Parallel Black-Box Complexity with Tail Bounds

Per Kristian Lehre, Dirk Sudholt

***Abstract*—We propose a new black-box complexity model for search algorithms evaluating λ search points in parallel. The parallel unary unbiased black-box complexity gives lower bounds on the number of function evaluations *every* parallel unary unbiased black-box algorithm needs to optimise a given problem. It captures the inertia caused by offspring populations in evolutionary algorithms and the total computational effort in parallel metaheuristics[1]**

We present complexity results for LeadingOnes and OneMax. Our main result is a general performance limit: we prove that on *every* function *every* λ-parallel unary unbiased algorithm needs at least a certain number of evaluations (a function of problem size and λ) to find any desired target set of up to exponential size, with an overwhelming probability. This yields lower bounds for the typical optimisation time on unimodal and multimodal problems, for the time to find any *local* optimum, and for the time to even get close to any optimum. The power and versatility of this approach is shown for a wide range of illustrative problems from combinatorial optimisation. Our performance limits can guide parameter choice and algorithm design; we demonstrate the latter by presenting an optimal λ-parallel algorithm for OneMax that uses parallelism most effectively.

I. Introduction

BLACK-BOX optimisation describes a challenging realm of problems where no algebraic model or gradient information is available. The problem is regarded a black box, and knowledge about the problem in hand can only be obtained by evaluating candidate solutions. General-purpose metaheuristics like evolutionary algorithms, simulated annealing, ant colony optimisers, tabu search, and particle swarm optimisers are well suited for black-box optimisation as they generally work well without any problem-dependent knowledge.

A lot of research has focussed on designing powerful metaheuristics, yet it is often unclear which search paradigm works best for a particular problem class, and whether and how better performance can be obtained by tailoring a search paradigm to the problem class in hand.

Black-box complexity is a powerful tool that describes limits on the efficiency of black-box algorithms. The black-box complexity of search algorithms captures the difficulty of problem classes in black-box optimisation. It describes the minimum number of function evaluations that *every* black-box algorithm needs to make to optimise a problem from a given class. It provides a rigorous theoretical foundation through capturing limits to the efficiency of all black-box search algorithms, providing a baseline for performance comparisons across all known and future metaheuristics as well as tailored black-box algorithms. Also it prevents algorithm designers from wasting effort on trying to achieve impossible performance.

Many different models of black-box complexities have been developed. The first black-box complexity model by Droste, Jansen, and Wegener [28] makes no restriction on the black-box algorithm. This leads to some unrealistic results, such as polynomial black-box complexities of NP-hard problems [28]. Subsequent research introduced refined models that restrict the power of black-box algorithms, leading to more realistic results [18], [20], [21], [28], [57], where black-box algorithms can only query for the relative order of function values of search points [20], [57] as well as memory restrictions [21], [28] and restrictions on which search points are allowed to be stored [23]–[25]. Lehre and Witt [45] introduced the unbiased black-box model where black-box algorithms may only use operators without a search bias (see Section II). This model initially considered unary operators (such as mutation) and was later extended to higher arity operators (such as crossover) [16] and more general search spaces [53]. It also led to the discovery of more efficient EA variants [11]. For further details we refer to the comprehensive survey by Doerr [22].

A shortcoming of the above models is that they do not capture the implicit or explicit parallelism at the heart of many common search algorithms. Evolutionary algorithms (EAs) such as (μ+λ) EAs or (μ,λ) EAs generate λ offspring in parallel. Using a large offspring population in many cases can decrease the number of generations needed to find an optimal solution[2]. However, the number of function evaluations may increase as evolution can only act on information from the previous generation. A large offspring population can lead to inertia that slows down the optimisation process. Existing black-box models are unable to capture this inertia as they assume all search points being created in sequence.

The same goes for parallel metaheuristics such as island models evolving multiple populations in parallel (see, e. g. Luque and Alba [47]). Parallelisation can decrease the number of generations, or parallel time. But the overall computational effort, the number of function evaluations across all islands, may increase. Lässig and Sudholt [44] used the following notion. Let T_λ be the random number of generations an island model with λ islands (each creating one offspring) needed to find a global optimum for a given problem. If using λ islands can decrease the parallel time by a factor of order λ, compared to just one island, $\lambda \cdot \mathrm{E}(T_\lambda) = O(\mathrm{E}(T_1))$, this is called a *linear speedup* (with regards to the parallel time, the number of generations). In other words, a linear speedups means that the total number of function evaluations, $\lambda \cdot \mathrm{E}(T_\lambda)$, does not increase beyond a constant factor.

P. K. Lehre is with the School of Computer Science, University of Birmingham, United Kingdom.

D. Sudholt is with the Department of Computer Science, University of Sheffield, United Kingdom.

[1]This paper significantly extends preliminary results which appeared in [1].

[2]This does not hold for all problems; Jansen, De Jong, and Wegener [38] constructed problems where offspring populations drastically increase the number of generations.

Previous work [43], [44], [48] considered illustrative problems from pseudo-Boolean optimisation and combinatorial optimisation, showing sufficient conditions for linear speedups. However, the absence of matching lower bounds makes it impossible to determine exactly for which parameters λ linear speedups are achieved.

We provide a parallel black-box model that captures and quantifies the inertia caused by offspring populations of size λ and parallel EAs evaluating λ search points in parallel. We present lower bounds on the black-box complexity for the well known LO (LEADINGONES) problem and for the general class of functions with a unique optimum, revealing how the number of function evaluations increases with the problem size n and the degree of parallelism, λ. The results complement existing upper bounds [44], allowing us to characterise the realm of linear speedups, where parallelisation is effective.

Our lower bound for functions with a unique optimum is asymptotically tight: for the ONEMAX problem, we present a simple (1+λ) EA with an adaptive mutation rate that achieves an asymptotically optimal performance amongst all parallel unary unbiased black-box algorithms. Our adaptive mutation rates decrease the expected running time by a factor of order $\ln \ln \lambda$, compared to the (1+λ) EA with the standard mutation rate $1/n$ [17].

The paper extends a previous conference paper [1] with parts of the results. A major novelty in this manuscript is the introduction of black-box complexity results with tail bounds. Existing black-box complexity results only make statements about the *expected* number of evaluations it takes to find a global optimum[3]. However, it is often not clear whether the expectation is a good reflection of the performance observed in practice. We provide black-box complexity lower bounds that apply with an overwhelming probability. More precisely, using the notation $\ln^+ x := \max(1, \ln x)$ whenever the argument can be smaller than the logarithm's base[4], we show for every target search point x^* we can choose that *every* λ-parallel unary unbiased black-box algorithm needs at least

$$\max\left\{\frac{c\lambda n}{\ln^+ \lambda}, (1-\delta) n \ln n\right\} = \Omega\left(\frac{\lambda n}{\ln^+ \lambda} + n \ln n\right) \quad (1)$$

function evaluations to find x^*, with an overwhelming probability[5], where c is a constant with $c \geq 1/60$. The leading constant $1-\delta$ in the $n \ln n$ term can be chosen[6] arbitrarily close to 1. This means that it is practically impossible for any unary unbiased black-box algorithm to find a designated target with less than $\frac{c\lambda n}{\ln^+ \lambda}$ or less than $(1-\delta) n \ln n$ evaluations. The latter bound applies to parallel and non-parallel unary unbiased algorithms.

In addition, if the probability of finding a single target x^* in the stated time is exponentially small, the probability of finding *many* target points is still exponentially small. This simple union bound argument opens up a range of opportunities for obtaining stronger results that are much more relevant to practice than the state-of-the-art. Our method is powerful and versatile since we can choose any set of target search points, up to an exponential size. This allows for different applications.

1) Considering global optimisation, our lower bound (1) applies to highly multimodal functions, even allowing for up to exponentially many optima. Apart from results tailored to specific problem classes [18], the only generic black-box complexity lower bounds apply to functions with one unique global optimum. Our lower bound yields a general baseline that applies to all unary unbiased black-box algorithms and a wide range of problems.
2) Choosing all local optima as target search points, we also get that for functions with up to exponentially many local optima, every λ-parallel unary unbiased algorithm needs at least the stated time (1) to find any *local* optimum.
3) Since we can have exponentially many target search points, we can even afford to consider all search points within an almost linear Hamming distance to any local optimum as target. Then our results imply that even the time to get close to any local or global optimum is bounded from below by (1).

We demonstrate the applicability and versatility of our main result by deriving the first black-box complexity lower bounds for a wide range of illustrative function classes, from synthetic problems (TWOMAX, H-IFF, JUMP$_k$, CLIFF) that are very popular in the evolutionary computation literature to classes of benchmark functions [41] and important problems from combinatorial optimisation such as VERTEX COLOURING, MINCUT, PARTITION, KNAPSACK and MAXSAT.

In addition to providing a solid unifying theoretical foundation for black-box algorithms, we believe that our results are of immediate relevance to practice. Our black-box complexity with tail bounds gives hard limits on the capabilities of (unary unbiased) black-box algorithms. These limits can be used to set stopping criteria appropriately, avoiding stopping an algorithm before it has had a chance to come close to local or global optima. They are useful to set parameters such as the offspring population size λ: if we have a limited computational budget of T evaluations, (1) implies that we must choose λ satisfying $\lambda / \ln^+ \lambda \leq T/(cn)$ as for larger values T is lower than (1), meaning that every λ-parallel unary unbiased black-box algorithm fails badly with overwhelming probability. Moreover, our lower bounds can serve as baseline in performance comparisons across various algorithms. And, last but not least, knowing what is *impossible* is vital for guiding the search for the *best possible* algorithm. The feasibility of this approach is demonstrated in this work as we present an optimal λ-parallel algorithm for ONEMAX that uses parallelism most effectively.

II. A PARALLEL BLACK-BOX MODEL

Following Lehre and Witt [45], we only use unary unbiased variation operators, i. e., operators creating a new search point out of one search point. This includes local search, mutation in evolutionary algorithms, but it does not include recombination.

[3]A notable exception is the p-Monte Carlo runtime introduced by Doerr and Lengler [23], defined as the minimum number of steps needed in order to find an optimum with probability at least $1-p$.

[4]When a logarithm appears in an asymptotic formula, we may assume that n is large enough to make $\ln(n) = \ln^+(n)$. The same holds for terms of $\ln \ln n$, $\ln \ln n$, etc. We only use $\ln^+$ when the argument is a function of λ.

[5]An overwhelming probability is defined as $1 - 2^{-\Omega(n^\varepsilon)}$ for some constant $\varepsilon > 0$.

[6]The precise result contains a trade-off between the leading constant and the exponent of the overwhelming probability formula, see Theorem 13.

A unary variation operator can be formally described as a conditional probability distribution $p(\cdot \mid \cdot)$, where for any pairs of bitstrings $x, y \in \{0,1\}^n$, $p(y \mid x)$ is the probability that the variation operator produces an "offspring" y from the "parent" x. A unary variation operator is called *unbiased* (see Lehre and Witt [45] and Rowe and Vose [53]) if for all bitstrings $x, y, z \in \{0,1\}^n$ and permutations $\sigma : [n] \to [n]$

(1) $p(y \mid x) = p(y \oplus z \mid x \oplus z)$
(2) $p(y \mid x) = p(\sigma_b(y) \mid \sigma_b(x))$

where $\oplus$ is the xor operator, and the function $\sigma_b(x)$ is the permutation over the bit-positions, defined by

$$\sigma_b(x_1 x_2 \cdots x_n) := x_{\sigma(1)} x_{\sigma(2)} \cdots x_{\sigma(n)}.$$

Informally, unbiasedness means that there is no bias towards particular regions of the search space; unbiased operators over $\{0,1\}^n$ must treat all bit values $0, 1$ and all bit positions $1, \ldots, n$ symmetrically. This is the case for many common variation operators, such as standard bit mutation.

Throughout this paper we only deal with unbiased algorithms as the performance of biased algorithms may depend on the particular encoding used. For example, the (1+1) EA with the asymmetric mutation operator defined in Jansen and Sudholt [39] flips zeros and ones with different probabilities. This leads to improved expected times of $O(n)$ and $O(n^{3/2})$ on ONEMAX and LO, respectively, but this advantage disappears when the fitness function is transformed with operators $\oplus$ or σ_b [39]. Unbiased algorithms show the same performance on all possible transformations $\oplus$, σ_b of a fitness function.

