_t^{(\lambda)}$ be λ individuals that are sampled from the joint probability distribution (1), then μ of which with the fittest fitness are selected to obtain the next model p_{t+1}. Let $x_{t,i}^{(k)}$ denote the value of the i-th bit position of the k-th individual in the current sorted population P_t. For each $i \in [n]$, the corresponding marginal of the next model is

$$p_{t+1}(i) := \frac{1}{\mu}\sum_{k=1}^{\mu} x_{t,i}^{(k)},$$

which can be interpreted as the frequency of ones among the μ fittest individuals at bit-position i.

The extreme probabilities—zero and one—must be avoided for each marginal $p_t(i)$; otherwise, the bit in position i would remain fixed forever at either zero or one, obstructing some regions of the search space. To avoid this, all marginals $p_{t+1}(i)$ are usually restricted within the closed interval $[1/n, 1-1/n]$, and such values $1/n$ and $1-1/n$ are called *lower* and *upper borders*, respectively. The algorithm in this case is known as the UMDA *with margins*.

2.3 Level-Based Theorem

We are interested in the optimisation time of the UMDA, which is a non-elitist algorithm; thus, tools for analysing runtime for this class of algorithms are of importance. Currently in the literature, *drift theorems* have often been used to derive upper and lower bounds on the expected optimisation time of the UMDA, see, e.g., [26,44] because they allow us to examine the dynamics of each marginal in the vector-based probabilistic model. In this paper, we take another perspective where we consider the population of individuals. To do this, we make use of the so-called level-based theorem.

Introduced by Corus et al. [9], the level-based theorem is a general tool that provides upper bounds on the expected optimisation time of many non-elitist population-based

Algorithm 1: UMDA with margins

```
parameter: offspring population size $\lambda$, parent population size $\mu$, maximising $f$
$t \leftarrow 0$
initialise $p_0(i) \leftarrow 1/2$ for each $i \in [n]$
repeat
    for $k = 1, 2, \ldots, \lambda$ do
        sample $x_{t,i}^{(k)} \sim \text{Bernoulli}(p_t(i))$ for each $i \in [n]$
    sort $P_t \leftarrow \{x_t^{(1)}, x_t^{(2)}, \ldots, x_t^{(\lambda)}\}$ s.t. $f(x_t^{(1)}) \geq f(x_t^{(2)}) \geq \ldots \geq f(x_t^{(\lambda)})$
    for $i = 1, 2, \ldots, n$ do
        $X_i \leftarrow \sum_{k=1}^{\mu} x_{t,i}^{(k)}$
        $p_{t+1}(i) \leftarrow \max\left\{\frac{1}{n}, \min\left\{1 - \frac{1}{n}, \frac{X_i}{\mu}\right\}\right\}$
    $t \leftarrow t + 1$
until termination condition is fulfilled
```

Algorithm 2: Non-elitist population-based algorithm

```
$t \leftarrow 0$
initialise population $P_0$
repeat
    for $i = 1, \ldots, \lambda$ do
        sample $P_{t+1}(i) \sim \mathcal{D}(P_t)$ independently
    $t \leftarrow t + 1$
until termination condition is fulfilled
```

algorithms on a wide range of optimisation problems [9]. It has been applied to analyse the expected optimisation time of Genetic Algorithms with or without crossover on various pseudo-Boolean functions and combinatorial optimisation problems [9], self-adaptive EAs [11], the UMDA with margins on OneMax and LeadingOnes [10], and very recently the PBIL with margins on LeadingOnes and BinVal [28].

The theorem assumes that the algorithm to be analysed can be described in the form of Algorithm 2. The population P_t in generation $t \in \mathbb{N}$ of λ individuals is represented as a vector $(P_t(1), \ldots, P_t(\lambda)) \in \mathcal{X}^\lambda$. The theorem is general because it does not assume specific fitness functions, selection mechanisms, or generic operators like mutation and crossover. Rather, the theorem assumes that there exists, possibly implicitly, a mapping $\mathcal{D}$ from the set of populations $\mathcal{X}^\lambda$ to the space of probability distributions over the search space $\mathcal{X}$. The distribution $\mathcal{D}(P_t)$ depends on the current population P_t, and all individuals in population P_{t+1} are sampled identically and independently from this distribution [9]. The assumption of independent sampling of the individuals holds for the UMDA, and many other algorithms.

Furthermore, the theorem assumes a partition $A_1, \ldots, A_m$ of the finite search space $\mathcal{X}$ into m subsets, which we call *levels*. We assume that the last level A_m consists of all optimal solutions. Given a partition of the search space $\mathcal{X}$, we can state the level-based theorem as follows:

Theorem 4 [9] *Given a partition* $(A_1, \ldots, A_m)$ *of* $\mathcal{X}$*, define* $T := \min\{t\lambda \mid |P_t \cap A_m| > 0\}$*, where for all* $t \in \mathbb{N}$*,* $P_t \in \mathcal{X}^\lambda$ *is the population of Algorithm 2 in generation* *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* $P_t \in \mathcal{X}^\lambda$,

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

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

- ***(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*

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

- ***(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 $z_* := \min_{j \in [m-1]}\{z_j\}$, *then*

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

Informally, the first condition (G1) requires that the probability of sampling an individual in levels $A_{\geq j+1}$ is at least z_j given that at least $\gamma_0\lambda$ individuals in the current population are in levels $A_{\geq j}$. Condition (G2) further requires that at least $\gamma\lambda$ of them are in levels $A_{\geq j+1}$, the probability of sampling an offspring in levels $A_{\geq j+1}$ is at least $(1+\delta)\gamma$. The last condition (G3) sets a lower limit on the population size λ. As long as the three conditions are satisfied, an upper bound on the expected time to reach the last level A_m of a population-based algorithm is guaranteed.

To apply the level-based theorem, it is recommended to follow the five-step procedure in [9]: (1) identifying a partition of the search space (2) finding appropriate parameter settings such that condition (G2) is met (3) estimating a lower bound z_j to satisfy condition (G1) (4) ensuring the the population size is large enough and (5) derive the upper bound on the expected time to reach level A_m.

Note in particular that Algorithm 2 assumes a mapping $\mathcal{D}$ from the space of populations $\mathcal{X}^\lambda$ to the space of probability distributions over the search space. The mapping $\mathcal{D}$ is often said to depend on the current population only [9]; however, this is not strictly necessary. Very recently, Lehre and Nguyen [28] applied Theorem 4 to analyse the expected optimisation time of the PBIL with a sufficiently large offspring population size $\lambda = \Omega(\log n)$ on LeadingOnes and BinVal, when the population for the next generation is sampled using a mapping that depends on the previous probabilistic model p_t in addition to the current population P_t. The rationale behind this is that, in each generation, the PBIL draws λ samples from the probability distribution (1), that correspond to λ individuals in the current population. If the number of samples λ is sufficiently large, it is highly likely that the empirical distributions for all positions among the entire population cannot deviate too far from the true distributions, i.e. marginals $p_t(i)$ [28], due to the Dvoretzky–Kiefer–Wolfowitz inequality [34].

2.4 Feige's Inequality

In order to verify conditions (G1) and (G2) of Theorem 4 for the UMDA on OneMax using a canonical f-based partition $A_1, \ldots, A_m$, we later need a lower bound on the probability of sampling an offspring in given levels, that is $\Pr_{y \sim p_t}(y \in A_{\geq j})$, where y is the offspring sampled from the joint probability distribution (1). Let Y denote the number of ones in the offspring y. It is well-known that the random variable Y follows a Poisson–Binomial distribution with expectation $\mathbb{E}[Y] = \sum_{i=1}^{n} p_t(i)$ and variance $\sigma_n^2 = \sum_{i=1}^{n} p_t(i)\,(1 - p_t(i))$. A general result due to Feige [16] provides such a lower bound when $Y < \mathbb{E}[Y]$; however, for our purposes, it will be more convenient to use the following variant [10].

Theorem 5 (Corollary 3 in [10]) *Let $Y_1, \ldots, Y_n$ be n independent random variables with support in* $[0, 1]$, *define $Y = \sum_{i=1}^{n} Y_i$ and $\mu = \mathbb{E}[Y]$. It holds for every $\varDelta > 0$ that*

$$\Pr(Y > \mu - \varDelta) \geq \min\left\{\frac{1}{13}, \frac{\varDelta}{1 + \varDelta}\right\}.$$

2.5 Anti-concentration Bound

In addition to Feige's inequality, it is also necessary to compute an upper bound on the probability of sampling an offspring in a given level, that is $\Pr_{y \sim p_t}\left(y \in A_j\right)$ for any $j \in [m]$, where $y \sim \Pr(\cdot \mid p_t)$ as defined in (1). Let Y be the random variable that follows a Poisson–Binomial distribution as introduced in the previous subsection. Baillon et al. [3] derived the following sharp upper bound on the probability $\Pr_{y \sim p_t}\left(y \in A_j\right)$.

Theorem 6 (Adapted from Theorem 2.1 in [3]) *Let Y be an integer-valued random variable that follows a Poisson–Binomial distribution with parameters n and p_t, and let $\sigma_n^2 = \sum_{i=1}^{n} p_t(i)(1 - p_t(i))$ be the variance of Y. For all n, y and p_t, it then holds that*

$$\sigma_n \cdot \Pr(Y = y) \leq \eta,$$

where η is an absolute constant being

$$\eta = \max_{x \geq 0} \sqrt{2x}e^{-2x} \sum_{k=0}^{\infty} \left(\frac{x^k}{k!}\right)^2 \approx 0.4688.$$

3 Runtime of the UMDA on LeadingOnes and BinVal

As a warm-up example, and to illustrate the method of level-based analysis, we consider the two functions—LeadingOnes and BinVal—as defined in Definitions 2 and 3. It is well-known that the expected optimisation time of the (1+1) EA on LeadingOnes is $\Theta(n^2)$, and that this is optimal for the class of *unary unbiased* black-box algorithms [29]. Early analysis of the UMDA on LeadingOnes [8] required an excessively large population, i.e. $\lambda = \omega(n^2 \log n)$. Our analysis below shows that a population size $\lambda = \Omega(\log n)$ suffices to achieve the expected optimisation time $\mathcal{O}(n^2)$.