Unbiased black-box algorithms query new search points based on the past history of function values, using unbiased variation operators. We define a *λ-parallel unbiased black-box algorithm* in the same way, with the restriction that in each round λ queries are made in parallel (see Algorithm 1). We use the abbreviation uar for *uniformly at random*. These λ queries only have access to the history of evaluations from previous rounds; they cannot access information from queries made in the same round. We refer to these λ search points as *offspring* to indicate search points created in the same round.

Algorithm 1 λ-parallel unbiased black-box algorithm

Let $t := 0$. Choose $x^1(0), \ldots, x^\lambda(0)$ uar, compute $f(x^1(0)), \ldots, f(x^\lambda(0))$, and initialise the history as $\mathcal{H} := (f(x^1(0)), \ldots, f(x^\lambda(0)))$.
repeat
  for $1 \le i \le \lambda$ **do**
    Given the history $\mathcal{H}$, choose indices $1 \le k \le \lambda$ and $0 \le j \le t$ and an unbiased variation operator p_v.
    Sample $x^i(t+1)$ according to $p_v(\cdot \mid x^k(j))$.
  for $1 \le i \le \lambda$ **do**
    Compute $f(x^i(t+1))$ and append $f(x^i(t+1))$ to $\mathcal{H}$.
  Let $t := t + 1$.
until termination condition met

This black-box model includes offspring populations in evolutionary algorithms, for example (μ+λ) EAs or (μ,λ) EAs (modulo minor differences in the initialisation). It can further model parallel evolutionary algorithms such as cellular EAs with λ cells, or island models with λ islands, each of which generates one offspring in each generation.

The (1+λ) EA maintains the current best search point x and creates λ offspring by flipping each bit in x independently with probability p (with default $p = 1/n$). The best offspring replaces its parent if it has fitness at least $f(x)$.

Algorithm 2 (1+λ) EA

Choose x uar.
repeat
  for $1 \le i \le \lambda$ **do**
    Create y_i by copying x and flipping each bit independently with probability $1/n$.
  Choose z uar from $\arg\max\{f(y_1), \ldots, f(y_\lambda)\}$.
  if $f(z) \ge f(x)$ **then** $x = z$
until termination condition met

A. Parallel black-box complexity

The *optimisation time* is commonly defined as the number of function evaluations made before a global optimum is found for the first time. The *unbiased black-box complexity (uBBC)* of a function class $\mathcal{F}$ is the minimum worst-case optimisation time among all unbiased black-box algorithms [45] (equivalent to Algorithm 1 with $\lambda = 1$). The *unbiased λ-parallel black-box complexity (λ-upBBC)* of a function class $\mathcal{F}$ is defined as the minimum worst-case number of function evaluations among all unbiased λ-parallel algorithms satisfying the framework of Algorithm 1.

With increasing λ access to previous queries becomes more and more restricted. It is therefore not surprising that the black-box complexity is non-decreasing with growing λ. For every family of function classes $\mathcal{F}_n$ and all $\lambda \in \mathbb{N}$,

$$\mathrm{uBBC}(\mathcal{F}_n) \le \lambda\text{-upBBC}(\mathcal{F}_n) \le \lambda \cdot \mathrm{uBBC}(\mathcal{F}_n) \qquad (2)$$

as any unbiased algorithm can be simulated by a λ-parallel unbiased black-box algorithm using one query in each round. Also note that the unary unbiased black-box complexity can be regarded as the 1-parallel unary unbiased black-box complexity, $\mathrm{uBBC}(\mathcal{F}_n) = 1\text{-upBBC}(\mathcal{F}_n)$.

The following lemma shows that the parallel black-box complexity increases with the degree of parallelism, modulo possible rounding issues.

Lemma 1. *For any $\alpha, \beta \in \mathbb{N}$, if $\alpha \le \beta$ then*

$$\alpha\text{-upBBC}(\mathcal{F}_n) \le \frac{\alpha}{\beta} \left\lceil \frac{\beta}{\alpha} \right\rceil \cdot \beta\text{-upBBC}(\mathcal{F}_n)$$

In particular, if $\frac{\beta}{\alpha} \in \mathbb{N}$ then α-upBBC $\le \beta$-upBBC.

A proof (in the context of distributed black-box complexity) was given in [2, Lemma 4].

Lemma 1 implies the following for all function classes $\mathcal{F}_n$ (we omit $\mathcal{F}_n$ for brevity): First, if $\frac{\beta}{\alpha} \in \mathbb{N}$ then α-upBBC $\le \beta$-upBBC. Otherwise, α-upBBC $\le (1 + \frac{\alpha}{\beta}) \cdot \beta$-upBBC $\le 2 \cdot \beta$-upBBC because $\lceil \frac{\beta}{\alpha} \rceil \le 1 + \frac{\beta}{\alpha}$ and $1 + \frac{\alpha}{\beta} \le 2$. In particular, this implies that for all $\alpha < \beta \in \mathbb{N}$,

$$\beta\text{-upBBC} = \Omega(\alpha\text{-upBBC}). \qquad (3)$$

We conclude that the λ-parallel black-box complexity does not asymptotically decrease with the degree of parallelism, $\lambda = \lambda(n)$. This implies that there is a *cut-off point* such that for all $\lambda = O(\lambda^*)$ the λ-parallel unbiased black-box complexity of $\mathcal{F}_n$ is asymptotically equal to the regular unbiased black-box complexity.[7]

Definition 2. *A value λ^* is a cut-off point if*

- *for all $\lambda = O(\lambda^*)$, λ-upBBC $= O(\text{uBBC})$ and*
- *for all $\lambda = \omega(\lambda^*)$, λ-upBBC $= \omega(\text{uBBC})$.*

Such a cut-off point always exists because due to (3) the parallel black-box complexity cannot decrease asymptotically, and values of $O(\text{uBBC})$ can always be attained for suitable λ^*, e. g. for $\lambda^* := 1$. Furthermore, the λ-parallel black-box eventually diverges for very large λ (e. g. $\lambda = \omega(\text{uBBC})$) as trivially λ-upBBC $\geq \lambda$.

Note that cut-off points are not unique: if λ^* is a cut-off point, then every $\lambda' = \Theta(\lambda^*)$ is also a cut-off point.

A cut-off point determines the realm of linear speedups [44], where parallelisation is most effective. Below the cut-off, for an optimal parallel black-box algorithm the number of function evaluations does not increase (beyond constant factors), but the number of rounds decreases by a factor of $\Theta(\lambda)$. The number of rounds corresponds to the parallel time if all λ evaluations are performed on parallel processors. Hence, below the cut-off it is possible to reduce the parallel time proportionally to the number of processors, without increasing the total computational effort (by more than a constant factor).

III. Parallel Black-Box Complexity of LeadingOnes

We consider the function $\text{LO}(x) := \sum_{i=1}^{n} \prod_{j=1}^{i} x_j$, counting the number of leading ones in x. It is an example of a unimodal function where a specific bit needs to be flipped to increase the fitness. Similarly, $\text{LZ}(x)$ counts the number of leading zeros in x. We first provide a tool for estimating the progress made by λ trials, which may or may not be independent. It is based on moment-generating functions (mgf).

Lemma 3. *Given λ random variables $X_1, \dots, X_\lambda \in \mathbb{N}$, not necessarily independent, let $X_{(\lambda)} := \max_{i \in [\lambda]} X_i$. If there exist $\eta, D \geq 0$, such that for all $i \in [\lambda]$, it holds $\text{E}\left(e^{\eta X_i}\right) \leq D$, then $\text{E}\left(X_{(\lambda)}\right) \leq (\ln(D\lambda) + 1)/\eta$.*

Proof. Note first that for any $i \in [\lambda]$ and $j \in \mathbb{N}$, it follows from Markov's inequality that $\Pr(X_i \geq j) = \Pr(e^{\eta X_i} \geq e^{\eta j}) \leq e^{-\eta j}\text{E}\left(e^{\eta X_i}\right) \leq e^{-\eta j} D$. Now, let $k := \ln(D\lambda)/\eta$. Recall that the expectation of any non-negative, integer-valued random variable N can be written as $\text{E}(N) = \sum_{i=1}^{\infty} \Pr(N \geq i)$. From this and a union bound, we get

$$\begin{aligned}\text{E}\left(X_{(\lambda)}\right) &= \sum_{i=1}^{\infty} \Pr(X_{(\lambda)} \geq i) \leq k + \sum_{i=1}^{\infty} \Pr(X_{(\lambda)} \geq k+i) \\ &\leq k + \sum_{i=1}^{\infty}\sum_{j=1}^{\lambda} \Pr(X_j \geq k+i) \leq k + \sum_{i=1}^{\infty} \frac{\lambda D}{e^{\eta(k+i)}} \\ &= k + e^{-\eta k}\frac{D\lambda}{e^\eta - 1} \leq k + \frac{D\lambda}{\eta e^{\eta k}} = \frac{(\ln(D\lambda)+1)}{\eta}. \quad \square\end{aligned}$$

We now state the λ-parallel black-box complexity of LO.

Theorem 4. *Let $\ln^+ x := \max(1, \ln x)$. The λ-parallel unbiased black-box complexity of* LO *is*

$$\Omega\left(\frac{\lambda n}{\ln^+(\lambda/n)} + n^2\right) \quad \text{and} \quad O\big(\lambda n + n^2\big).$$

The cut-off point is $\lambda^*_{\text{LO}} = n$. The parallel time for an optimal algorithm is $\Omega\left(\frac{n}{\ln^+(\lambda/n)} + \frac{n^2}{\lambda}\right)$ and $O\left(n + \frac{n^2}{\lambda}\right)$.

This result solves an open problem from Lässig and Sudholt [44], confirming that the analysis of the realm of linear speedups for LO from Lässig and Sudholt [44] is tight.

Proof of Theorem 4. The upper bound of $O\big(\lambda n + n^2\big)$ follows easily from an upper bound of $O\left(n + \frac{n^2}{\lambda}\right)$ on the number of generations for a (1+λ) EA from Lässig and Sudholt [43, Theorem 1][8]. The intuition behind this bound is that λ parallel queries can lead to a speedup of a factor of $\Theta(\lambda)$, compared to the expected time of $\Theta(n^2)$ for the (1+1) EA. The upper bound also contains an additive term of n for the number of non-optimal fitness values. This term limits the possible speedups that can be proven using the cited theorem.

A lower bound $\Omega(n^2)$ follows from the unary unbiased black-box complexity of LO [45], which by (2) is a lower bound on the λ-*parallel* unary unbiased black-box complexity. Hence the statement holds for the case $\lambda = O(n)$. Thus we only need to consider the case $\lambda = \omega(n)$ and to prove a lower bound of $\Omega\left(\frac{\lambda n}{\ln^+(\lambda/n)}\right) = \Omega\left(\frac{\lambda n}{\ln(\lambda/n)}\right)$ for this case.

We proceed by drift analysis. Let the "potential" of a search point x be

$$\max_{0 \leq j \leq t, 1 \leq i \leq \lambda} \{\text{LO}(x^i(j)), \text{LZ}(x^i(j)), n/2\}$$

and define the potential of the algorithm, P_t at time t to be the highest potential of all search points produced until time t.

Assume that the potential in generation t is $P_t = k$. In any generation t, let X_i for $i \in [\lambda]$ be the indicator variable for the event that all of the first $k+1$ bit-positions in individual i are 1-bits (or 0-bits). Furthermore, let Y_i be the number of consecutive 1-bits (or 0-bits) from position $k+2$ and onwards, ie., the number of "free riders".