Theorem 7 *The UMDA (with margins) with parent population size $\mu \geq c \log n$ for a sufficiently large constant $c > 0$, and offspring population size $\lambda \geq (1+\delta)e\mu$ for any constant $\delta > 0$, has expected optimisation time $\mathcal{O}(n\lambda \log \lambda + n^2)$ on* LeadingOnes *and* BinVal.

Proof We apply Theorem 4 by following the guidelines from [9].

Step 1 For both functions, we define the levels

$$A_j := \{x \in \{0,1\}^n \mid \text{LeadingOnes}(x) = j - 1\}.$$

Thus, there are $m = n + 1$ levels ranging from A_1 to A_{n+1}. Note that a constant γ_0 appearing later in this proof is set to $\gamma_0 := \mu/\lambda$, that coincides with the selective pressure of the UMDA.

For LeadingOnes, the partition is clearly f-based as it is canonical to the function. For BinVal, however, note that since all the $j-1$ leading bits of any $x \in A_j$ are ones, then the contribution of these bits to $\text{BinVal}(x)$ is $\sum_{i=1}^{j-1} 2^{n-i}$. On the other hand, the contribution of bit position j is 0, and that of the last $n-j$ bits is between 0 and $\sum_{i=j+1}^{n} 2^{n-i} = \sum_{i=0}^{n-j-1} 2^i = 2^{n-j} - 1$, so in overall

$$\sum_{i=1}^{j} 2^{n-i} - 1 \geq \text{BinVal}(x) \geq \sum_{i=1}^{j-1} 2^{n-i}.$$

Therefore, for any $j \in [n]$ and all $x \in A_j$, and all $y \in A_{j+1}$ we have that

$$\text{BinVal}(y) \geq \sum_{i=1}^{j} 2^{n-i} > \sum_{i=1}^{j} 2^{n-i} - 1 \geq \text{BinVal}(x);$$

thus, the partition is also f-based for BinVal. This observation allows us to carry over the proof arguments of LeadingOnes to BinVal.

Step 2 In (G2), for any level $j \in [n-1]$ satisfying $|P_t \cap A_{\geq j}| \geq \gamma_0 \lambda = \mu$ and $|P_t \cap A_{\geq j+1}| \geq \lambda\gamma$ for some $\gamma \in (0, \gamma_0]$, we seek a lower bound $(1+\delta)\gamma$ for $\Pr\left(y \in A_{\geq j+1}\right)$ where $y \sim \mathcal{D}(P_t)$. The given conditions on j imply that the μ fittest individuals of P_t have at least $j-1$ leading 1-bits and among them at least

$\lceil \gamma\lambda \rceil$ have at least j leading 1-bits. Hence, $p_{t+1}(i) = 1 - 1/n$ for $i \in [j-1]$ and $p_{t+1}(j) \geq \max(\min(1-1/n, \gamma\lambda/\mu), 1/n) \geq \min(1-1/n, \gamma/\gamma_0)$, so

$$\begin{aligned}\Pr\left(y \in A_{\geq j+1}\right) &\geq \prod_{i=1}^{j} p_{t+1}(i) \geq \min\left\{\left(1-\frac{1}{n}\right)^{j}, \left(1-\frac{1}{n}\right)^{j-1} \cdot \frac{\gamma\lambda}{\mu}\right\} \\ &\geq \min\left\{\frac{1}{e}, \frac{\gamma}{e\gamma_0}\right\} = \frac{\gamma}{e\gamma_0} = \frac{\lambda\gamma}{e\mu} \geq (1+\delta)\gamma,\end{aligned}$$

due to $\gamma \leq \gamma_0$ and $\lambda \geq (1+\delta)e\mu$ for any constant $\delta > 0$. Therefore, condition (G2) is now satisfied.

Step 3 In (G1), for any level $j \in [n]$ satisfying $|P_t \cap A_{\geq j}| \geq \gamma_0\lambda = \mu$ we need a lower bound $\Pr\left(y \in A_{\geq j+1}\right) \geq z_j$. Again the condition on level j gives that the μ fittest individuals of P_t have at least $j-1$ leading 1-bits, or $p_{t+1}(i) = 1 - \frac{1}{n}$ for $i \in [j-1]$. Due to the imposed lower margin, we can assume pessimistically that $p_{t+1}(j) = \frac{1}{n}$. Hence,

$$\Pr\left(y \in A_{\geq j+1}\right) \geq \prod_{i=1}^{j} p_{t+1}(i) \geq \left(1-\frac{1}{n}\right)^{j-1} \cdot \frac{1}{n} = \frac{1}{en} =: z_j.$$

So, (G1) is satisfied for $z_j := \frac{1}{en}$.

Step 4 Considering (G3), because δ is a constant, and both $1/z_*$ and m are $\mathcal{O}(n)$, there must exist a constant $c > 0$ such that $\mu \geq c \log n \geq (4/\delta^2)\ln(128m/(z_*\delta^2))$. Note that $\lambda = \mu/\gamma_0$, so (G3) is satisfied.

Step 5 All conditions of Theorem 4 are satisfied, so the expected optimisation time of the UMDA on LeadingOnes is

$$\mathbb{E}[T] = \mathcal{O}\left(\sum_{j=1}^{n}\left(\lambda \ln\left(\frac{\lambda}{1+\lambda/n}\right) + n\right)\right) = \mathcal{O}\left(n\lambda\log\lambda + n^2\right).$$

We now consider BinVal. In both problems, all that matters to determine the level of a bitstring is the position of the leftmost zero-bit. Now consider two bitstrings in the same level for BinVal, their rankings after the population is sorted are also determined by some other less significant bits; however, the proof thus far never takes these bits into account. Hence, the expected optimisation time of the UMDA on LeadingOnes can be carried over to BinVal for the UMDA with margins using truncation selection. □

4 Runtime of the UMDA on OneMax

We consider the problem in Definition 1, i.e., maximisation of the number of ones in a bitstring. It is well-known that OneMax can be optimised in expected time $\Theta(n \log n)$ using the simple (1 + 1) EA. The level-based theorem yielded the first upper bound $\mathcal{O}(n\lambda \log \lambda)$ on the expected optimisation time of the UMDA on OneMax, assuming that $\lambda = \Omega(\log n)$ [10]. This leaves open whether an improved bound $\mathcal{O}(n\lambda)$ can be obtained for the UMDA (with margins) on problem OneMax.

We now introduce additional notation used throughout the section. The following random variables related to the sampling of a Poisson Binomial distribution with the parameter vector $p_t = (p_t(1), \dots, p_t(n))$ are often used in the proofs.

- Let $Y := (Y_1, Y_2, \dots, Y_n)$ denote an offspring sampled from the probability distribution (1) in generation t, where $\Pr(Y_i = 1) = p_t(i)$ for each $i \in [n]$.
- Let $Y_{i,j} := \sum_{k=i}^{j} Y_k$ denote the number of ones sampled from the sub-vector $(p_t(i), p_t(i+1), \dots, p_t(j))$ of the model p_t where $1 \leq i \leq j \leq n$.

4.1 Small Parent Population Size

Our approach refines the analysis in [10] by considering anti-concentration properties of the random variables involved. As already discussed in Sect. 2.3, we need to verify the three conditions (G1), (G2) and (G3) of Theorem 4 to derive an upper bound on the expected optimisation time. The range of values of the marginals are (assuming that $\mu < n$)

$$p_t(i) \in \left\{ \frac{k}{\mu} \mid k \in [\mu - 1] \right\} \cup \left\{ 1 - \frac{1}{n}, \frac{1}{n} \right\}.$$

When $p_t(i) = 1 - 1/n$ or $1/n$, we say that the marginal is at the upper or lower border (or margin), respectively. Therefore, we can categorise values for $p_t(i)$ into three groups: those at the upper margin $1 - 1/n$, those at the lower margin $1/n$, 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. Given the current sorted population, recall that $X_i := \sum_{k=1}^{\mu} x_{t,i}^{(k)}$, and without loss of generality, we can re-arrange the bit-positions so that for two integers $k, \ell \geq 0$, it holds

- for all $i \in [1, k]$, $1 \leq X_i \leq \mu - 1$ and $p_t(i) = X_i/\mu$,
- for all $i \in (k, k + \ell]$, $X_i = \mu$ and $p_t(i) = 1 - 1/n$, and
- for all $i \in (k + \ell, n]$, $X_i = 0$ and $p_t(i) = 1/n$.

We define the levels using the canonical f-based partition

$$A_j := \left\{ x \in \{0, 1\}^n \mid \textsc{OneMax}(x) = j - 1 \right\}. \tag{2}$$

Note that the probability appearing in conditions (G1) and (G2) of Theorem 4 is the probability of sampling an offspring in levels $A_{\geq j+1}$, that is $\Pr\left(Y_{1,n} \geq j\right)$.

We aim at obtaining an upper bound $\mathcal{O}(n\lambda)$ on the expected optimisation time of the UMDA on OneMax using the level-based theorem. The logarithmic factor $\mathcal{O}(\log \lambda)$ in the previous upper bound $\mathcal{O}(n\lambda \log \lambda)$ in [10] stems from the lower bound $\Omega(1/\mu)$ on the parameter z_j in the condition (G1) of Theorem 4. We aim for the stronger bound $z_j = \Omega(\frac{n-j+1}{n})$. Note that in the following proofs, we choose the parameter $\gamma_0 := \mu/\lambda$.

Assume that the current level is A_j, that is $|P_t \cap A_{\geq j}| \geq \gamma_0\lambda = \mu$, which, together with the two variables k and ℓ, implies that there are at least $j - \ell - 1$ ones from the first k bit positions. To verify conditions (G1) and (G2) of Theorem 4, we need to calculate the probability of sampling an offspring in levels $A_{\geq j+1}$. It is thus more likely for the algorithm to maintain the ℓ ones for all bit positions $i \in (k, k+\ell]$ (actually this happens with probability at least $1/e$), and also sample at least $j - \ell$ ones from the remaining $n - \ell$ remaining bit positions. This lead us to consider three distinct cases according to different configurations of the current population with respect to the two parameters k and j in Step 3 of Theorem 8 below.

1. $k \geq \mu$. In this situation, the variance of $Y_{1,k}$ is not too small. By the result of Theorem 6, the distribution of $Y_{1,k}$ cannot be too concentrated on its mean $\mathbb{E}\left[Y_{1,k}\right] = j - \ell - 1$, and with probability at least $\Omega(1)$, the algorithm can sample at least $j - \ell$ ones from the first k bit positions to obtain an offspring with at least $(j - \ell) + \ell = j$ ones. Thus, the probability of sampling at least j ones is bounded from below by
$$\Pr(Y_{1,n} \geq j) \geq \Pr(Y_{1,k} \geq j - \ell) \Pr(Y_{k+1,k+\ell} = \ell) = \Omega(1).$$
2. $k < \mu$ and $j \geq n + 1 - \frac{n}{\mu}$. In this case, the current level is very close to the last level A_{n+1}, and the bitstring has few zeros. As already obtained from [10], the probability of sampling an offspring in $A_{\geq j+1}$ in this case is $\Omega(\frac{1}{\mu})$. Since the condition can be rewritten as $\frac{1}{\mu} \geq \frac{n-j+1}{n}$, it ensures that $z_j = \Omega(\frac{1}{\mu}) = \Omega(\frac{n-j+1}{n})$.
3. The remaining cases. Later will we prove that if $\mu \leq \sqrt{n(1-c)}$ for some constant $c \in (0,1)$, and excluding the two cases above, imply $0 \leq k < (1-c)(n-j+1)$. In this case, k is relatively small, and ℓ is not too large since the current level is not very close to the last level A_{n+1}. This implies that most zeros must be located among bit positions $i \in (k+\ell, n]$, and it suffices to sample an extra one from this region to get at least $(j - \ell - 1) + \ell + 1 = j$ ones. The probability of sampling an offspring in levels $A_{\geq j+1}$ is then $z_j = \Omega(\frac{n-j+1}{n})$.

We now present our detailed runtime analysis for the UMDA on OneMax, when the population size is small, that is, $\mu = \Omega(\log n) \cap \mathcal{O}(\sqrt{n})$.

Theorem 8 *For some constant $a > 0$ and any constant $c \in (0,1)$, the UMDA (with margins) with parent population size $a\ln(n) \leq \mu \leq \sqrt{n(1-c)}$, and offspring population size $\lambda \geq (13e/(1-c))\mu$, has expected optimisation time $\mathcal{O}(n\lambda)$ on* OneMax.

Proof We re-arrange the bit positions as explained above and follow the recommended 5-step procedure for applying Theorem 4 [9].

Step 1 The levels are defined as in Eq. (2). There are exactly $m = n + 1$ levels from A_1 to A_{n+1}, where level A_{n+1} consists of the optimal solution.

Step 2 We verify condition (G2) of Theorem 4. In particular, for some $\delta \in (0,1)$, for any level $j \in [m-2]$ and any $\gamma \in (0, \gamma_0]$, assuming that the population is configured such that $|P_t \cap A_{\geq j}| \geq \gamma_0\lambda = \mu$ and $|P_t \cap A_{\geq j+1}| \geq \gamma\lambda > 0$, we must show that the probability of sampling an offspring in levels $A_{\geq j+1}$ must be no less than $(1+\delta)\gamma$. By the re-arrangement of the bit-positions mentioned earlier, it holds that

$$\sum_{i=k+1}^{k+\ell} X_i = \mu\ell \quad \text{and} \quad \sum_{i=k+\ell+1}^{n} X_i = 0, \tag{3}$$

where X_i for all $i \in [n]$ are given in Algorithm 1. By assumption, the current population P_t consists of $\gamma\lambda$ individuals with at least j ones and $\mu - \gamma\lambda$ individuals with exactly $j-1$ ones. Therefore,

$$\sum_{i=1}^{n} X_i \geq \gamma\lambda j + (\mu - \gamma\lambda)(j-1) = \gamma\lambda + \mu(j-1). \tag{4}$$

Combining (3), (4) and noting that $\lambda = \mu/\gamma_0$ yield

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

Let $Z = Y_{1,k} + Y_{k+\ell+1,n}$ be the integer-valued random variable, which describes the number of ones sampled in the first k and the last $n-k-\ell$ bit positions. Since $k + \ell \leq n$, the expected value of Z is

$$\mathbb{E}[Z] = \sum_{i=1}^{k} p_t(i) + \sum_{i=k+\ell+1}^{n} p_t(i) = \frac{1}{\mu}\sum_{i=1}^{k} X_i + \frac{n-k-\ell}{n} \geq j - \ell - 1 + \frac{\gamma}{\gamma_0}. \tag{5}$$

In order to obtain an offspring with at least j ones, it is sufficient to sample ℓ ones in positions $k+1$ to $k+\ell$ and at least $j-\ell$ ones from the other positions. The probability of this event is bounded from below by

$$\Pr\left(Y_{1,n} \geq j\right) \geq \Pr\left(Z \geq j - \ell\right) \cdot \Pr\left(Y_{k+1,k+\ell} = \ell\right). \tag{6}$$

The probability to obtain $\ell \geq n-1$ ones in the middle interval from position $k+1$ to $k+\ell$ is

$$\Pr\left(Y_{k+1,k+\ell} = \ell\right) = \left(1 - \frac{1}{n}\right)^{\ell} \geq \left(1 - \frac{1}{n}\right)^{n-1} \geq \frac{1}{e} \tag{7}$$

by the result of Lemma 10 for $t = -1$. We now estimate the probability $\Pr(Z \geq j - \ell)$ using Feige's inequality. Since Z takes integer values only, it follows by (5) that

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

Applying Theorem 5 for $\Delta = \gamma/\gamma_0 \leq 1$ and noting that we chose μ and λ such that $1/\gamma_0 = \lambda/\mu \geq 13e/(1-c) = 13e(1+\delta)$ yield

$$\Pr(Z \geq j - \ell) \geq \min\left\{\frac{1}{13}, \frac{\Delta}{\Delta+1}\right\} \geq \frac{\Delta}{13} = \frac{\gamma}{13\gamma_0} \geq e(1+\delta)\gamma. \tag{8}$$

Combining (6), (7), and (8) yields $\Pr\left(Y_{1,n} \geq j\right) \geq (1+\delta)\gamma$, and, thus, condition (G2) of Theorem 4 holds.

Step 3 We now consider condition (G1) for any level j. Let P_t be any population where $|P_t \cap A_{\geq j}| \geq \gamma_0\lambda = \mu$. For a lower bound on $\Pr\left(Y_{1,n} \geq j\right)$, we modify the population such that any individual in levels $A_{\geq j+1}$ is moved to level A_j. Thus, the μ fittest individuals belong to level A_j. By the definition of the UMDA, this will only reduce the probabilities $p_{t+1}(i)$ on the OneMax problem. Hence, by Lemma 13, the distribution of $Y_{1,n}$ for the modified population is stochastically dominated by $Y_{1,n}$ for the original population. A lower bound z_j that holds for the modified population therefore also holds for the original population. All the μ fittest individuals in the current sorted population P_t have exactly $j-1$ ones, and, therefore,$\sum_{i=1}^{n} X_i = \mu(j-1)$ and $\sum_{i=1}^{k} X_i = \mu(j-\ell-1)$. There are four distinct cases that cover all situations according to different values of variables k and j. We aim to show that in all four cases, we can use the parameter $z_j = \Omega(\frac{n-j+1}{n})$.

Case 0 $k = 0$. In this case, $p_t(i) = 1 - 1/n$ for $1 \leq i \leq j-1$, and $p_t(i) = 1/n$ for $j \leq i \leq n$. To obtain j ones, it suffices to sample only ones in the first $j-1$ positions, and exactly a one in the remaining positions, i.e.,

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

Case 1 $k \geq \mu$. We will apply the anti-concentration inequality in Theorem 6. To lower bound the variance of the number of ones sampled in the first k positions, we use the bounds $1/\mu \leq p_i(t) \leq 1 - 1/\mu$ which hold for $1 \leq i \leq k$. In particular,

$$\operatorname{Var}\left[Y_{1,k}\right] = \sum_{i=1}^{k} p_t(i)(1 - p_t(i)) \geq \frac{k}{\mu}\left(1 - \frac{1}{\mu}\right) \geq \frac{9k}{10\mu} \geq \frac{9}{10},$$

where the second inequality holds for sufficiently large n because $\mu \geq a\ln(n)$ for some constant $a > 0$. Theorem 6 applied with $\sigma_k \geq \sqrt{9/10}$ now gives

$$\Pr\left(Y_{1,k} = j - \ell - 1\right) \leq \eta/\sigma_k.$$

Furthermore, since $\mathbb{E}\left[Y_{1,k}\right]$ is an integer, Lemma 11 implies that

$$\Pr\left(Y_{1,k} \geq \mathbb{E}\left[Y_{1,k}\right]\right) \geq 1/2. \tag{9}$$

By combining these two probability bounds, the probability of sampling an offspring with at least $j-\ell$ ones from the first k positions is

$$\begin{aligned}\Pr\left(Y_{1,k} \geq j-\ell\right) &= \Pr\left(Y_{1,k} \geq j-\ell-1\right) - \Pr\left(Y_{1,k} = j-\ell-1\right)\\ &= \Pr\left(Y_{1,k} \geq \mathbb{E}\left[Y_{1,k}\right]\right) - \Pr\left(Y_{1,k} = j-\ell-1\right)\\ &\geq \frac{1}{2} - \frac{\eta}{\sigma_k} > \frac{1}{2} - \frac{0.4688}{\sqrt{9/10}} = \Omega(1).\end{aligned}$$

In order to obtain an offspring in levels $A_{\geq j+1}$, it is sufficient to sample at least $j-\ell$ ones from the k first positions and ℓ ones from position $k+1$ to position $k+\ell$. Therefore, using (7) and the above lower bound, this event happens with probability bounded from below by