To bound the progress in potential, we now estimate a bound on the expectation of $\max_{i \in [\lambda]} X_i Y_i$. We first claim that $\Pr(X_i = 1) = O(1/n)$ by recapping arguments from the proof of Theorem 2 in Lehre and Witt [45]. For any previously generated search point x, the number of 0-bits (or 1-bits) s in the first $k+1$ positions satisfies $1 \leq s \leq k+1$. Assume that the algorithm creates a new search point x' by flipping r bits uniformly at random in the selected search point x. Clearly, in order for the offspring x' to have only 1-bits (or 0-bits) in the first $k+1$ bit-positions, it is necessary that $r \geq s$. Focusing only on the first $k+1$ bit-positions, the algorithm must flip exactly s 0-bits in the first $k+1$ positions, and no 1-bits.

[7] Strictly speaking, we should be writing $\lambda(n) = O(\lambda^*(n))$ as the degree of parallelism may depend on n. We omit this parameter for ease of presentation. Asymptotic statements always refer to n.

[8] The cited theorem gives an upper bound for an island model with a complete topology; however, the differences to a (1+λ) EA are irrelevant in the context of this upper bound.

Optimistically assuming that the algorithm flips exactly s bit-positions within the first $k+1$ positions, the algorithm needs to choose s bits correctly out of $k+1$ bit positions. Thus, the probability that the first $k+1$ bits in the new search point x' are only 1-bits (or only 0-bits) is therefore no more than

$$\frac{1}{\binom{k+1}{s}} = \frac{s}{k+1} \cdot \frac{s-1}{k} \cdot \dots \cdot \frac{1}{k-s+1} \leq \frac{1}{k+1} = O(1/n).$$

The claim now follows by a union bound, taking into account the probability of having all 0-bits or all 1-bits in the first $k+1$ bit-positions.

Defining $M := \sum_{i=1}^{\lambda} X_i$, we therefore have $\mathrm{E}(M) = O(\lambda/n)$. For all $\lambda - M$ indices i where $X_i = 0$, we clearly have $X_i Y_i = 0$. For the other M indices i where $X_i = 1$, we have $X_i Y_i = Y_i$. Since the algorithm uses unary unbiased variation operators, Lemma 1 in Lehre and Witt [45] implies that each random variable Y_i, $i \in [\lambda]$, is stochastically dominated by a geometric random variable Z_i with parameter $1/2$. The expected progress in potential is therefore

$$\mathrm{E}\left(\Delta_{(\lambda)}\right) = \mathrm{E}\left(\max_{i\in[\lambda]} X_i Y_i\right) \leq \mathrm{E}\left(\max_{i\in[M]} Z_i\right).$$

The mgf of the geometric random variable Z_i is $M_{Z_i}(\eta) = 1/(2-e^{\eta})$. The tower property of the expectation and Lemma 3 with $\eta := \ln(3/2)$ and $D := 2$ give

$$\begin{aligned}\mathrm{E}\left(\Delta_{(\lambda)}\right) &\leq \mathrm{E}\left(\mathrm{E}\left(\max_{i\in[M]} Z_i \mid M\right)\right)\\ &\leq \mathrm{E}\left((\log(DM)+1)/\eta\right)\\ &\leq (\log(\mathrm{E}(DM))+1)/\eta = O(\ln^{+}(\lambda/n)),\end{aligned}$$

where the last inequality follows from Jensen's inequality and the last equality follows from $\log(\lambda/n) = \Omega(1)$. With overwhelmingly high probability, the initial potential is at least $n/2$. Hence, by classical additive drift theorems [36], the expected number of rounds to reach the optimum is $\Omega(n/\ln^{+}(\lambda/n))$. Multiplying by λ gives the number of function evaluations. □

IV. Parallel Black-Box Complexity of Functions with One Unique Optimum

Jansen, De Jong, and Wegener [38] considered the (1+λ) EA and established a cut-off point for λ where the running time increases from $\Theta(n \log n)$ to $\omega(n \log n)$:

$$\lambda^*_{\text{(1+}\lambda\text{) EA on OneMax}} = \Theta((\ln n)(\ln\ln n)/(\ln\ln\ln n)) \quad (4)$$

Doerr and Künnemann [17] presented the following tight bounds for bounded λ:

Theorem 5 (Adapted from Doerr and Künnemann [17])**.** *The expected optimisation time of the (1+λ) EA on* OneMax *is*

$$\Theta\left(n \cdot \frac{\lambda \ln^{+}\ln^{+}\lambda}{\ln^{+}\lambda} + n\log n\right)$$

where the upper bound holds for $\lambda = O(n^{1-\varepsilon})$ and the lower bound holds for $\lambda = O(n)$.

We show that the parallel black-box complexity is lower than the bound from Theorem 5 for large λ by a factor of order $\ln^{+}\ln^{+}\lambda$.

Theorem 6. *For any $\lambda \leq e^{\sqrt{n}}$ the λ-parallel unbiased unary black-box complexity for any function with a unique optimum is at least*

$$\Omega\left(\frac{\lambda n}{\ln^{+}\lambda} + n\log n\right).$$

The corresponding parallel time for an optimal algorithm is $\Omega\left(\frac{n}{\ln^{+}\lambda} + \frac{n\log n}{\lambda}\right)$.

We will show in the next section that this bound is tight for OneMax. Consequently, the cut-off point for OneMax is

$$\lambda^*_{\text{OneMax}} = \Theta(\log(n) \cdot \log\log n).$$

This is higher than the cut-off point for the (1+λ) EA with the standard mutation rate $p = 1/n$ from (4) and Jansen, De Jong, and Wegener [38].

To prove Theorem 6 we consider the progress made during a round of λ variations in terms of a potential function defined in the following. The following definitions and arguments, including several lemmas shown in the following, will also be used in Section VI to prove lower bounds that hold with overwhelming probability.

Without loss of generality, we assume that the search point 1^n is the optimum. Following Lehre and Witt [45], we assume a "mirrored" sampling process, where every time a bit string x is queried (including in the initial generation), the algorithm queries the complement bit string $\overline{x}$ for "free". This is necessary as a black-box algorithm can try to locate the complement of the global optimum and it then just needs to flip all bits to find the optimum. Thus, we have to consider the progress towards the global optimum as well as the progress towards its complement.

Definition 7. *Define the 0-potential s_0^t as the minimum number of zeros in all search points queried in all steps up to time t. For all $s_0^t \leq m \leq n - s_0^t$ and $r \in \{0,\dots,n\}$ we define the random variable $\Delta_0(s_0^t, m, r) := \max\{0, s_0^t - |y|_0\}$ where $|y|_0$ is the number of zeros in a random search point y obtained by applying unbiased variation with radius r to a search point with m zeros. Define the 1-potential s_1^t and Δ_1 symmetrically with respect to the number of ones.*

Due to mirrored sampling, we always have $s_0^t = s_1^t$, hence we simply write s^t or just s if we refer to the current point in time. Then we define the progress in terms of the potential as $\Delta(s,m,r) = \max\{\Delta_0(s,m,r), \Delta_1(s,m,r)\}$.

Note in particular that for all $z \in \mathbb{N}$ we have

$$\begin{aligned}&\Pr\left(\Delta(s,m,r) \geq z\right)\\ &\leq \Pr\left(\Delta_0(s,m,r) \geq z\right) + \Pr\left(\Delta_1(s,m,r) \geq z\right). \quad (5)\end{aligned}$$

Also note that by symmetry of zeros and ones $\Delta_0(s,m,r)$ has the same distribution as $\Delta_1(s,n-m,r)$, hence it suffices to study the distribution of Δ_0. We also have for all s,m,r with $s \leq m \leq n-s$,

$$\Delta_0(s,m,r) = \Delta_0(s,n-m,n-r) \quad (6)$$

as flipping all bits (in the transition from m to $n-m$) and then flipping all but r bits in the variation has the same effect as flipping r bits in the first place. Hence it suffices to consider $\Delta_0(s,m,r)$ for $s \leq m \leq n/2$.

Now consider the progress $\Delta_0(s,m,r)$. Let Z be the number of 0-bits that flipped to 1, then there are $r-Z$ new 0-bits that were originally 1. Therefore, the number of 0-bits in the new generated search point is $m-Z+(r-Z)$ where Z can be described by the hypergeometric distribution with parameters n, m and r. We only make progress if the number of 0-bits in the new search point is less than s. Hence the progress (decrease in 0-potential) is

$$\begin{aligned}\Delta_0(s,m,r) &= \max\{Z-(r-Z)+(s-m),0\}\\ &= \max\{2Z-r+s-m,0\}.\end{aligned}$$

We show a tail inequality for hypergeometric variables and use this to derive a progress bound for the 0-potential.

Lemma 8. *Let Z be a hypergeometrically distributed random variable with parameters n (number of balls), m (number of red balls), and r (number of balls drawn). For all $z \in \mathbb{N}_0$,*

$$\Pr(Z=z) \le \binom{r}{z}\cdot\frac{m^z}{n^z}.$$

If $z \ge r/2$, this is at most $\left(\frac{4m}{n}\right)^z$.

Proof. We assume $z \le m$ and $z \le r$ as otherwise $\Pr(Z=z)=0$. We further assume $z \ge 1$ as for $z=0$ the probability bound is 1 and the statement is trivial. Now,

$$\begin{aligned}\Pr(Z=z) &= \binom{m}{z}\binom{n-m}{r-z}\Big/\binom{n}{r}\\ &= \frac{m!(n-m)!r!(n-r)!}{z!(m-z)!(r-z)!(n-m-r+z)!n!}\\ &= \binom{r}{z}\cdot\frac{m!(n-m)!(n-r)!}{(m-z)!(n-m-r+z)!n!}. \qquad (7)\end{aligned}$$

The fraction can be written as

$$\frac{m(m-1)\cdot\ldots\cdot(m-z+1)}{n(n-1)\cdot\ldots\cdot(n-z+1)}\cdot \frac{(n-m)(n-m-1)\cdot\ldots\cdot(n-m-r+z+1)}{(n-z)(n-z-1)\cdot\ldots\cdot(n-r+1)}$$

Since $z \le m$, the second fraction above is at most 1. The first fraction is at most m^z/n^z as $(m-i)/(n-i) \le m/n$ for all $i \in \mathbb{N}$ and $m \le n$. Plugging this into (7) yields

$$\Pr(Z=z) \le \binom{r}{z}\cdot\frac{m^z}{n^z}.$$

If $z \ge r/2$, this is at most $\left(\frac{4m}{n}\right)^z$ as $\binom{r}{z} \le 2^r \le 2^{2z} = 4^z$. □

The next lemma shows that for any radius r the probability of having a progress of z decreases exponentially with z.

Lemma 9. *Let s denote the current 0-potential. If $s \le m \le n/8$, then for all $z \in \mathbb{N}$ and $r \in \{1,\ldots,n\}$,*

$$\Pr(\Delta_0(s,m,r)=z) \le \left(\frac{1}{2}\right)^{z/2}.$$

Proof. Applying Lemma 8 to the hypergeometric random variable Z with parameters m and r we have, for all $z \in \mathbb{N}_0$,

$$\begin{aligned}\Pr(\Delta_0(s,m,r)=z) &= \Pr\left(Z=\frac{z+r+m-s}{2}\right)\\ &\le \left(\frac{4m}{n}\right)^{(z+r+m-s)/2} \le \left(\frac{1}{2}\right)^{z/2}.\ \square\end{aligned}$$

The following lemma gives another tail bound that will be used to exclude steps where a search point of potential $m \gg s$ is chosen for variation. The probability of having a positive progress decreases rapidly with growing $m-s$.

Lemma 10. *For every $s \le m \le n/2$ and every $r \in \{1,\ldots,n\}$*

$$\Pr(\Delta_0(s,m,r)>0) \le \exp\left(-\frac{(m-s)^2}{2r}\right).$$

Proof: We use the following well-known tail bound for the hypergeometric distribution [4]: $\Pr(Z \ge \mathrm{E}(Z)+r\delta) \le \exp(-2\delta^2 r)$, where $\mathrm{E}(Z)=\frac{rm}{n}$. The first inequality follows from $r/(2r)-m/n = 1/2-m/n \ge 0$.

$$\begin{aligned}&\Pr(\Delta_0(s,m,r)>0)\\ &= \Pr\left(Z > \frac{r+m-s}{2}\right)\\ &= \Pr\left(Z > \frac{rm}{n} + r\cdot\left(\frac{r+m-s}{2r}-\frac{m}{n}\right)\right)\\ &\le \Pr\left(Z \ge \frac{rm}{n} + r\cdot\left(\frac{m-s}{2r}\right)\right)\\ &\le \exp\left(-2r\left(\frac{m-s}{2r}\right)^2\right) = \exp\left(-\frac{(m-s)^2}{2r}\right).\end{aligned}$$

■

Putting all lemmas together shows that the expected progress is at most logarithmic in λ.