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

Case 2 $1 \leq k < \mu$ and $j \geq n(1-1/\mu)+1$. The second condition is equivalent to $1/\mu \geq (n-j+1)/n$. The probability of sampling an offspring in levels $A_{\geq j+1}$ is then bounded from below by

$$\begin{aligned}\Pr\left(Y_{1,n} \geq j\right) &\geq \Pr\left(Y_{1,1} = 1\right) \Pr\left(Y_{2,k} \geq j-\ell-1\right) \Pr\left(Y_{k+1,k+\ell} = \ell\right)\\ &\geq \frac{1}{\mu} \Pr\left(Y_{2,k} \geq j-\ell-1\right) \frac{1}{e} \geq \frac{1}{14e\mu},\end{aligned}$$

where we used the inequality $\Pr\left(Y_{2,k} \geq j-\ell-1\right) \geq 1/14$ for $\mu \geq 14$ proven in [10]. Since $1/\mu \geq (n-j+1)/n$, we can conclude that

$$\Pr\left(Y_{1,n} \geq j\right) \geq \frac{1}{14e\mu} \geq \frac{n-j+1}{14en} = \Omega\left(\frac{n-j+1}{n}\right).$$

Case 3 $1 \leq k < \mu$ and $j < n(1-1/\mu)+1$. This case covers all the remaining situations not included by the first two cases. The latter 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 shortened to $1 \leq k < (1-c)(n-j+1)$. In this case, the probability of sampling j ones is

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

where the $1/2$ factor in the last inequality is due to (9). Since $\ell \leq j-1$ and $k < (1-c)(n-j+1)$, it follows that

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

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

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

and by defining $z_j = c \cdot \frac{n-j+1}{n}$ for a sufficiently small $c > 0$ and choosing $z_* := \min_{j\in[n]}\{z_j\} = \Omega(1/n)$, condition (G1) of Theorem 4 is satisfied.

Step 4 We consider condition (G3) regarding the population size. We have $1/\delta^2 = \mathcal{O}(1)$, $1/z_* = \mathcal{O}(n)$, and $m = \mathcal{O}(n)$. 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.

Step 5 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 optimisation time is therefore

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

We simplify the two terms separately. By Stirling's approximation (see Lemma 12), the first term is

$$\mathcal{O}\left(\lambda\sum_{j=1}^{n}\ln\left(\frac{n}{n-j+1}\right)\right) = \mathcal{O}\left(\lambda\ln\prod_{j=1}^{n}\frac{n}{n-j+1}\right)$$
$$= \mathcal{O}\left(\lambda\ln\left(\frac{n^n}{n!}\right)\right) = \mathcal{O}\left(\lambda\ln\frac{n^n\cdot e^n}{n^{n+1/2}}\right) = \mathcal{O}\left(n\lambda\right).$$

The second term is

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

Since $\lambda > \mu = \Omega(\log n)$, the expected optimisation time is

$$\mathbb{E}\left[T\right] = \mathcal{O}\left(n\lambda\right) + \mathcal{O}\left(n \log n\right) = \mathcal{O}\left(n\lambda\right).$$

□

4.2 Large Parent Population Size

For larger parent population sizes, i.e., $\mu = \Omega(\sqrt{n} \log n)$, we prove the upper bound $\mathcal{O}(\lambda\sqrt{n})$ on the expected optimisation time of the UMDA on OneMax. Witt [44] obtained a similar result, and we actually rely on one of his lemmas to derive our improved result. In overall, our proof is not only significantly simpler but also holds for different settings of μ and λ, that is, $\lambda = \Omega(\mu)$ instead of $\lambda = \Theta(\mu)$.

Theorem 9 *For sufficiently large constants $a > 1$ and $c > 0$, the UMDA (with margins) with offspring population size $\lambda \geq a\mu$, and parent population size $\mu \geq c\sqrt{n} \log n$, has expected optimisation time $\mathcal{O}\left(\lambda\sqrt{n}\right)$ on* OneMax.

Here, we are mainly interested in the parent population size $\mu \geq c\sqrt{n} \log n$ for a sufficiently large constant $c > 0$. In this case, Witt [44] found that $\Pr(T \leq n^{cc'}) = \mathcal{O}(n^{-cc'})$, where c' is another positive constant and $T := \min\{t \geq 0 \mid p_t(i) \leq 1/4\}$ for an arbitrary bit $i \in [n]$. This result implies that the probability of not sampling at least an optimal solution within $n^{cc'}$ generations is bounded by $\mathcal{O}(n^{-cc'})$. Therefore, the UMDA needs $\mathcal{O}(n\lambda \log \lambda)/\lambda = \mathcal{O}(n \log \lambda)$ generations [10] with probability $\mathcal{O}(n^{-cc'})$ and $\mathcal{O}(\lambda\sqrt{n})/\lambda = \mathcal{O}(\sqrt{n})$ with probability $1 - \mathcal{O}(n^{-cc'})$ to optimise OneMax. The expected number of generations is

$$\mathcal{O}(n^{-cc'}) \cdot \mathcal{O}(n \log \lambda) + (1 - \mathcal{O}(n^{-cc'})) \cdot \mathcal{O}(\sqrt{n})$$

If we choose the constant c large enough, then $n \log \lambda$ can subsume any polynomial number of generations, i.e. $n \log \lambda \in \text{poly}(n)$, which leads to $\mathcal{O}(n^{-cc'}) \cdot \mathcal{O}(n \log \lambda) = \mathcal{O}(1)$. Therefore, the overall expected number of generations is still bounded by $\mathcal{O}(\sqrt{n})$, so the expected optimisation time is $\mathcal{O}(\lambda\sqrt{n})$.

In addition, the analysis by Witt [44] implies that all marginals will generally move to higher values and are unlikely to drop by a large distance. We then pessimistically assume that all marginals are lower bounded by a constant $p_{\min} = 1/4$. Again, we rearrange the bit positions such that there exist two integers $0 \leq k, \ell \leq n$, where $k + \ell = n$ and

- $p_t(i) \in \left[p_{\min}, 1 - \frac{1}{\mu}\right]$ for all $1 \leq i \leq k$,

- $p_t(i) = 1 - \frac{1}{n}$ for all $k+1 \le i \le n$.

Note that $k > 0$ because if $k = 0$ we would have sampled a globally optimal solution.

Proof of Theorem 9 We apply Theorem 4.

Step 1 We partition the search space into the m subsets $A_1, \ldots, A_m$ (i.e. levels) defined by

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

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

$$f_0 := 0 \text{ and } f_{i+1} := f_i + \lceil d\sqrt{n - f_i}\rceil. \tag{10}$$

The range of d will be specified later, but for now note that $m = \min\{i \mid f_i = n\} + 1$ and due to Lemma 15,[1] we know 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 $m = \Theta(\sqrt{n})$.

Step 2 For (G2), we assume that $|P_t \cap A_{\ge j}| \ge \gamma_0\lambda = \mu$ and $|P_t \cap A_{\ge j+1}| \ge \gamma\lambda$. Additionally, we make the pessimistic assumption that $|P_t \cap A_{\ge j+2}| = 0$, i.e. the current population contains exactly $\gamma\lambda$ individuals in A_{j+1}, $\mu - \gamma\lambda$ individuals in level A_j, and $\lambda - \mu$ individuals in the levels below A_j. In this case,

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

and

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

The expected value of $Y_{1,k}$ is

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

[1] This and some other lemmas are stated in "Appendix".

Due to the assumption $p_t(i) \geq p_{\min} = 1/4$, the variance of $Y_{1,k}$ is

$$\begin{aligned}\mathrm{Var}\left[Y_{1,k}\right] &= \sum_{i=1}^{k} p_t(i)(1-p_t(i)) \\ &\geq p_{\min}(k - \mathbb{E}\left[Y_{1,k}\right]) \\ &= \frac{1}{4}\left(n - \ell - \mathbb{E}\left[Y_{1,k}\right]\right) \\ &= \frac{1}{4}\left(n - \ell - f_{j-1} - \frac{\gamma}{\gamma_0}\left(f_j - f_{j-1}\right) + \ell\right) \\ &\geq \frac{1}{4}\left(n - f_{j-1} - d\left(n - f_{j-1}\right)\right) = \frac{1}{4}\left(n - f_{j-1}\right)(1-d)\,.\end{aligned}$$

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

$$\Pr\left(Y_{1,n} \geq f_j\right) \geq \Pr(Y_{1,k} \geq f_j - \ell) \cdot \Pr(Y_{k+1,n} = \ell),$$

where

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

and

$$\Pr\left(Y_{1,k} \geq f_j - \ell\right) \geq \Pr\left(Y_{1,k} \geq \mathbb{E}\left[Y_{1,k}\right]\right) - \Pr\left(\mathbb{E}\left[Y_{1,k}\right] \leq Y_{1,k} \leq f_j - \ell\right). \quad (11)$$

By Theorem 6, we have

$$\begin{aligned}\Pr\left(\mathbb{E}\left[Y_{1,k}\right] \leq Y_{1,k} \leq f_j - \ell\right) &\leq \frac{\eta\left(f_j - \ell - \mathbb{E}[Y_{1,k}]\right)}{\sqrt{\mathrm{Var}\left[Y_{1,k}\right]}} \\ &= \eta\left(1 - \frac{\gamma}{\gamma_0}\right)\frac{f_j - f_{j-1}}{\sqrt{\mathrm{Var}\left[Y_{1,k}\right]}} \\ &= 2\eta\left(1 - \frac{\gamma}{\gamma_0}\right)\frac{d}{\sqrt{1-d}} \\ &\leq \left(1 - \frac{\gamma}{\gamma_0}\right)\frac{d}{\sqrt{1-d}}.\end{aligned}$$

The last inequality follows from $\eta \approx 0.4688 < 1/2$. Note that $\Pr\left(Y_{1,k} \geq \mathbb{E}\left[Y_{1,k}\right]\right) \geq \psi = \Omega(1)$ due to Lemma 16, so (11) becomes

$$\Pr(Y_{1,k} \geq f_j - \ell) \geq \psi - \left(1 - \frac{\gamma}{\gamma_0}\right)\frac{d}{\sqrt{1-d}} \geq \psi\frac{\gamma}{\gamma_0}. \quad (12)$$

The last inequality is satisfied if for any $j \in [m-1]$,

$$\frac{d}{\sqrt{1-d}} \le \psi \iff \psi^{-2}d^2 + d - 1 \le 0.$$

The discriminant of this quadratic equation is $\Delta = 1 + 4\psi^{-2} > 0$. Vieta's formula [43] yields that the product of its two solutions is negative, implying that the equation has two real solutions $d_1 < 0$ and $d_2 > 0$. Specifically,

$$d_1 = -(1+\sqrt{\Delta})\psi^2/2 < 0 \quad \text{and} \quad d_2 = (-1+\sqrt{\Delta})\psi^2/2 \in (0,1). \tag{13}$$

Therefore, if we choose any value of d such that $0 < d \le d_2$, then inequality (12) always holds. The probability of sampling an offspring in $A_{\ge j+1}$ is therefore bounded from below by

$$\Pr(Y_{1,n} \ge f_j) \ge \frac{1}{e} \cdot \psi \frac{\gamma}{\gamma_0} \ge (1+\delta)\gamma.$$

The last inequality holds if we choose the population size in the UMDA such that $\mu/\lambda = \gamma_0 \le \psi/(1+\delta)e$, where $\delta \in (0,1]$. Condition (G2) then follows.