Lemma 11. *Let $\Delta_0^{(\lambda)}$ be the maximum of λ random variables $\Delta_0(s,m_1,r_1),\ldots,\Delta_0(s,m_\lambda,r_\lambda)$ for arbitrary values $m_1,\ldots,m_\lambda$ and $r_1,\ldots,r_\lambda$ with $s \le m_i \le n/2$ for all $1 \le i \le \lambda$. For $s \le n/16$ we have $\mathrm{E}\left(\Delta_0^{(\lambda)}\right) = O(\ln^+\lambda)$.*

Proof. If $n/8 < m_i \le n/2$ then $m_i - s \ge n/16$ and by Lemma 10 we have

$$\Pr(\Delta_0(s,m_i,r_i)>0) \le e^{-n^2/(512 r_i)} \le e^{-\Omega(n)}.$$

This means that the probability of making any progress is exponentially small, for any r_i. Thus in the following we assume that $m_i \le n/8$ for all i.

Under this assumption, applying Lemma 9, for all $z \in \mathbb{N}_0$,

$$\Pr(\Delta_0(s,m_i,r_i)=z) \le \left(\frac{1}{2}\right)^{z/2} = \left(\frac{1}{\sqrt{2}}\right)^z$$

Hence, for $\eta := \ln(4/3)$ and $D := 9+6\sqrt{2}$,

$$\begin{aligned}\mathrm{E}\left(e^{\Delta_0(s,m_i,r_i)}\right) &\le \sum_{z=0}^{\infty}\left(\frac{1}{\sqrt{2}}\right)^z e^{\eta z} = \sum_{z=0}^{\infty}\left(\frac{4}{3\sqrt{2}}\right)^z\\ &= \frac{1}{1-\frac{4}{3\sqrt{2}}} = D.\end{aligned}$$

Applying Lemma 3 proves $\mathrm{E}\left(\Delta_0^{(\lambda)}\right) = O(\ln^+\lambda)$. □

Now we are in a position to prove Theorem 6.

Proof of Theorem 6. The lower bound $\Omega(n \log n)$ follows from unbiased unary black-box complexity [45]. Hence, it suffices to prove the lower bound $\Omega(\lambda n/\ln^+ \lambda)$.

Consider any λ-parallel unary unbiased black-box algorithm. We grant the algorithm an advantage by revealing all search points with Hamming distance at least $n/16$ to both 0^n and 1^n at no cost. Hence the potential is always $s \leq n/16$. By Chernoff bounds and a union bound over λ trials, the potential after initialisation is $n/16$ with overwhelming probability.

Assuming this is the case, let $\Delta_0^{(\lambda)}$ be the progress due to reduction of the 0-potential in one step, and $\Delta_1^{(\lambda)}$ be the progress due to reduction of the 1-potential. Owing to the symmetry of Δ_0 and Δ_1, Lemma 11 also applies to $\Delta_1^{(\lambda)}$. Hence the expected change in potential per round is at most

$$\mathrm{E}\left(\Delta_0^{(\lambda)}\right) + \mathrm{E}\left(\Delta_1^{(\lambda)}\right) = O(\ln^+ \lambda).$$

Hence, by the additive drift theorem [36], the expected number of rounds until one of the search points 0^n or 1^n is obtained is $\Omega(n/\ln^+ \lambda)$. Multiplying by λ proves the claim. □

V. An Optimal Parallel Black-Box Algorithm for OneMax

The following theorem shows that the lower bound on the black-box complexity from Theorem 6 is tight. We show that the (1+λ) EA has a better optimisation time if the mutation rate is chosen adaptively, according to the current best fitness. This is similar to common ideas from artificial immune systems, particularly the clonal selection algorithm. Adaptive mutation rates for OneMax have been studied by Zarges [63], however the standard parameters for the clonal selection algorithm were too drastic to even obtain polynomial running times. Better results were obtained when using a population-based adaptation [64].

The following result reveals an optimal choice for the mutation rate of the (1+λ) EA, depending on n and λ.

Theorem 12. *On OneMax, the expected number of function evaluations of the (1+λ) EA with an adaptive mutation rate $p_i = \max\{\ln(\lambda)/(n\ln(en/i)),\ 1/n\}$, where i is the number of zeros in the current search point, for any $\lambda \leq e^{\sqrt{n}}$, is at most*

$$O\left(\frac{\lambda n}{\ln^+ \lambda} + n \log n\right).$$

The parallel time (number of generations) is $O\left(\frac{n}{\ln^+ \lambda} + \frac{n \log n}{\lambda}\right)$.

Proof. For $\lambda = 1$ the algorithm boils down to a (1+1) EA with mutation rate $1/n$, hence we assume $\lambda \geq 2$ where $\ln^+ \lambda = \Theta(\ln \lambda)$. Let i be the current number of zeros and p_i be the corresponding mutation rate. The probability of decreasing the number of zeros by any $k \in \mathbb{N}$ with $k \leq i$ is at least

$$\begin{aligned}\Pr(\Delta \geq k) &\geq \binom{i}{k} \cdot p_i^k \cdot (1-p_i)^{n-k} \\ &\geq \frac{i^k}{k^k} \cdot p_i^k \cdot (1-p_i)^{n-k} = (1-p_i)^{n-k} \cdot \left(\frac{ip_i}{k}\right)^k.\end{aligned}$$

Then the probability that one of λ offspring will decrease the number of zeros by at least k is at least, using $1-(1-p_i)^\lambda \geq 1 - e^{-p_i\lambda} \geq 1 - 1/(1+p_i\lambda) = p_i\lambda/(1+p_i\lambda)$,

$$\begin{aligned}\Pr\left(\Delta_{(\lambda)} \geq k\right) &\geq 1 - (1 - \Pr(\Delta \geq k))^\lambda \\ &\geq \frac{\lambda(1-p_i)^{n-k} \cdot (ip_i/k)^k}{1 + \lambda(1-p_i)^{n-k} \cdot (ip_i/k)^k}.\end{aligned}$$

Hence for any $k \leq i$ the drift is at least

$$\mathrm{E}\left(\Delta_{(\lambda)}\right) \geq k \cdot \frac{\lambda(1-p_i)^{n-k} \cdot (ip_i/k)^k}{1 + \lambda(1-p_i)^{n-k} \cdot (ip_i/k)^k}.$$

For $i > en/\ln\lambda$, which implies $p_i n > 1$, we set $k := p_i n = \ln(\lambda)/\ln(en/i)$. We have $k \leq i$ since $k \leq \ln(\lambda) \leq \sqrt{n} \leq en/\ln\lambda$. We use $k := 1$ for $i \leq en/\ln\lambda$, the realm where $p_i = 1/n$. This results in the following drift function h:

$$h(i) := \begin{cases} \frac{\lambda(1-1/n)^{n-1} \cdot i/n}{1+\lambda(1-1/n)^{n-1} \cdot i/n} & \text{if } i \leq en/\ln\lambda \\ p_i n \cdot \frac{\lambda(1-p_i)^{n-p_i n} \cdot (i/n)^{p_i n}}{1+\lambda(1-p_i)^{n-p_i n} \cdot (i/n)^{p_i n}} & \text{otherwise} \end{cases}$$

We estimate the number of function evaluations by multiplying the number of generations by λ. The number of generations is estimated using Johannsen's variable drift theorem [42] (see Theorem 1 in [52]), with the above function h. Along with $(1-1/n)^{n-1} \geq 1/e$, this gives an upper bound of

$$\begin{aligned}&\frac{\lambda}{h(1)} + \int_1^n \frac{\lambda}{h(i)}\ \mathrm{d}i \\ &= \frac{1 + \lambda(1-1/n)^{n-1} \cdot 1/n}{(1-1/n)^{n-1} \cdot 1/n} + \lambda \int_1^n \frac{1}{h(i)}\ \mathrm{d}i \\ &\leq en + \lambda + \lambda \int_1^{en/\ln\lambda} \frac{1}{h(i)}\ \mathrm{d}i + \lambda \int_{en/\ln\lambda}^n \frac{1}{h(i)}\ \mathrm{d}i.\end{aligned}$$

The first terms are at most

$$\begin{aligned}&en + \lambda + \lambda \int_1^{en/\ln\lambda} \frac{1 + \lambda(1-1/n)^{n-1} \cdot i/n}{\lambda(1-1/n)^{n-1} \cdot i/n} \mathrm{d}i \\ &\leq en + \lambda + \lambda \int_1^{en/\ln\lambda} \left(1 + \frac{1}{\lambda \cdot i/(en)}\right)\ \mathrm{d}i \\ &\leq \frac{\lambda en}{\ln\lambda} + en\left(1 + \int_1^{en/\ln\lambda} \frac{1}{i}\ \mathrm{d}i\right) \\ &\leq \frac{\lambda en}{\ln\lambda} + en \cdot (2 + \ln n).\end{aligned}$$

The second integral is bounded using $(1-1/x)^{x-1} \geq e^{-1}$ for $x \geq 1$ and $(1-p_i)^{n-p_i n} = (1-p_i)^{(1/p_i - 1)np_i} \geq e^{-p_i n}$,

$$\begin{aligned}&\int_{en/\ln\lambda}^n \frac{1 + \lambda(1-p_i)^{n-p_i n} \cdot (i/n)^{p_i n}}{p_i n \cdot (1-p_i)^{n-p_i n} \cdot (i/n)^{p_i n}}\ \mathrm{d}i \\ &= \int_{en/\ln\lambda}^n \left(\frac{(n/i)^{p_i n}}{p_i n \cdot (1-p_i)^{n-p_i n}} + \frac{\lambda}{p_i n}\right)\ \mathrm{d}i \\ &\leq \int_{en/\ln\lambda}^n \left(\frac{(en/i)^{p_i n}}{p_i n} + \frac{\lambda}{p_i n}\right)\ \mathrm{d}i \\ &= \int_{en/\ln\lambda}^n \left(\frac{\lambda}{p_i n} + \frac{\lambda}{p_i n}\right)\ \mathrm{d}i \\ &= \int_{en/\ln\lambda}^n \frac{2\lambda \ln(en/i)}{\ln\lambda}\ \mathrm{d}i \\ &\leq \frac{2\lambda}{\ln\lambda} \int_0^n \ln(en/i)\ \mathrm{d}i = \frac{4\lambda n}{\ln\lambda}.\end{aligned}$$

This gives the upper bound $\frac{(4+e)\lambda n}{\ln(\lambda)} + en \cdot (2 + \ln n)$. □

Note that the optimal mutation rate $p = \max\{\ln(\lambda)/(n\ln(en/i)),\ 1/n\}$, in particular the functional relationship between the mutation rate and the current fitness i, is quite hard to guess through experimentation and was only revealed through the present theoretical analysis. After the result from Theorem 12 was first published [1], Doerr, Gießen, Witt, and Yang [13] presented a self-adjusting scheme for choosing the mutation rate in the (1+λ) EA and showed that it is able to match the upper bound from Theorem 12 without knowing the functional relationship between the mutation rate and the current fitness.

VI. Tail Bounds

In this section we now show that the lower bound for all λ-parallel unbiased unary black-box algorithms from Theorem 6 holds with high probability. In particular, it also applies to (non-parallel) unbiased unary black-box algorithms, for which only lower bounds on the expectation were known before [45]. Our main result is as follows.

Theorem 13. *For every fitness function $f\colon \{0,1\}^n \to \mathbb{R}$, every constant $0 < \delta < 1$ and every set S of up to $\exp(o(n^\delta/\log n))$ search points, the following holds. Every unary unbiased λ-parallel black-box algorithm $\mathcal{A}$ on f, with probability $1-\exp(-\Omega(n^\delta/\log n))$, does not query any search point from S within time*

$$\max\left\{\frac{\lambda n}{60\ln^+\lambda}, (1-\delta)n\ln n\right\} = \Omega\left(\frac{\lambda n}{\ln^+\lambda} + n\ln n\right).$$

The expected time also satisfies the asymptotic bound.