Step 3 Assume that $|P_t \cap A_{\ge j}| \ge \gamma_0\lambda = \mu$. This means that the μ fittest individuals in the current sorted population P_t belong to levels $A_{\ge j}$. In other words,

$$\sum_{i=1}^{n} X_i \ge \mu f_{j-1},$$

and

$$\sum_{i=1}^{k} X_i = \sum_{i=1}^{n} X_i - \sum_{i=k+1}^{n} X_i \ge \mu f_{j-1} - \mu\ell = \mu(f_{j-1} - \ell).$$

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

$$\mathbb{E}\left[Y_{1,n}\right] = \sum_{i=1}^{n} p_t(i) = \frac{1}{\mu}\sum_{i=1}^{k} X_i + \sum_{i=k+1}^{n}\left(1 - \frac{1}{n}\right) \ge f_{j-1} - \frac{\ell}{n}. \tag{14}$$

An individual belonging to the higher levels $A_{\ge j+1}$ must have at least f_j ones. The probability of sampling an offspring $y \in A_{\ge j+1}$ is equivalent to $\Pr(Y_{1,n} \ge f_j)$. According to the level definitions and following the result of Lemma 17, we have

$$\begin{aligned}\Pr\left(Y_{1,n} \ge f_j\right) &= \Pr\left(Y_{1,n} \ge f_{j-1} + \lceil d\sqrt{n - f_{j-1}}\rceil\right) \\ &\ge \Pr\left(Y_{1,n} \ge \mathbb{E}\left[Y_{1,n}\right] + d\sqrt{n - \mathbb{E}\left[Y_{1,n}\right]}\right).\end{aligned}$$

In order to obtain a lower bound on $\Pr\left(Y_{1,n} \geq f_j\right)$, we need to bound the probability $\Pr\left(Y_{1,n} \geq \mathbb{E}\left[Y_{1,n}\right] + d\sqrt{n - \mathbb{E}\left[Y_{1,n}\right]}\right)$ from below by a constant. We obtain such a bound by applying the result of Lemma 14. This lemma with constant $d^* \geq 1/p_{\min} = 4$ and $d \leq d^*$ yields

$$\begin{aligned}\Pr\left(Y_{1,n} \geq f_j\right) &\geq \Pr\left(Y_{1,n} \geq \mathbb{E}\left[Y_{1,n}\right] + d\sqrt{n - \mathbb{E}\left[Y_{1,n}\right]}\right) \\ &\geq \Pr\left(Y_{1,n} \geq \min\left\{\mathbb{E}\left[Y_{1,n}\right] + d^*\sqrt{n - \lfloor \mathbb{E}\left[Y_{1,n}\right] \rfloor}, n\right\}\right) \\ &\geq \kappa > 0,\end{aligned}$$

where κ is a constant. Hence, the probability of sampling an offspring in levels $A_{\geq j+1}$ is bounded from below by a positive constant $z_j := \kappa$ independent of n.

Step 4 We consider condition (G3) regarding the population size. We have $1/\delta^2 = \mathcal{O}(1)$, $1/z_* = \mathcal{O}(1)$, and $m = \mathcal{O}(\sqrt{n})$. 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.

Step 5 The probability of sampling an offspring in levels $A_{\geq j+1}$ is bounded from below by $z_j = \kappa$. Having satisfied all three conditions, Theorem 4 then guarantees an upper bound on the expected optimisation time of the UMDA on OneMax, assuming that $\mu = \Omega(\sqrt{n}\log n)$,

$$\mathbb{E}\left[T\right] = \mathcal{O}\left(\lambda\sum_{j=1}^{m}\frac{1}{z_j} + \sum_{j=1}^{m}\frac{1}{z_j}\right) = \mathcal{O}(m\lambda) = \mathcal{O}\left(\lambda\sqrt{n}\right)$$

since $m = \Theta(\sqrt{n})$ due to Lemma 15. □

5 Empirical Results

We have proved upper bounds on the expected optimisation time of the UMDA on OneMax, LeadingOnes and BinVal. However, they are only asymptotic upper bounds as growth functions of the problem and population sizes. They provide no information on the multiplicative constants or the influences of lower order terms. Our goal is also to investigate the runtime behaviour for larger populations. To complement

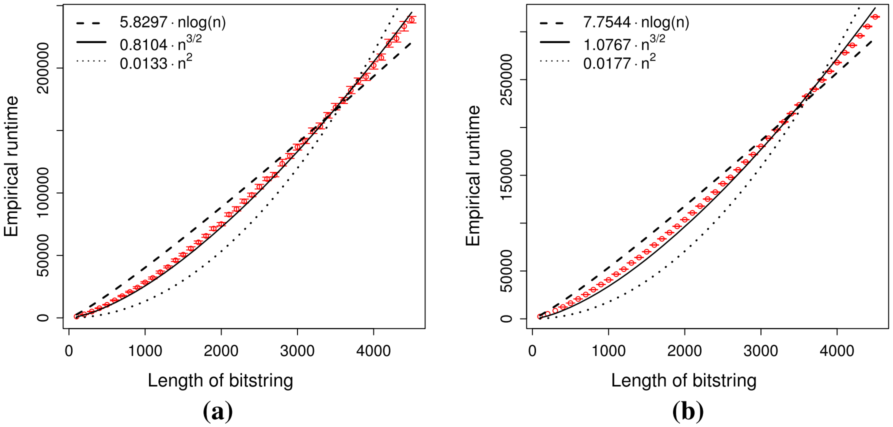

Fig. 1 Average runtime of the UMDA on OneMax with 95% confidence intervals plotted with error bars in red colour. Models are also fitted via non-linear regression. **a** Small μ. **b** Large μ (Color figure online)

the theoretical findings, we therefore carried out some experiments by running the UMDA on the three functions.

For each function, the parameters were chosen consistently with the theoretical analyses. Specifically, we set $\lambda = n$, and $n \in \{100, 200, \ldots, 4500\}$. Although the theoretical results imply that significantly smaller population sizes would suffice, e.g. $\lambda = O(\log n)$ for Theorem 8 we chose a larger population size in the experiments to more easily observe the impact of λ on the running time of the algorithm. The results are shown in Figs. 1, 2 and 3. For each value of n, the algorithm is run 100 times, and then the average runtime is computed. The average runtime for each value of n is estimated with 95% confidence intervals using the *bootstrap percentile method* [30] with 100 bootstrap samples. Each average point is plotted with two error bars to illustrate the upper and lower margins of the confidence intervals.

5.1 OneMax

In Sect. 4, we obtained two upper bounds on the expected optimisation time of the UMDA on OneMax, which are tighter than the earlier upper bound $\mathcal{O}(n\lambda \log \lambda)$ in [10], as follows

- $\mathcal{O}(\lambda n)$ with parent population sizes $\mu = \Omega(\log n) \cap \mathcal{O}(\sqrt{n})$,
- $\mathcal{O}(\lambda\sqrt{n})$ with parent population sizes $\mu = \Omega(\sqrt{n}\log(n))$.

We therefore experimented with two different settings for the parent population size: $\mu = \sqrt{n}$ and $\mu = \sqrt{n}\log(n)$. We call the first setting small population and the other large population. The empirical runtimes are shown in Fig. 1. Theorem 8 implies the upper bounds $\mathcal{O}(n^2)$ for the setting of small population and $\mathcal{O}(n^{3/2})$ for the setting of large population. Following [30], we identify the three positive constants c_1, c_2 and c_3 that best fit the models $c_1 n \log n$, $c_2 n^{3/2}$ and $c_3 n^2$ in non-linear least square regression. Note in particular that these models were chosen because they are close to

Table 2 Correlation coefficient ρ for the best-fit models in the experiments with OneMax shown in Fig. 1a, b

Setting	Model	ρ
$\mu = \sqrt{n}$	$5.8297\ n \log n$	0.9968
	$0.8104\ n^{3/2}$	0.9996
	$0.0133\ n^2$	0.9910
$\mu = \sqrt{n} \log n$	$7.7544\ n \log n$	0.9974
	$1.0767\ n^{3/2}$	0.9995
	$0.0177\ n^2$	0.9903

Table 3 Correlation coefficient ρ for the best-fit models in the experiments with LeadingOnes shown in Fig. 2

Setting	Model	ρ
$\mu = \sqrt{n}$	$646.14\ n \log n$	0.9756
	$91.160\ n^{3/2}$	0.9928
	$1.5223\ n^2$	0.9999
	$0.1851\ n^2 \log n$	0.9999

the theoretical results. The correlation coefficient ρ is then calculated for each model to find the best-fit model.

In Table 2, we observe that for small parent populations (i.e. $\mu = \sqrt{n}$), model $0.8104\ n^{3/2}$ fits the empirical data best, while the quadratic model gives the worst result. For larger parent population (i.e. $\mu = \sqrt{n} \log n$), the model $1.0767\ n^{3/2}$ fits best the empirical data among the three models. Since $0.8104\ n^{3/2} \in \mathcal{O}(n^2)$, these findings are consistent with the theoretical expected optimisation time and may further suggest that the quadratic bound in case of small population is not tight.

5.2 LeadingOnes

We conducted experiments with $\mu = \sqrt{n}$, and $\lambda = n$. According to Theorem 7, the upper bound of the expected runtime is in this case $\mathcal{O}(n\lambda \log \lambda + n^2) = \mathcal{O}(n^2 \log n)$. Figure 2 shows the empirical runtime. Similarly to the OneMax problem, we fit the empirical runtime with four different models—$c_1 n \log n$, $c_2 n^{3/2}$, $c_3 n^2$ and $c_4 n^2 \log n$—using non-linear regression. The best values of the four constants are shown in Table 3 along with the correlation coefficients of the models.

Figure 2 and Table 3 show that both the model $1.5223\ n^2$ and the model $0.1851\ n^2 \log n$, having the same correlation coefficient, fit well with the empirical data (i.e. the empirical data lie between these two curves). This finding is consistent with the theoretical runtime bound $\mathcal{O}(n^2 \log n)$. Note also that these two models differ asymptotically by $\Theta(\log n)$, suggesting that our analysis of the UMDA on LeadingOnes is nearly tight.

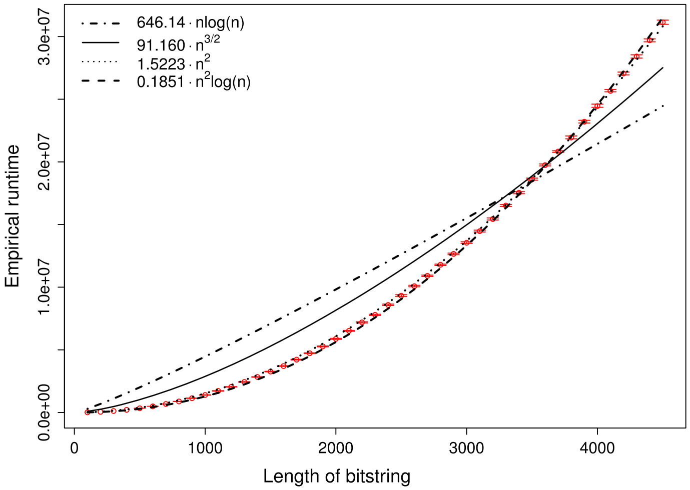

Fig. 2 Average runtime of the UMDA on LEADINGONES with 95% confidence intervals plotted with error bars in red colour. Models are also fitted via non-linear regression (Color figure online)

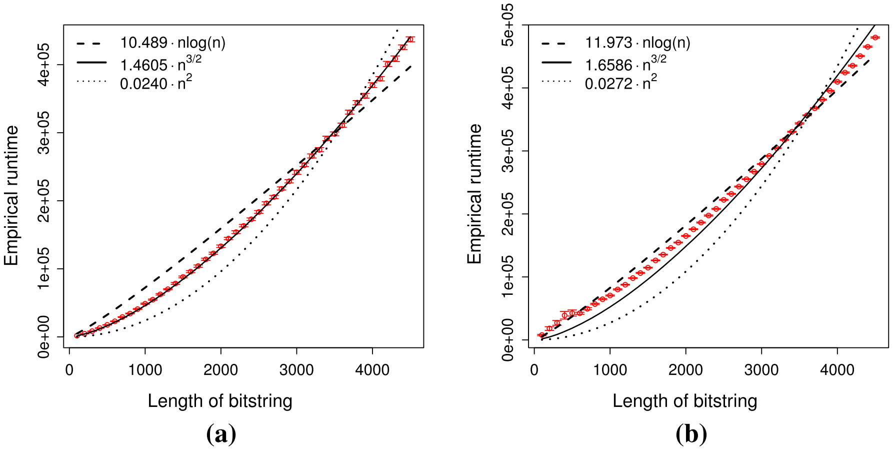

Fig. 3 Average runtime of the UMDA on BINVAL with 95% confidence intervals plotted with error bars in red colour. Models are also fitted via non-linear regression. **a** Small μ. **b** Large μ (Color figure online)

5.3 BinVal

Finally, we consider BINVAL. The upper bound $\mathcal{O}(n\lambda \log \lambda + n^2)$ from Theorem 7 for the function is identical to the bound for LEADINGONES. Since BINVAL is also a linear function like ONEMAX, we decided to set the experiments similarly for these functions, i.e. with different parent populations $\mu = \sqrt{n}$ and $\mu = \sqrt{n} \log n$. The empirical results are shown in Fig. 3. Again the empirical runtime is fitted to the three models $c_1 n \log n$, $c_2 n^{3/2}$ and $c_3 n^2$. The best values of c_1, c_2 and c_3 are listed in Table 4, along with the correlation coefficient for each model.

Table 4 Correlation coefficient ρ for the best-fit models in the experiments with BinVal shown in Fig. 3a, b

Setting	Model	ρ
$\mu = \sqrt{n}$	$10.489\, n \log n$	0.9952
	$1.4605\, n^{3/2}$	0.9999
	$0.0240\, n^2$	0.9933
$\mu = \sqrt{n} \log n$	$11.973\, n \log n$	0.9972
	$1.6596\, n^{3/2}$	0.9994
	$0.0272\, n^2$	0.9903

Theorem 7 gives the upper bound of $\mathcal{O}(n^2 \log n)$ for the expected runtime of BinVal. However, Fig. 3 and Table 4 show clearly that the model $1.4605\ n^{3/2}$ fits best the empirical runtime for $\mu = \sqrt{n}$. On the other hand, the empirical runtime lies between the two models $11.973\ n \log n$ and $1.6586\ n^{3/2}$ when $\mu = \sqrt{n} \log n$. While these observations are consistent with the theoretical upper bound since $\mathcal{O}(n^{3/2})$ and $\mathcal{O}(n \log n)$ are all members of $\mathcal{O}(n^2 \log n)$, they also suggest that our analysis of the UMDA on BinVal given by Theorem 7 may be loose.

6 Conclusion

Despite the popularity of EDAs in real-world applications, little has been known about their optimisation time, even for apparently simple settings such as the UMDA on toy functions. More results for the UMDA on these simple problems with well-understood structures provide a way to describe and compare the performance of the algorithm with other search heuristics. Furthermore, results about the UMDA are not only relevant to evolutionary computation, but also to population genetics where it corresponds to the notion of *linkage equilibrium* [36,41].

We have analysed the expected optimisation time of the UMDA on three benchmark problems: OneMax, LeadingOnes and BinVal. For both LeadingOnes and BinVal, we proved the upper bound $\mathcal{O}(n\lambda \log \lambda + n^2)$, which holds for $\lambda = \Omega(\log n)$. For OneMax, two upper bounds of $\mathcal{O}(\lambda n)$ and $\mathcal{O}(\lambda\sqrt{n})$ were obtained for $\mu = \Omega(\log n) \cap \mathcal{O}(\sqrt{n})$ and $\mu = \Omega(\sqrt{n} \log n)$, respectively. Although our result assumes that $\lambda \geq (1 + \beta)\mu$ for some positive constant $\beta > 0$, it no longer requires that $\lambda = \Theta(\mu)$ as in [44]. Note that if $\lambda = \Theta(\log n)$, a tight bound $\Theta(n \log n)$ on the expected optimisation time of the UMDA on OneMax is obtained, matching the well-known tight bound $\Theta(n \log n)$ for the (1 + 1) EA on the class of linear functions. Although we did not obtain a runtime bound when the parent population size is $\mu = \Omega(\sqrt{n}) \cap \mathcal{O}(\sqrt{n} \log n)$, our results finally close the existing $\Theta(\log \log n)$-gap between the first upper bound $\mathcal{O}(n \log n \log \log n)$ for $\lambda = \Omega(\mu)$ [10] and the relatively new lower bound $\Omega(\mu\sqrt{n} + n \log n)$ for $\lambda = (1 + \Theta(1))\mu$ [26].

Our analysis further demonstrates that the level-based theorem can yield, relatively easily, asymptotically tight upper bounds for non-trivial, population-based algorithms. An important additional component of the analysis was the use of anti-concentration properties of the Poisson–Binomial distribution. 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. We expect that similar arguments will lead to new results in runtime analysis of evolutionary algorithms.

Acknowledgements Funding was provided by Seventh Framework Programme (Grant No. 618091).

Appendix

Lemma 10 ([35]) *For all $t \in \mathbb{R}$ and $n \in \mathbb{R}^+$,*

$$\left(1+\frac{t}{n}\right)^n \le e^t \le \left(1+\frac{t}{n}\right)^{n+t/2}.$$

Lemma 11 (Theorem 3.2, [23]) *Let $Y_1, Y_2, \ldots, Y_n$ be n independent Bernoulli random variables, and $Y := \sum_{i=1}^n Y_i$ is the sum of these random variables. If $\mathbb{E}[Y]$ is an integer, then*

$$\Pr\left(Y \ge \mathbb{E}[Y]\right) \ge 1/2.$$

Lemma 12 (Stirling's approximation [31]) *For all $n \in \mathbb{N}$,*

$$n! = \Theta\left(\frac{n^{n+1/2}}{e^n}\right).$$

Lemma 13 (Lemma 8.4, [12]) *Let $X_1, \ldots, X_n$ be independent random variables defined over some common probability space. Let $Y_1, \ldots, Y_n$ be independent random variables defined over a possibly different probability space. If $X_i \preceq Y_i$ for all $i \in \{1, 2, \ldots, n\}$, then $\sum_{i=1}^n X_i \preceq \sum_{i=1}^n Y_i$.*

Lemma 14 (Lemma 3, [46]) *Let $Y_1, Y_2, \ldots, Y_n$ be n independent Bernoulli random variables with success probabilities $p_1, p_2, \ldots, p_n$. Let $Y := \sum_{i=1}^n Y_i$ be the sum of these variables. If $p_i \ge p_{\min}$ for all $i \in [n]$, where $p_{\min} > 0$ is a constant, and any constant $d^* \ge 1/p_{\min}$ then*

$$\Pr\left(Y \ge \min\left\{\mathbb{E}\left[Y\right] + d^*\sqrt{n - \lfloor\mathbb{E}[Y]\rfloor}, n\right\}\right) \ge \kappa,$$

where κ is a positive constant independent of n.