Theorem 13 establishes very general limits to the performance of large classes of algorithms, including mutation-only evolutionary algorithms with standard mutation operators, local search, and simulated annealing. In particular, putting $\delta := 0.01$ (say), Theorem 13 shows that every unary unbiased search algorithm needs to be run for at least $n\ln n$ evaluations as the probability of finding one of few global optima within $0.99n\ln n$ evaluations is overwhelmingly small. The same holds for λ-parallel unary unbiased algorithms like mutation-only evolutionary algorithms with offspring populations of size λ. Here stopping a run before $\lambda n/(60\ln^+\lambda)$ evaluations is futile as with overwhelming probability no optimum will have been found yet.

In addition, Theorem 13 makes a statement about a target set of up to exponential size. This means that the lower bounds also apply to functions with many global optima, with respect to the optimisation time, but it can also be used to bound the time to find local optima or any set of high-fitness individuals of size at most $\exp(o(n^\delta/\log n))$. Section VII gives illustrative applications to a broad range of well-known problems.

Theorem 13 will be shown by separately showing lower bounds of $\Omega(\lambda n/\ln^+\lambda)$ and $\Omega(n\log n)$ for the time to locate any fixed target search point x^* that both hold with overwhelming probability. Then we use a union bound to show that even the probability to find one of exponentially many target search points within the stated time is still exponentially small. Again, we will assume "mirrored" sampling, i. e. every queried search point x also evaluates $\overline{x}$ for free.

A. Lower Bound $\Omega(\lambda n/\ln^+\lambda)$ with overwhelming probability

We start with a bound of $\Omega(\lambda n/\ln^+\lambda)$ for the time to find a particular target search point x^*, w. l. o. g. $x^* = 1^n$. Recall from Definition 7 that due to mirrored sampling, we can define the potential as the minimum number zeros, or equivalently number of ones, in all search points up to time t. We will use Theorem 2 from [46] for a tail bound on the runtime, which requires the mgf. of the progress

$$\Delta^{(\lambda)}(s) := \max_{m,r}\left\{\Delta_0^{(\lambda)}(s,m,r), \Delta_1^{(\lambda)}(s,m,r)\right\},$$

where $\Delta_0^{(\lambda)}(s,m,r)$ is the maximal progress in the 0-potential, and $\Delta_1^{(\lambda)}(s,m,r)$ is the maximal progress in the 1-potential, given current potential s, where the selected search point has m 0-bits, respectively 1-bits, and r bits are flipped.

Lemma 14. *Let s denote the current potential. If $s \le \frac{n}{8}$ and $\gamma := \ln\left(\frac{3}{4}\sqrt{2}\right)$, then $\mathbf{E}\left[e^{\gamma\Delta^{(\lambda)}(s)}\right] \le 8\lambda$.*

Proof. As noted in Definition 7 and (6)

$$\Delta_1(s,m,r) = \Delta_0(s,n-m,r) = \Delta_0(s,m,n-r).$$

Hence, by a union bound

$$\begin{aligned}&\Pr\left(\Delta(s,m,r)=z\right)\\ &\le \Pr\left(\Delta_0(s,m,r)=z\right) + \Pr\left(\Delta_1(s,m,r)=z\right)\\ &= \Pr\left(\Delta_0(s,m,r)=z\right) + \Pr\left(\Delta_0(s,m,n-r)=z\right) \le 2^{1-\frac{z}{2}}\end{aligned}$$

where the last inequality follows by Lemma 9. We now have

$$\mathbf{E}\left[e^{\gamma\Delta^{(\lambda)}(s,m,r)}\right] = \sum_{z=0}^{\infty}\Pr\left(\Delta^{(\lambda)}=z\right)e^{\gamma z},$$

by a union bound over λ parallel runs

$$\le \sum_{z=0}^{\infty}\lambda\max_{r\in[n],m\ge s}\Pr\left(\Delta(s,m,r)=z\right)e^{\gamma z}$$

the definition of γ gives

$$\le 2\lambda\sum_{z=0}^{\infty}\left(\frac{1}{2}\right)^{z/2}\left(\frac{3}{4}\sqrt{2}\right)^{z} = 8\lambda. \qquad \square$$

Theorem 15. *For every unary unbiased λ-parallel black-box algorithm $\mathcal{A}$, the probability that $\mathcal{A}$ finds any fixed target search point x^* within $\lambda n/(60\ln^+\lambda)$ steps is $e^{-\Omega(n)}$.*

Proof. Following the proof of Theorem 6, we assume without loss of generality that the search point 1^n is the optimum, and let $(X_t)_{t\in\mathbb{N}}$ be the potential as defined before.

We apply the last part of Theorem 2 (iv), from [46], with the parameters $g(x) := x$, $x_{\min} := 1$, $x_{\max} := n$, $a := 0$, $S := \{0\} \cup [x_{\min}, x_{\max}]$, and $\beta_l(t) := 8\lambda$, for all $t \in \mathbb{N}$. We consider the number of *parallel runs* T' until the process reaches potential $a = 0$.

Define $c := \frac{3}{10}\gamma$ where $\gamma := \ln\left(\frac{3}{4}\sqrt{2}\right)$. By Lemma 14

$$\mathbf{E}\left[e^{\gamma(g(X_t)-g(X_{t+1}))}\,;\ X_t > a \mid \mathcal{F}_t\right]$$

$$\leq \mathbf{E}\left[e^{\gamma\Delta^{(\lambda)}(s)}\right] \leq 8\lambda = \beta_\ell(t)$$

Furthermore, by the definition of the process, if the process reaches the set $S \cap \{x \mid x \leq a\} = \{0\}$ then it never leaves this set, i.e., the set $S \cap \{x \mid x \leq a\}$ is *absorbing*. Thus for $t := \frac{cn}{\ln^+ \lambda}$,

$$\begin{aligned}\Pr\left(T' < t \mid X_0 > 0\right) &\leq \left(\prod_{i=0}^{t-1} \beta_\ell(i)\right) \cdot e^{-\gamma(g(X_0)-g(a))} \\ &< (8\lambda)^t \cdot e^{-\gamma n} \\ &= (8\lambda)^{\frac{cn}{\ln^+ \lambda}} \cdot e^{-\gamma n} \\ &= e^{\left(\frac{cn}{\ln^+ \lambda}\right)\ln(8\lambda) - \gamma n}\end{aligned}$$

using that $\ln(8\lambda) = \ln(\lambda) + 3\ln(2) \leq 3\ln^+ \lambda$ gives

$$\leq e^{(3c-\gamma)n} = e^{-\gamma n/10}.$$

The result follows by taking into account that the algorithm makes λ fitness evaluations per iteration, i. e., $T = \lambda T'$, and that $c > 1/60$. □

B. Lower Bound $\Omega(n \log n)$ with overwhelming probability

Now we show a lower bound of $\Omega(n \log n)$ with overwhelming probability. Note that this result is independent of λ and thus unrelated to parallel black-box complexity; it gives limitations for general (parallel or non-parallel) unary unbiased black-box algorithms. Recall that every λ-parallel unary unbiased algorithm is also a unary unbiased algorithm, hence the result applies to a strictly larger class of algorithms. Previously only lower bounds on the expectation were known: Lehre and Witt [45] showed an asymptotic bound of $\Omega(n \log n)$ and Doerr, Doerr, and Yang [12] presented a more precise lower bound of $n \ln n - O(n)$.

Theorem 16. *For every unary unbiased black-box algorithm $\mathcal{A}$ and every constant $0 < \delta \leq 1$, the probability that $\mathcal{A}$ finds any fixed target search point x^* within $(1-\delta)n \ln n$ steps is $\exp(-\Omega(n^\delta / \log n))$.*

Before presenting the proof of Theorem 16, we present the main idea behind the proof, and the challenges to overcome.

The proof will be based on the following well-known "coupon collector" argument that we discuss first for a simple algorithm such as Randomised Local Search (RLS) or the (1+1) EA. For these algorithms, we can argue that with high probability there will be cn bits in the initial search point that differ from the optimum, for an appropriate constant $0 < c < 1/2$. Each such bit has a probability of $1/n$ of being flipped in each step of the algorithm. For a time period of $T := (1-\delta)(n-1)\ln n$ steps, the probability that any fixed bit is never being flipped is at least

$$\left(1 - \frac{1}{n}\right)^T \geq \left(1 - \frac{1}{n}\right)^{(1-\delta)(n-1)\ln n} \geq n^{-(1-\delta)}$$

using $(1-1/n)^{n-1} \geq 1/e$. Now the probability that there is a bit among the cn incorrect bits that is never being flipped is at least

$$\left(1 - n^{-(1-\delta)}\right)^{cn} \leq \exp(-cn^\delta).$$

This implies that with the above probability the optimum has not been found in $T = \Omega(n \log n)$ steps.

This argument works for RLS and the (1+1) EA for the following reasons:

1) The algorithms evolve a single lineage from the initial search point, which allows us to argue with "incorrect" bits that need to be flipped at least once.
2) The same variation operator is applied at all times, which establishes the formula $(1-1/n)^T$.
3) All bits are treated independently, which is implicitly used in the derivation of the term $(1-n^{-(1-\delta)})^{cn}$.

In order to prove Theorem 16, we have to consider *all* unary unbiased black-box algorithms, for which the above properties do not hold. In particular, algorithms may easily generate several lineages. This makes it unclear how "incorrect" bits can be defined. Also note that an algorithm might flip many "incorrect" bits in one step simply by choosing a very large radius. So the simple argument that we need to flip all incorrect bits at least once breaks down. Algorithms may choose different variation operators at different times, possibly depending on fitness values generated so far. This makes it difficult to argue that no variation flips a bit over a period of time. Finally, mutations with a fixed radius $r \geq 2$ may introduce dependencies between bits, which needs to be addressed.

We tackle these challenges as follows. Assume w. l. o. g. that $x^* = 1^n$. We give away knowledge of all search points x that have Hamming distance at least $n^* := n/(2^{13}\ln n)$ to both 0^n and 1^n. Hence we start with a potential of $s = n^*$. Moreover, whenever the algorithm decreases the potential from s to $s' < s$, we grant the algorithm knowledge of all solutions with Hamming distance at least s' from both 0^n and 1^n. This assumption implies that the current knowledge of the algorithm can be fully described by the current potential, and the progress of the algorithm can be bounded by considering the transitions of the potential.

Note that all solutions with the same potential are isomorphic to the algorithm. Pick a set of n^* bit positions, w. l. o. g. the first n^* ones. We define these bits as "incorrect" bits that need to be set to 1 in order to reach the optimum. Since the behaviour of the algorithm is fully determined by the current potential, and the bit positions are irrelevant for transitions between potential values, we may assume w. l. o. g. that, whenever the algorithm performs a variation of a search point x_t with k ones, $x_t = 0^{n-k}1^k$.

Now variations that decrease the potential by decreasing the number of zeros will fix some of the incorrect bits accordingly. Variations that do not decrease the potential only create search points that are already known and thus can be ignored as they have no effect. Hence we require that these incorrect bits are flipped *in variations that decrease the potential*.

Having laid the foundation for arguing with "incorrect" bits being fixed, we now show that with overwhelming probability, $\mathcal{A}$ does not find 1^n within $T := (1-\delta)(n-1)\ln n$ steps.

Note that $\mathcal{A}$ can choose the radius in each step. We distinguish between single-bit variations where $r = 1$ (or, symmetrically, $r = n-1$) and multi-bit variations where $2 \leq r \leq n-2$. We first show that in at most T steps with multi-bit variations, not too many incorrect bits are being fixed.

Then we show later that at most T single-bit variations are not enough to fix all incorrect bits that are not being fixed by multi-bit variations. Note that the algorithm can interleave single-bit variations and multi-bit variations arbitrarily. Our arguments work for arbitrary sequences of single-bit and multi-bit variations; they even hold if the algorithm is allowed to make T single-bit variations *and* T multi-bit variations at the cost of T queries.

The following lemma considers multi-bit variations and bounds transition probabilities of the potential.

Lemma 17. *Let $s \leq n^*$ for $n^* := n/(2^{13}\ln n)$, then for every $m \in [s, 2n^*] \cup [n-2n^*, n-s]$, every radius $2 \leq r \leq n-2$ and every $1 \leq z \leq n$ we have*

$$\Pr\left(\Delta_0(s,m,r) = z\right) \leq \left(\frac{16n^*}{n}\right)^2 \cdot 2^{-z}.$$

If $2n^* < m < n - 2n^*$ we have

$$\Pr\left(\Delta_0(s,m,r) = z\right) \leq e^{-\Omega(n^{*2}/n)}.$$

Proof. Recall that by (6) it suffices to consider the case $m \leq n/2$. If $2n^* \leq m \leq n/2$ then by Lemma 10

$$\Pr\left(\Delta_0(s,m,r) > 0\right) \leq \exp\left(-\frac{(m-s)^2}{2r}\right) = e^{-\Omega(n^{*2}/n)}.$$

Now assume $s \leq m \leq 2n^*$. As shown in the proof of Lemma 9,

$$\Pr\left(\Delta_0(s,m,r) = z\right) \leq \left(\frac{4m}{n}\right)^{(z+r+m-s)/2} \leq \left(\frac{8n^*}{n}\right)^{(z+r)/2}$$

We claim that the above is bounded by $\left(\frac{16n^*}{n}\right)^2 \cdot 2^{-z}$ for all $z \geq 1$ and all $r \geq 2$.

Note that $\Pr\left(\Delta_0(s,m,r) = z\right) = 0$ if $z > r$ or if $z = 1$ and $r = 2$ as the progress must be an even number. For $z = 1$ and $r \geq 3$ we get

$$\left(\frac{8n^*}{n}\right)^{(z+r)/2} = \left(\frac{8n^*}{n}\right)^2 \cdot \left(\frac{8n^*}{n}\right)^{(r-3)/2} \leq \left(\frac{16n^*}{n}\right)^2 \cdot 2^{-1}.$$

For $z = 2$ and all $r \geq 2$ we get

$$\left(\frac{8n^*}{n}\right)^{(z+r)/2} = \left(\frac{8n^*}{n}\right)^2 \cdot \left(\frac{8n^*}{n}\right)^{(r-2)/2} \leq \left(\frac{16n^*}{n}\right)^2 \cdot 2^{-2}.$$

For $z = 3$ and $r = 3$ we get, using $(8n^*/n)^{1/2} \leq 1/2$,

$$\left(\frac{8n^*}{n}\right)^{(z+r)/2} \leq \left(\frac{8n^*}{n}\right)^2 \cdot \left(\frac{8n^*}{n}\right)^{(r-1)/2} \leq \left(\frac{16n^*}{n}\right)^2 \cdot 2^{-3}.$$

□

For all $r \geq 4$ we have, using $(8n^*/n)^{1/2} \leq 1/2$,

$$\left(\frac{8n^*}{n}\right)^{(z+r)/2} \leq \left(\frac{8n^*}{n}\right)^2 \cdot \left(\frac{8n^*}{n}\right)^{z/2} \leq \left(\frac{8n^*}{n}\right)^2 \cdot 2^{-z}.$$

Using Lemma 17 now allows us to express the progress of any algorithm using stochastic domination and a combination of two simple random variables:

Lemma 18. *Let $s \leq n^*$ for $n^* := n/(2^{13}\ln n)$, then for every $s \leq m \leq n-s$ and every radius $2 \leq r \leq n-2$ the progress $\Delta(s,m,r)$ is stochastically dominated by $X_t Y_t$ where $X_t \in \{0,1\}$ is a Bernoulli random variable with $\Pr(X_t = 1) = 2\left(\frac{16n^*}{n}\right)^2$ and Y_t is a geometric random variable with parameter $1/2$, X_t and Y_t being independent of each other and independent of other time steps $t' \neq t$.*

Proof. By Lemma 17 and the definition of X_t, Y_t,

$$\Pr\left(\Delta_0(s,m,r) = z\right) \leq \left(\frac{16n^*}{n}\right)^2 \cdot 2^{-z} = \frac{\Pr\left(X_t Y_t = z\right)}{2}$$

for every $z \geq 1$ and all $m \in [s, 2n^*] \cup [n-2n^*, n-s]$. The same clearly also holds in case $2n^* < m < n - 2n^*$ by the second statement of Lemma 17. This implies $\Pr\left(\Delta_0(s,m,r) \geq z\right) \leq \Pr\left(X_t Y_t \geq z\right)/2$ for all $z \geq 1$.

The probability bounds for Δ_0 also apply to Δ_1 by symmetry of zeros and ones, and thus by the union bound $\Pr\left(\Delta(s,m,r) \geq z\right) \leq \Pr\left(\Delta_0(s,m,r) \geq z\right) + \Pr\left(\Delta_1(s,m,r) \geq z\right)$ we get $\Pr\left(\Delta(s,m,r) \geq z\right) \leq \Pr\left(X_t Y_t \geq z\right)$ for all $z \geq 1$. The last inequality also holds trivially for $z = 0$ as then both sides are 1. This completes the proof. □

We use Lemma 18 to show tail bounds for the progress made in multi-bit variations. The following lemma shows that at most half of the incorrect bits are being fixed by multi-bit variation steps, even when considering a time span of $n \ln n$ steps instead of $(1-\delta) n \ln n$.

Lemma 19. *Let $n^* := n/(2^{13}\ln n)$. Within $T := n \ln n$ multi-bit variation steps at most $n^*/2$ incorrect bits are being fixed, with probability $1 - 2^{-\Omega(n/\log n)}$.*

Proof. We give a tail bound for the sum of variables $X_t Y_t$ defined in Lemma 18; by stochastic domination, the tail bound then also holds for the real progress. Recall that X_t as well as Y_t are both sequences of iid variables and that all variables are mutually independent.

By Chernoff bounds, with overwhelming probability the number of X_t variables attaining value 1 is bounded by at most twice its expectation:

$$\Pr\left(\sum_{t=1}^{T} X_t \geq 4T\left(\frac{16n^*}{n}\right)^2\right) \leq \exp\left(-\frac{2T}{3}\left(\frac{16n^*}{n}\right)^2\right) = e^{-\Omega(n/\log n)}.$$

If $\sum_{t=1}^{T} X_t \leq \left\lfloor 4T\left(\frac{16n^*}{n}\right)^2 \right\rfloor =: k$ then there are at most k variables Y_t that contribute to $\sum_{t=1}^{T} X_t Y_t$. For ease of notation, we assume that these are variables $Y_1, \ldots, Y_k$.

We apply Chernoff bounds for sums of geometric random variables [10, Theorem 3] to bound the contribution of k variables $Y_1, \ldots, Y_k$. Note that $\mathrm{E}\left(\sum_{t=1}^{k} Y_t\right) = 2k$.

$$\Pr\left(\sum_{t=1}^{k} Y_t \geq 4k\right) \leq \exp\left(-\frac{k-1}{4}\right) = e^{-\Omega(n/\log n)}.$$

Hence if both "typical" events occur,

$$\sum_{t=1}^{T} X_t Y_t \leq 4k \leq 16T \cdot \frac{16^2 n^* \cdot n^*}{n^2} = \frac{16n\ln(n) n^*}{2^5 n \ln n} = \frac{n^*}{2}.$$

Taking the union bound for the two probabilities $2^{-\Omega(n/\log n)}$ that the typical events do not happen completes the proof. □

Now we are ready to give a proof for Theorem 16.

Proof of Theorem 16. As explained earlier, it suffices to consider n^* incorrect bits and to show that with the claimed probability not all of these bits will be fixed within T unbiased variations.

Lemma 19 implies that with overwhelming probability there exist $n^*/2$ incorrect bits that are not being fixed by up to T multi-bit variations. We now use coupon collector arguments (similar to those sketched earlier) to show that, in up to T single-bit variations, with overwhelming probability these $n^*/2$ incorrect bits will not all be fixed.

The probability that any fixed bit i will not be flipped in a single-bit variation amongst the first T steps is at least, using $(1-1/x)^{x-1} \geq 1/e$ for $x > 1$,

$$\left(1-\frac{1}{n}\right)^{T} = \left(1-\frac{1}{n}\right)^{(1-\delta)(n-1)\ln n} \geq n^{-(1-\delta)}.$$

Hence the probability that a fixed bit i will be flipped in up to T single-bit variations is at least $1-n^{-(1-\delta)}$. Hence the probability that all of the $n^*/2$ incorrect bits are being flipped in T steps is at most

$$(1-n^{-(1-\delta)})^{n^*/2} \leq \exp(-\Omega(n^\delta/\log n)). \qquad \square$$

Theorems 15 and 16 imply our main result, Theorem 13.

Proof of Theorem 13. Fix a target search point x^* from the target set. By Theorem 15 the probability of finding x^* within $\frac{\lambda n}{60\ln^+\lambda}$ steps is $\exp(-\Omega(n))$. Applying Theorem 16 with parameter δ yields that the probability of finding x^* within $(1-\delta)n\ln n$ steps is $\exp(-\Omega(n^\delta/\log n))$. By the union bound, the probability that one of these lower bounds does not apply is $\exp(-\Omega(n)) + \exp(-\Omega(n^\delta/\log n)) \leq 2\exp(-\Omega(n^\delta/\log n))$. Repeating the above arguments for all target search points and using a union bound over at most $\exp(o(n^\delta/\log n))$ search points yields an overall probability bound of

$$\begin{aligned}&\exp(o(n^\delta/\log n)) \cdot 2\exp(-\Omega(n^\delta/\log n))\\ &= \exp(-\Omega(n^\delta/\log n) + o(n^\delta/\log n) + \ln 2)\\ &= \exp(-\Omega(n^\delta/\log n)).\end{aligned}$$

Finally, the claimed equality

$$\max\left\{\frac{\lambda n}{60\ln^+\lambda}, (1-\delta)n\ln n\right\} = \Omega\left(\frac{\lambda n}{\ln^+\lambda} + n\ln n\right)$$

follows from $\max\{x,y\} \geq (x+y)/2$ and $1-\delta = \Omega(1)$. □

VII. Black-Box Complexity Results for Illustrative Function Classes

In this section we give a number of examples of how to exploit the fact that our lower bounds apply to the time for finding an arbitrary target set of up to exponentially many search points. This leads to novel results for functions with many global optima, but can also be used to bound the time for reaching local optima or search points within a certain distance from any local or global optimum.

A. Black-Box Complexity Lower Bounds for Functions with Many Optima

Previous black-box complexity results like Theorem 6 or results on (non-parallel) unbiased black-box complexity [45] were limited to functions with a unique optimum. These results apply to popular test functions like OneMax and LO and function classes like linear functions or monotone functions [14]. However, they do not apply when considering functions with more than one optimum. Apart from tailored analyses for specific problems classes (e. g. problems from combinatorial optimisation [18]), we are not aware of any generic black-box complexity results that apply to functions with multiple optima.

Theorem 13 overcomes this limitation, yielding novel black-box complexity results for the unary unbiased black-box complexity and its λ-parallel variant across a range of problems with several global optima, including some widely studied problem classes. These black-box complexity results give general limitations that can serve as baselines for performance comparisons and guide the search for the most efficient algorithms, including those using parallelism most effectively (as demonstrated successfully for OneMax in Section V).

There are many examples of relevant problem classes to which Theorem 13 applies. The most obvious class is that of all functions with $\exp(o(n^\delta/\log n))$ optima. Note that when choosing, say, $\delta := 0.995$ then $\exp(n^{0.99}) \leq \exp(o(n^\delta/\log n))$; the reader may choose to think of the latter expression as $\exp(n^{0.99})$ as this may be easier to digest.