Lemma 15 *For any $n \in \mathbb{N}$, any constant $d \in (0, 1]$ independent to n and the sequence $(f_i)_{i\in\mathbb{N}}$ defined according to (10), it holds that*

(i) $f_i \leq n$ *for all* $i \in \mathbb{N}$*, and* $\exists j \in \mathbb{N}\colon\ f_j = n$*,*
(ii) if $\ell = \min\{i \in \mathbb{N} \mid f_i = n\}$ *then* $\ell = \Theta(\sqrt{n})$*.*

Proof We first prove (i), it is easy to see that f_i are all integer, 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, $\exists k \geq 0\colon\ f_k \leq n-1$ 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 $f_k < n - 1$ 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 hit n exactly at one point in time then it will remain at that value.

To bound ℓ in (ii), we pair $(f_i)_{i\in\mathbb{N}}$ with $(r_i := \sqrt{n - f_i})_{i\in\mathbb{N}}$; thus, this sequence starts at $r_0 = \sqrt{n}$, then decreases and eventually hits 0, that is, $\sqrt{n} = r_0 > r_1 > r_2 > \cdots > r_{\ell-1} > r_\ell = 0$. From (10), we have

$$(r_i - r_{i+1})(r_i + r_{i+1}) = r_i^2 - r_{i+1}^2 = f_{i+1} - f_i = \lceil dr_i \rceil,$$

note that $1 + dr_i > \lceil dr_i \rceil \geq dr_i$, then for $i \leq \ell - 1$, we can divide both sides by $r_i + r_{i+1} > 0$ to get

$$\frac{1 + dr_i}{r_i + r_{i+1}} > r_i - r_{i+1} \geq \frac{dr_i}{r_i + r_{i+1}}.$$

Always restricted to $i \leq \ell - 1$, we have that $1 > r_{i+1}/r_i \geq 0$, and therefore $dr_i/(r_i + r_{i+1}) = d/(1 + r_{i+1}/r_i) > d/2$. In addition, $f_i \leq n - 1$ then $r_i = \sqrt{n - f_i} \geq 1$ or $1/r_i \leq 1$, so $(1 + dr_i)/(r_i + r_{i+1}) = (1/r_i + d)/(1 + r_{i+1}/r_i) \leq d + 1$. Therefore, for all $i \leq \ell - 1$

$$d + 1 > r_i - r_{i+1} > \frac{d}{2}.$$

Summing all these terms gives that

$$\ell(d + 1) > \sum_{i=0}^{\ell-1}(r_i - r_{i+1}) = r_0 - r_\ell = \sqrt{n} > \frac{\ell d}{2},$$

and this implies $2\sqrt{n}/d > \ell > \sqrt{n}/(d + 1)$, or $\ell = \Theta(\sqrt{n})$. □

Lemma 16 *Let* $Y_1, Y_2, \ldots, Y_k$ *be* k *(*$k \geq 1$*) independent Bernoulli random variables with success probabilities* $p_1, p_2, \ldots, p_k$*, where* $p_i \geq p_{\min} = 1/4$ *for each* $i \in [k]$*. Let* $Y_{1,k} := \sum_{i=1}^{k} Y_i$*. Then we always have*

$$\Pr\left(Y_{1,k} \geq \mathbb{E}\left[Y_{1,k}\right]\right) \geq \Omega(1).$$

Proof We start by considering small values of k. If $k = 1$, then

$$\Pr\left(Y_{1,1} \geq \mathbb{E}\left[Y_{1,1}\right]\right) = \Pr(Y_1 = 1) = p_1 \geq 1/4.$$

If $k = 2$, then

$$\Pr\left(Y_{1,2} \geq \mathbb{E}[Y_{1,2}]\right) \geq \Pr\left(Y_1 = 1\right) \cdot \Pr\left(Y_2 = 1\right) \geq p_1 p_2 \geq (1/4)^2.$$

For larger values of k, following [46] we introduce another random variable $Z = (Z_1, \ldots, Z_k)$ with success probabilities $z_1, \ldots, z_k$, where $z_i \geq p_{\min}$ and $\mathbb{E}[Z_{1,k}] = \sum_{i=1}^{k} z_i = \sum_{i=1}^{k} p_i = \mathbb{E}\left[Y_{1,k}\right]$. However, we shift the total weight $\mathbb{E}\left[Y_{1,k}\right]$ as far as possible to the Z_i with smaller indices as follows. We define $m = \lfloor \frac{\mathbb{E}[Y_{1,k}] - kp_{\min}}{1 - p_{\min}} \rfloor$, and let $Z_1, \ldots, Z_m$ all get success probability 1, and $Z_{m+2}, \ldots, Z_k$ get $z_i = p_{\min}$, more precisely

$$z_i = \begin{cases} 1, & \text{for } i = 1, \ldots, m, \\ q, & \text{for } i = m+1, \\ p_{\min}, & \text{for } i = m+2, \ldots, k, \end{cases}$$

where $q \in [p_{\min}, 1]$. It is quite clear that $(z_1, \ldots, z_k)$ majorises $(p_t(1), \ldots, p_t(k))$. From [19,33], we now have

$$\Pr\left(Y_{1,k} \geq \mathbb{E}\left[Y_{1,k}\right]\right) \geq \Pr\left(Y_{1,k} \geq \mathbb{E}\left[Y_{1,k}\right] + 1\right) \geq \Pr(Z_{1,k} \geq \mathbb{E}\left[Z_{1,k}\right] + 1).$$

Furthermore, with probability 1 we can get m ones and

$$\mathbb{E}[Z_{m+2,k}] = \mathbb{E}\left[Z_{1,k}\right] - m - q \iff \mathbb{E}[Z_{m+2,k}] + q = \mathbb{E}\left[Z_{1,k}\right] - m,$$

then

$$\begin{aligned} \Pr(Z_{1,k} \geq \mathbb{E}\left[Z_{1,k}\right] + 1) &\geq \Pr(Z_{m+1,k} \geq \mathbb{E}\left[Z_{1,k}\right] + 1 - m) \\ &\geq \Pr(Z_{m+1} = 1) \cdot \Pr(Z_{m+2,k} \geq \mathbb{E}\left[Z_{1,k}\right] - m) \\ &= q \cdot \Pr(Z_{m+2,k} \geq \mathbb{E}[Z_{m+2,k}] + q) \\ &\geq p_{\min} \cdot \Pr(Z_{m+2,k} \geq \mathbb{E}[Z_{m+2,k}] + 1). \end{aligned}$$

The last inequality follows the fact that $p_{\min} \leq q \leq 1$. We now need a lower bound on the probability $\Pr(Z_{m+2,k} \geq \mathbb{E}[Z_{m+2,k}] + 1)$, where

$$Z_{m+2,k} \sim \text{Bin}\left(k - m - 1, \frac{1}{4}\right).$$

Now let $k - m - 1 = 4t + x = 4(t-1) + x + 4$, where $t \in \mathbb{N}$ and $x \in \{0, 1, 2, 3\}$. Then $\mathbb{E}[Z_{m+2,k}] = t + \frac{x}{4}$, and

$$\begin{aligned}
&\Pr(Z_{m+2,k} \geq \mathbb{E}[Z_{m+2,k}] + 1) \\
&\quad = \Pr(Z_{m+2,k} \geq t + \frac{x}{4} + 1) \\
&\quad \geq \Pr\left(Z_{m+2,k} \geq 4(t-1) + x + 4\right) \\
&\quad \geq \Pr(Z_{m+2,m+2+4(t-1)-1} \geq t-1) \cdot \Pr(Z_{m+2+4(t-1),n} \geq x+4) \\
&\quad = \Pr(Z_{m+2,m+2+4(t-1)-1} \geq \mathbb{E}[Z_{m+2,m+2+4(t-1)-1}]) \cdot \Pr(Z_{m+2+4(t-1),n} \geq x+4) \\
&\quad \geq \frac{1}{2} \cdot \left(\frac{1}{4}\right)^{x+4} \geq \frac{1}{2} \cdot \left(\frac{1}{4}\right)^{7}.
\end{aligned}$$

The result follows the result of Lemma 11, where $\mathbb{E}[Z_{m+2,m+2+4(t-1)-1}]$ is an integer, and $x \leq 3$. This proves the Lemma. □

We note that a similar result (without a specific value for the constant) can be found in [12, Lemma 10.16].

Lemma 17 *For any constant $d \leq 1$ and $\mathbb{E}[Y_{1,n}] \geq f_{j-1} - \ell/n$, it holds that*

$$\mathbb{E}\left[Y_{1,n}\right] + d\sqrt{n - \mathbb{E}\left[Y_{1,n}\right]} \geq f_{j-1} + d\sqrt{n - f_{j-1}}. \tag{15}$$

Proof Let us rewrite (14) by introducing a variable $x \geq 0$ as follows:

$$\mathbb{E}\left[Y_{1,n}\right] = f_{j-1} - \frac{\ell}{n} + x \tag{16}$$

We consider two different cases.

- **Case 1** If $x = \ell/n$, then $\mathbb{E}\left[Y_{1,n}\right] = f_{j-1}$, and the lemma holds for all values of d.
- **Case 2** If $x \neq \ell/n$, then substituting (16) into (15) and let $y := x - \ell/n \in [-\ell/n, 0) \cup (0, n - f_{j-1}]$, we have

$$d \leq g\left(y, f_{j-1}\right) := \frac{y}{\sqrt{n - f_{j-1}} - \sqrt{n - f_{j-1} - y}}$$

This always holds if we pick a constant $d \leq \min_{y, f_{j-1}} g\left(y, f_{j-1}\right)$. From $\partial g/\partial y = 0$, we obtain $y = 0$. Note that when $y = 0$, $\partial^2 g/\partial y^2 < 0$. This means $g(y, f_{j-1})$ reaches the maximum value when $y = 0$ with respect to f_{j-1}, and

$$\begin{aligned} g\left(y, f_{j-1}\right) &\geq \min\left\{g\left(-\ell/n, f_{j-1}\right), g\left(n - f_{j-1}, f_{j-1}\right)\right\} \\ &= \min\left\{\sqrt{n - f_{j-1}}, \sqrt{n - f_{j-1}} + \sqrt{n - f_{j-1} + \ell/n}\right\} \\ &= \sqrt{n - f_{j-1}} \\ &\geq \min_{f_{j-1}}\left\{\sqrt{n - f_{j-1}}\right\} \\ &= 1 \end{aligned}$$

due to $f_{j-1} \leq n - 1$.