Following Witt [62], the mentioned function class includes problems where all optima have at most $n^\delta/\log^3 n$ ones or at most $n^\delta/\log^3 n$ zeros. This is because the number of such search points is bounded by

$$2\sum_{i=0}^{n^\delta/\log^3 n}\binom{n}{i} = O\left(n^{n^\delta/\log^3 n}\right) = \exp(o(n^\delta/\log n)), \quad (8)$$

where the last step used $n^{n^{\delta/\log^3 n}} = \exp(\Theta(n^{\delta/\log^2 n})) = \exp(o(n^{\delta/\log n}))$.

In the following we survey a number of illustrative problems that have been studied previously and for which we give the first black-box complexity results. In terms of combinatorial problems, there are a lot of well-studied problems with a property called *bit-flip symmetry*: flipping all bits gives a solution of the same fitness. This means that there are always at least two global optima. Such problems have been popular as search algorithms need to break the symmetry between good solutions [32].

Well-known examples include the function TwoMax := $\max\{\sum_{i=1}^n x_i, \sum_{i=1}^n (1-x_i)\}$ [32], which has been used as a challenging test bed in theoretical studies of diversity-preserving mechanisms [6], [7], [50]. The function H-Iff (Hierarchical If and only If) [59] consists of hierarchical building blocks that need to attain equal values in order to contribute to the fitness. It was studied theoretically [9], [35] and is frequently used in empirical studies, see, e. g. [33], [58].

In terms of classical combinatorial problems, the Vertex Colouring problem asks for an assignment of colours to vertices such that no two adjacent vertices share the same colour.

For two colours, a natural setting is to use a binary encoding for the colours of all vertices and to maximise the number of bichromatic edges (edges with differently coloured end points). A closely related setting is that of simple Ising models, where the goal is to *minimise* the number of bichromatic edges. For bipartite (that is, 2-colourable) graphs, this is identical to maximising the number of bichromatic edges as inverting one set of the bipartition turns all monochromatic edges into bichromatic ones and vice versa. Previous theoretical work includes evolutionary algorithms on ring/cycle graphs [30], the Metropolis algorithm on toroids [29] and evolutionary algorithms on binary trees [54].

Other combinatorial problems with bit-flip symmetry include cutting and selection problems. Given an undirected graph, the problems MAXCUT and MINCUT seek to partition the graph into two non-empty sets such as to maximise or minimise the number of edges running between those two sets, respectively. Using a straightforward binary encoding for all vertices, this results in bit-flip symmetry and multiple optima. Theoretical studies of evolutionary algorithms on cutting problems include Neumann, Reichel, and Skutella [49] and Sudholt [55]; the latter paper considers a simple instance of two equal-sized cliques that leads to two complementary optima. Concerning selection problems, the well-known NP hard PARTITION problem asks whether it is possible to schedule a set of n jobs on two identical machines such that both machines will have identical loads. An optimisation problem is obtained by trying to minimise the load of the fuller machine, also called the *makespan*. A straightforward encoding is used: every bit indicates which machine the corresponding job should be assigned to. Witt [61] analysed the performance of the (1+1) EA for this problem, including random instance models where job sizes are drawn randomly from a real range, according to a uniform or an exponential distribution, respectively. In both cases such instances will almost surely have two complementary optima[9].

Wegener and Witt [60] considered monotone polynomials: a sum of monomials (products of variables, e. g. $x_1x_3x_4$) with positive weights. Here 1^n is always a global optimum, but more optima can exist if there are variables that do not appear in any monomial: each such variable doubles the number of optima as it is not relevant for the fitness. Hence if there are $o(n^\delta/\log n)$ such variables then there are at most $2^{o(n^\delta/\log n)} \leq \exp(o(n^\delta/\log n))$ optima.

Jansen and Zarges [41] presented instance classes called *nearest peak functions* and *weighted nearest peak functions*. Both are defined with respect to an arbitrary number of peaks: search points with an associated height and slope. For nearest peak functions the fitness of a search point is determined by its closest peak: for the peak itself the fitness is equal to the height of the peak and for other search points the fitness decreases gradually with the distance from the peak, according to the slope of the peak. Weighted nearest peak functions are defined similarly, but all peaks are considered and higher peaks can dominate shallower peaks. This function class was introduced as a test bed allowing to create an arbitrary number of optima. It is shown in Jansen and Zarges [41] that the set of local optima is a subset of all peaks. Hence the number of peaks is an upper bound on the number of global (and local) optima. The two function classes were named Jansen-Zarges function classes in Covantes Osuna and Sudholt [7], where they were used as benchmarks for the *clearing* diversity mechanism.

Finally we consider random planted MAX-3-SAT instances as a popular benchmark model in both experimental [34] and theoretical studies [3], [19], [56]. The fitness function is the number of satisfied clauses and each clause contains exactly 3 literals (negated or non-negated variables from the set $\{x_1, \dots, x_n\}$). In this model, we fix a planted optimum x^* and generate clauses independently such that they are satisfied by x^*. This means that at least one literal needs to evaluate to `true` in x^*. The variables for each clause are chosen uniformly at random (with or without replacement) from $\{x_1, \dots, x_n\}$. We may assume that instances are generated by first deciding which of the 3 literals will match x^* and which won't. In a second step, the indices of variables will be picked. We further assume that there is at least a constant probability c_1 of a clause having one matching literal and at least a constant probability c_3 of a clause having three matching variables[10]. In this setup, x^* is a global optimum, but there may be more global optima. We argue that the number of optima is bounded if the number of clauses, m, is chosen large enough.

Consider a solution x with Hamming distance $H := H(x, x^*)$ to x^*. We argue that for any clause, the probability that the clause will be satisfied under x is $\Omega(H/n)$. If $H \leq n/2$ then with probability c_1 we will choose one matching literal and the probability that only the variable of this literal will be chosen among the H ones that differ in x and x^* is $\Omega(H(n-H)^2/n^3) = \Omega(H/n)$. Likewise, if $H > n/2$ then with probability c_3 we will choose three matching literals and the probability that they are all different in x and x^* is $\Omega(H^3/n^3) = \Omega(H/n)$. Now since all clauses are generated independently, the probability that all m clauses are satisfied under x is $(1-\Omega(H/n))^m \leq \exp(-\Omega(Hm/n))$.

Hence for all search points x with $H \geq n^\delta/\log^3 n$ the probability that x is a global optimum is at most $\exp(-\Omega(n^\delta/(\log^3 n) \cdot m/n)) = \exp(-\Omega(n \log n))$ if the number of clauses is $m = \Omega(n^{2-\delta} \log^4 n)$. In this case, the probability that any such search point will be a global optimum is at most $2^n \cdot \exp(-\Omega(n \log n)) = \exp(-\Omega(n \log n))$, a failure probability so small that it can be absorbed in the failure probabilities for our tail bounds. Now, with overwhelming probability the number of global optima is bounded by the number of search points with Hamming distance less than $n^\delta/\log^3 n$ from x^*. By (8), this number is $\exp(o(n^\delta/\log n))$.

The following theorem summarises all the above.

[9]More than two optima only exist if there are different combinations of job sizes (beyond symmetries) that add up to the same value. Since the weight of each job size is drawn from a continuous range and the number of values that could lead to equal values is finite, this almost surely never happens.

[10]This is the case in [3], [19], [56] where implicitly $c_1 = 3/7$ and $c_3 = 1/7$ and in [34] where $c_1 = 4/6$ and $c_3 = 1/6$. The latter probabilities favour clauses with only one matching literal in order not to give an obvious bias towards the values of x^*. Note that we do not care about the value of c_2 (two matching literals).

Theorem 20. *Every unary unbiased λ-parallel black-box algorithm $\mathcal{A}$ needs more than*

$$\max\left\{\frac{\lambda n}{60\ln^+\lambda}, (1-\delta)n\ln n\right\} = \Omega\left(\frac{\lambda n}{\ln^+\lambda} + n\ln n\right)$$

evaluations, with probability $1-\exp(-\Omega(n^\delta/\log n))$, to find a global optimum for all of the following settings.

1) *All functions with $\exp(o(n^\delta/\log n))$ optima.*
2) *All functions where all optima have at most $n^\delta/\log^3 n$ ones or at most $n^\delta/\log^3 n$ zeros.*
3) *TwoMax $:= \max\{\sum_{i=1}^n x_i, \sum_{i=1}^n (1-x_i)\}$.*
4) *H-Iff (Hierarchical If and only If).*
5) *Vertex colouring/Ising model problems: maximising or minimising the number of bichromatic edges when trying to colour a connected bipartite graph with 2 colours.*
6) *MinCut instances with two equal-sized cliques.*
7) *Partition instances having two symmetric optimal solutions (which almost surely applies to random instances)*
8) *Monotone polynomials with positive weights where all but $o(n^\delta/\log n)$ variables appear in at least one monomial.*
9) *Jansen-Zarges nearest peak functions and weighted nearest peak functions with $\exp(o(n^\delta/\log n))$ peaks.*
10) *Random planted Max-3-Sat instances as described above with at least $m = \Omega(n^{2-\delta}\log^4 n)$ clauses.*

The expected time also satisfies the asymptotic bound.

We remark that results on the expectation are tight for some of these problems: for TwoMax and the mentioned MinCut instances, the (1+λ) EA with adaptive mutation rates and appropriate restart schemes can find global optima in expected $O(\lambda n/(\ln^+\lambda) + n\ln n)$ fitness evaluations (this easily follows from the analysis on OneMax). Other function classes from Theorem 20 contain functions with an exponential black-box complexity, for instance the Needle function. Our results should be regarded as a general baseline that applies to all unary unbiased black-box algorithms and a wide range of problems.

B. Lower Bounds on the Time to Reach Local Optima

For many multimodal problems where the lower bounds from Theorem 20 are not tight, there is another significant application of Theorem 13. It can also be applied to bound the time until any unary unbiased black-box algorithm has found a *local* optimum, or any search point of reasonably high fitness, if the number of such points is bounded.

This includes functions with $\exp(o(n^\delta/\log n))$ *local* optima, and those where all local optima have at most $n^\delta/\log^3 n$ ones or at most $n^\delta/\log^3 n$ zeros. The latter function class includes the well-known $\textsc{Jump}_k$ functions [8], [26], where a gap of Hamming distance k has to be "jumped" to reach a global optimum, with parameter $k \le n^\delta/\log^3 n$: here all search points with k zeros are local optima, in addition to the global optimum 1^n. A similar function class $\textsc{Cliff}_d$ was used in [5], [37], [51], where the same holds for d in lieu of k; the difference between these two functions is that in the region "between" local and global optima $\textsc{Jump}_k$ has a gradient pointing back towards the local optima whereas $\textsc{Cliff}_d$ points towards the global optimum 1^n.

Functions with difficult local optima include a modified version of TwoMax used in [31]: in TwoMax$'$ $:= \max\{\sum_{i=1}^n x_i, \sum_{i=1}^n (1-x_i)\} + \prod_{i=1}^n x_i$ the point 1^n is the only global optimum and 0^n is a local optimum that is very hard to escape from. A combinatorial example of a MaxSat instance with difficult local optima was studied in the context of evolutionary algorithms in Droste, Jansen, and Wegener [27], with variables $x_1, \dots, x_n$ and clauses

$$\{(x_i \vee \overline{x_j} \vee \overline{x_k}) \mid i \neq j \neq k \neq i\} \cup \{(x_i) \mid 1 \le i \le n\}. \quad (9)$$

Here the optimum is again 1^n, and all n search points with a single 1-bit are local optima. Likewise, the MinCut instance from Theorem 20 has $O(n)$ local optima as well: all search points with exactly one 1-bit or one 0-bit are locally optimal. Sudholt [55] further presented a hard Knapsack instance with $(n+1)/2$ "small" objects of weight and value n and $(n-1)/2$ "big" objects of weight and value $n+1$. The weight limit is set to $(n+1)/2 \cdot n$, such that including all small objects yields a global optimum, but selecting all but one big object gives a local optimum. Similar as above, the number of local optima is $O(n)$.

Finally, the arguments for Jansen-Zarges function classes also hold with respect to the number of local optima.

The following theorem summarises all the above.