The lemma is proved by combining results of the two cases. □

References

1. Armañanzas, R., Inza, I., Santana, R., Saeys, Y., Flores, J.L., Lozano, J.A., Van de Peer, Y., Blanco, R., Robles, V., Bielza, C., Larrañaga, P.: A review of estimation of distribution algorithms in bioinformatics. BioData Min **1**(1), 6 (2008)
2. Asoh, H., Mühlenbein, H.: On the mean convergence time of evolutionary algorithms without selection and mutation. In: Proceedings of the 3rd International Conference on Parallel Problem Solving from Nature, PPSN III, pp. 88–97 (1994)
3. Baillon, J.-B., Cominetti, R., Vaisman, J.: A sharp uniform bound for the distribution of sums of Bernoulli trials. Comb. Probab. Comput. **25**(3), 352–361 (2016)
4. Baluja, S.: Population-based incremental learning: a method for integrating genetic search based function optimization and competitive learning. Technical Report, Carnegie Mellon University (1994)
5. Chen, T., Lehre, P.K., Tang, K., Yao, X.: When is an estimation of distribution algorithm better than an evolutionary algorithm? In: Proceedings of 2009 IEEE Congress on Evolutionary Computation, pp. 1470–1477 (2009)
6. Chen, T., Tang, K., Chen, G., Yao, X.: On the analysis of average time complexity of estimation of distribution algorithms. In: Proceedings of 2007 IEEE Congress on Evolutionary Computation, pp. 453–460 (2007)
7. Chen, T., Tang, K., Chen, G., Yao, X.: Rigorous time complexity analysis of univariate marginal distribution algorithm with margins. In: Proceedings of 2009 IEEE Congress on Evolutionary Computation, pp. 2157–2164 (2009)
8. Chen, T., Tang, K., Chen, G., Yao, X.: Analysis of computational time of simple estimation of distribution algorithms. IEEE Trans. Evol. Comput. **14**(1), 1–22 (2010)
9. Corus, D., Dang, D.-C., Eremeev, A.V., Lehre P.K.: Level-based analysis of genetic algorithms and other search processes. IEEE Trans. Evol. Comput. https://doi.org/10.1109/TEVC.2017.2753538 (2017)
10. Dang D.-C., Lehre P.K.: Simplified runtime analysis of estimation of distribution algorithms. In: Proceedings of Genetic and Evolutionary Computation, GECCO'15, pp. 513–518 (2015)
11. Dang, D.-C., Lehre, P.K.: Self-adaptation of mutation rates in non-elitist populations. In: Proceedings of the 14th International Conference on Parallel Problem Solving from Nature, PPSN XIV, pp. 803–813 (2016)
12. Doerr, B.: Probabilistic tools for the analysis of randomized optimization heuristics. CoRR. arXiv:1801.06733 (2018)
13. Doerr, B., Krejca, M.S.: Significance-based estimation-of-distribution algorithms. In: Proceedings of Genetic and Evolutionary Computation Conference, GECCO'18, pp. 1483–1490 (2018)
14. Droste, S.: A rigorous analysis of the compact genetic algorithm for linear functions. Natl. Comput. **5**(3), 257–283 (2006)
15. Ducheyne, E.I., De Baets, B., De Wulf, R.: Probabilistic Models for Linkage Learning in Forest Management, pp. 177–194. Springer, Berlin (2005)
16. Feige, U.: On sums of independent random variables with unbounded variance and estimating the average degree in a graph. SIAM J. Comput. **35**(4), 964–984 (2006)
17. Friedrich, T., Kötzing, T., Krejca, M., Sutton, A.M.: The compact genetic algorithm is efficient under extreme Gaussian noise. IEEE Trans. Evol. Comput. **21**(3), 477–490 (2017)

18. Friedrich, T., Kötzing, T., Krejca, M.S.: EDAs cannot be balanced and stable. In: Proceedings of Genetic and Evolutionary Computation Conference, GECCO'16, pp. 1139–1146 (2016)
19. Gleser, L.J.: On the distribution of the number of successes in independent trials. Ann. Probab. **3**(1), 182–188 (1975)
20. Gu, W., Wu, Y., Zhang, G.Y.: A hybrid univariate marginal distribution algorithm for dynamic economic dispatch of units considering valve-point effects and ramp rates. Int. Trans. Electr. Energy Syst. **25**(2), 374–392 (2015)
21. Harik, G.R., Lobo, F.G., Goldberg, D.E.: The compact genetic algorithm. IEEE Trans. Evol. Comput. **3**(4), 287–297 (1999)
22. Hauschild, M., Pelikan, M.: An introduction and survey of estimation of distribution algorithms. Swarm Evol. Comput. **1**(3), 111–128 (2011)
23. Jogdeo, K., Samuels, S.M.: Monotone convergence of binomial probabilities and a generalization of ramanujan's equation. Ann. Math. Stat. **39**(4), 1191–1195 (1968)
24. Kollat, J.B., Reed, P.M., Kasprzyk, J.R.: A new epsilon-dominance hierarchical Bayesian optimization algorithm for large multiobjective monitoring network design problems. Adv. Water Resour. **31**(5), 828–845 (2008)
25. Krejca, M.S., Witt, C.: Theory of estimation-of-distribution algorithms. CoRR. arXiv:1806.05392 (2018)
26. Krejca, M.S., Witt, C.: Lower bounds on the run time of the univariate marginal distribution algorithm on OneMax. In: Proceedings of Foundations of Genetic Algorithms XIV, FOGA'17, pp. 65–79 (2017)
27. Lehre, P.K., Nguyen, P.T.H.: Improved runtime bounds for the univariate marginal distribution algorithm via anti-concentration. In: Proceedings of Genetic and Evolutionary Computation Conference, GECCO'17, pp. 1383–1390 (2017)
28. Lehre, P.K., Nguyen, P.T.H.: Level-based analysis of the population-based incremental learning algorithm. In: Proceedings of the 15th International Conference on Parallel Problem Solving from Nature, PPSN XV, pp. 105–116 (2018)
29. Lehre, P.K., Witt, C.: Black-box search by unbiased variation. In: Proceedings of Genetic and Evolutionary Computation Conference, GECCO'10, pp. 1441–1448 (2010)
30. Lehre, P.K., Yao, X.: Runtime analysis of the (1+1) EA on computing unique input output sequences. Inf. Sci. **259**, 510–531 (2014)
31. Leiserson, C.E., Stein, C., Rivest, R., Cormen, T.H.: Introduction to Algorithms. MIT Press, Cambridge (2009)
32. Lengler, J., Sudholt, D., Witt, C.: Medium step sizes are harmful for the compact genetic algorithm. In: Proceedings of Genetic and Evolutionary Computation Conference, GECCO'18, pp. 1499–1506 (2018)
33. Marshall, A.W., Olkin, I., Arnold, B.C.: Inequalities: Theory of Majorization and its Applications. Springer, New York (2011)
34. Massart, P.: The tight constant in the Dvoretzky–Kiefer–Wolfowitz inequality. Ann. Probab. **18**(3), 1269–1283 (1990)
35. Mitrinovic, D.S.: Analytic Inequalities. Springer, Berlin (1970)
36. Mühlenbein, H., Mahnig, T.: Evolutionary computation and wright's equation. Theor. Comput. Sci. **287**, 145–165 (2002)
37. Mühlenbein, H., Paaß, G.: From recombination of genes to the estimation of distributions I. Binary parameters. In: Proceedings of the 9th International Conference on Parallel Problem Solving from Nature, PPSN IV, pp. 178–187 (1996)
38. Rubinstein, R.Y., Kroese, D.P.: The Cross Entropy Method: A Unified Approach To Combinatorial Optimization, Monte–Carlo Simulation (Information Science and Statistics). Springer, New York (2004)
39. Santana, R., Mendiburu, A., Lozano, J.A.: A review of message passing algorithms in estimation of distribution algorithms. Natl. Comput. **15**(1), 165–180 (2016)
40. Shapiro, J.L.: Drift and scaling in estimation of distribution algorithms. Evol. Comput. **13**(1), 99–123 (2005)
41. Slatkin, M.: Linkage disequilibrium—understanding the evolutionary past and mapping the medical future. Nat. Rev. Genet. **9**(6), 477–485 (2008)
42. Sudholt, D., Witt, C.: Update strength in EDAs and ACO: how to avoid genetic drift. In: Proceedings of Genetic and Evolutionary Computation Conference, GECCO'16, pp. 61–68 (2016)
43. van der Waerden, B.L.: Algebra, vol. 1. Springer, New York (1991)

44. Witt, C.: Upper bounds on the runtime of the univariate marginal distribution algorithm on OneMax. In: Proceedings of Genetic and Evolutionary Computation Conference, GECCO'17, pp. 1415–1422 (2017)
45. Witt, C.: Domino convergence: why one should hill-climb on linear functions. In: Proceedings of Genetic and Evolutionary Computation Conference, GECCO'18, pp. 1539–1546 (2018)
46. Wu, Z., Kolonko, M., Möhring, R.H.: Stochastic runtime analysis of the cross-entropy algorithm. IEEE Trans. Evol. Comput. **21**(4), 616–628 (2017)
47. Yu, T.-L., Santarelli, S., Goldberg, D.E.: Military Antenna Design Using a Simple Genetic Algorithm and hBOA, pp. 275–289. Springer, Berlin (2006)
48. Zinchenko, L., Mühlenbein, H., Kureichik, V., Mahnig, T.: Application of the univariate marginal distribution algorithm to analog circuit design. In: Proceedings of 2002 NASA/DoD Conference on Evolvable Hardware, pp. 93–101 (2002)