Theorem 21. *Every unary unbiased λ-parallel black-box algorithm $\mathcal{A}$ needs more than*

$$\max\left\{\frac{\lambda n}{60\ln^+\lambda}, (1-\delta)n\ln n\right\} = \Omega\left(\frac{\lambda n}{\ln^+\lambda} + n\ln n\right)$$

*evaluations, with probability $1-\exp(-\Omega(n^\delta/\log n))$, to find a **local** or global optimum for all of the following settings.*

1) *All functions with $\exp(o(n^\delta/\log n))$ local optima.*
2) *All functions where all local optima have at most $n^\delta/\log^3 n$ ones or at most $n^\delta/\log^3 n$ zeros.*
3) *$\textsc{Jump}_k$ functions with $k \le n^\delta/\log^3 n$.*
4) *$\textsc{Cliff}_d$ functions with $d \le n^\delta/\log^3 n$.*
5) *TwoMax $:= \max\{\sum_{i=1}^n x_i, \sum_{i=1}^n (1-x_i)\}$ as well as the modified TwoMax function TwoMax$'$ $:= \max\{\sum_{i=1}^n x_i, \sum_{i=1}^n (1-x_i)\} + \prod_{i=1}^n x_i$*
6) *MinCut instances with two equal-sized cliques.*
7) *The hard MaxSat instance from (9).*
8) *The hard Knapsack instance mentioned above.*
9) *Jansen-Zarges nearest peak functions and weighted nearest peak functions with $\exp(o(n^\delta/\log n))$ peaks.*

The expected time also satisfies the asymptotic bound.

We can even push our applications a bit further. Again using (8), there are at most $\exp(o(n^\delta/\log n))$ search points within a Hamming ball of radius $n^\delta/\log^3 n$ around any search point. If there are $\exp(o(n^\delta/\log n))$ global or local optima then the number of all search points within the union of Hamming balls around all these points is still $\exp(o(n^\delta/\log n)) \cdot \exp(o(n^\delta/\log n)) = \exp(o(n^\delta/\log n))$. Hence our main result from Theorem 13 still applies when considering the time to get to within Hamming distance $n^\delta/\log^3 n$ of any global or local optimum.

Theorem 22. *Theorem 20 and Theorem 21 still apply when replacing "to find a global optimum" with "to find any search point within Hamming distance $n^\delta / \log^3 n$ to any global optimum" in Theorem 20 and replacing "to find a local or global optimum" with "to find any search point within Hamming distance $n^\delta / \log^3 n$ to any local or global optimum" in Theorem 21.*

In particular, this implies that with overwhelming probability no unary unbiased black-box algorithm can find a search point of fitness at least $n - n^\delta / \log^3 n$ for ONEMAX, LO and TWOMAX within the stated time. In other words, the expected fitness after the stated time is $n - n^\delta / \log^3 n + o(1)$ (where the $o(1)$ term accounts for an exponentially small failure probability, in case of which the fitness could be as large as n). Such results are known as *fixed-budget results* [15], [40]. This shows that our λ-parallel black-box complexity results with tail bounds can be applied in a large variety of settings.

VIII. CONCLUSIONS AND FUTURE WORK

We have introduced the parallel unbiased black-box complexity to quantify the limits on the performance of parallel search heuristics, including offspring populations, island models and multi-start methods. We proved that *every* λ-parallel unbiased black-box algorithm needs at least $\Omega\big(\frac{\lambda n}{\ln^+ \lambda} + n \ln n\big)$ function evaluations on every function with unique optimum, and at least $\Omega\Big(\frac{\lambda n}{\ln^+(\lambda/n)} + n^2\Big)$ function evaluations on LO. Corresponding parallel times are by a factor of λ smaller. For LO and ONEMAX we identified the cut-off point for λ, above which the asymptotic number of function evaluations increases, compared to non-parallel algorithms ($\lambda = 1$). All smaller λ allow for linear speedups with regard to the parallel time. For ONEMAX this cut-off point is higher than that for the standard (1+λ) EA; optimal performance for all λ is achieved by a (1+λ) EA with an adaptive mutation rate.

In a novel and more detailed analysis we have established tail bounds showing that the lower bound $\Omega\big(\frac{\lambda n}{\ln^+ \lambda} + n \ln n\big)$ holds with overwhelming probability, for parallel and non-parallel algorithms (where $\lambda = 1$) and for finding any target set of search points we can choose. This makes it a very general, powerful and versatile statement: we obtain lower bounds on the optimisation time on functions with many optima, the time to find a local optimum, and the time to even get close to any local or global optimum. We demonstrated the usefulness of this approach by deriving the first black-box complexity lower bounds for a range of popular and illustrative problems, from synthetic problems (TWOMAX, H-IFF, JUMP$_k$, CLIFF) to classes of multimodal benchmark functions [41] and important problems from combinatorial optimisation such as VERTEX COLOURING, MINCUT, PARTITION, KNAPSACK and MAXSAT.

A major open problem for future work is to derive lower bounds for the λ-parallel unbiased black-box complexity when allowing binary operators like crossover, or operators combining many search points as in EDAs or swarm intelligence algorithms. Currently even in the non-parallel case no non-trivial lower bounds on the binary unbiased black-box complexity are known.

REFERENCES

[1] G. Badkobeh, P. K. Lehre, and D. Sudholt, "Unbiased black-box complexity of parallel search," in *Proc. of PPSN '14*, Springer, 2014, pp. 892–901.

[2] ——, "Black-box complexity of parallel search with distributed populations," in *Proc. of FOGA '15*, ACM, 2015, pp. 3–15.

[3] M. Buzdalov and B. Doerr, "Runtime analysis of the (1+(λ,λ)) genetic algorithm on random satisfiable 3-CNF formulas," in *Proc. of GECCO '17*, ACM, 2017, pp. 1343–1350.

[4] V Chvátal, "The tail of the hypergeometric distribution," *Discrete Math.*, vol. 25, no. 3, pp. 285–287, 1979.

[5] D. Corus, P. S. Oliveto, and D. Yazdani, "On the runtime analysis of the opt-IA artificial immune system," in *Proc. of GECCO '17*, ACM, 2017, pp. 83–90.

[6] E. Covantes Osuna and D. Sudholt, "Runtime analysis of probabilistic crowding and restricted tournament selection for bimodal optimisation," in *Proc. of GECCO '18*, ACM, 2018, pp. 929–936.

[7] ——, "On the runtime analysis of the clearing diversity-preserving mechanism," *Evolutionary Computation*, vol. 27, no. 3, pp. 403–433, 2019.

[8] D.-C. Dang, T. Friedrich, T. Kötzing, M. S. Krejca, P. K. Lehre, P. S. Oliveto, D. Sudholt, and A. M. Sutton, "Escaping local optima using crossover with emergent diversity," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 3, pp. 484–497, 2018.

[9] M. Dietzfelbinger, B. Naudts, C. V. Hoyweghen, and I. Wegener, "The analysis of a recombinative hill-climber on H-IFF," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 5, pp. 417–423, 2003.

[10] B. Doerr, E. Happ, and C. Klein, "Tight analysis of the (1+1)-EA for the single source shortest path problem," *Evolutionary Computation*, vol. 19, no. 4, pp. 673–691, 2011.

[11] B. Doerr, C. Doerr, and F. Ebel, "From Black-Box Complexity to Designing New Genetic Algorithms," *Theoretical Computer Science*, vol. 567, no. 0, pp. 87–104, 2015.

[12] B. Doerr, C. Doerr, and J. Yang, "Optimal parameter choices via precise black-box analysis," in *Proc. of GECCO '16*, ACM, 2016, pp. 1123–1130.

[13] B. Doerr, C. Gießen, C. Witt, and J. Yang, "The (1+λ) evolutionary algorithm with self-adjusting mutation rate," *Algorithmica*, vol. 81, no. 2, pp. 593–631, 2019.

[14] B. Doerr, T. Jansen, D. Sudholt, C. Winzen, and C. Zarges, "Mutation rate matters even when optimizing monotonic functions," *Evolutionary Computation*, vol. 21, no. 1, pp. 1–21, 2013.

[15] B. Doerr, T. Jansen, C. Witt, and C. Zarges, "A method to derive fixed budget results from expected optimisation times," in *Proc. of GECCO '13*, ACM, 2013, pp. 1581–1588.

[16] B. Doerr, D. Johannsen, T. Kötzing, P. K. Lehre, M. Wagner, and C. Winzen, "Faster black-box algorithms through higher arity operators," in *Proc. of FOGA '11*, ACM, 2011, pp. 163–172.

[17] B. Doerr and M. Künnemann, "Optimizing linear functions with the (1+λ) evolutionary algorithm—different asymptotic runtimes for different instances," *Theoretical Computer Science*, vol. 561, pp. 3–23, 2015.

[18] B. Doerr, T. Kötzing, J. Lengler, and C. Winzen, "Black-box complexities of combinatorial problems," *Theoretical Computer Science*, vol. 471, pp. 84–106, 2013.

[19] B. Doerr, F. Neumann, and A. M. Sutton, "Improved runtime bounds for the (1+1) EA on random 3-CNF formulas based on fitness-distance correlation," in *Proc. of GECCO '15*, ACM, 2015, pp. 1415–1422.

[20] B. Doerr and C. Winzen, "Towards a complexity theory of randomized search heuristics: Ranking-based black-box complexity," in *Proc. of CSR '11*, Springer, 2011, pp. 15–28.

[21] ——, "Playing Mastermind with Constant-Size Memory," *Theory of Computing Systems*, 2012.

[22] C. Doerr, "Complexity theory for discrete black-box optimization heuristics," *CoRR*, vol. abs/1801.02037, 2018.

[23] C. Doerr and J. Lengler, "Elitist black-box models: Analyzing the impact of elitist selection on the performance of evolutionary algorithms," in *Proc. of GECCO '15*, ACM, 2015, pp. 839–846.

[24] ——, "Introducing elitist black-box models: When does elitist behavior weaken the performance of evolutionary algorithms?" *Evolutionary Computation*, vol. 25, no. 4, pp. 587–606, 2017.

[25] ——, "The $(1 + 1)$ elitist black-box complexity of LeadingOnes," *Algorithmica*, vol. 80, no. 5, pp. 1579–1603, 2018.

[26] S. Droste, T. Jansen, and I. Wegener, "On the analysis of the (1+1) evolutionary algorithm," *Theoretical Computer Science*, vol. 276, no. 1–2, pp. 51–81, 2002.

[27] ——, "Optimization with randomized search heuristics—the (A)NFL theorem, realistic scenarios, and difficult functions," *Theoretical Computer Science*, vol. 287, no. 1, pp. 131–144, 2002.

[28] ——, "Upper and lower bounds for randomized search heuristics in black-box optimization," *Theory of Computing Systems*, vol. 39, no. 4, pp. 525–544, 2006.

[29] S. Fischer, "A polynomial upper bound for a mutation-based algorithm on the two-dimensional Ising model," in *Proc. of GECCO '04*, Springer, 2004, pp. 1100–1112.

[30] S. Fischer and I. Wegener, "The one-dimensional Ising model: Mutation versus recombination," *Theoretical Computer Science*, vol. 344, no. 2–3, pp. 208–225, 2005.

[31] T. Friedrich, P. S. Oliveto, D. Sudholt, and C. Witt, "Analysis of diversity-preserving mechanisms for global exploration," *Evolutionary Computation*, vol. 17, no. 4, pp. 455–476, 2009.

[32] D. E. Goldberg, C. Van Hoyweghen, and B. Naudts, "From TwoMax to the Ising model: Easy and hard symmetrical problems," in *Proc. of GECCO '02*, Morgan Kaufmann, 2002, pp. 626–633.

[33] B. W. Goldman and W. F. Punch, "Fast and efficient black box optimization using the parameter-less population pyramid," *Evolutionary Computation*, vol. 23, no. 3, pp. 451–479, 2015.

[34] B. W. Goldman and W. F. Punch, "Parameter-less population pyramid," in *Proc. of GECCO '14*, ACM, 2014, pp. 785–792.

[35] B. W. Goldman and D. Sudholt, "Runtime analysis for the parameter-less population pyramid," in *Proc. of GECCO '16*, ACM, 2016, pp. 669–676.

[36] J. He and X. Yao, "A Study of Drift Analysis for Estimating Computation Time of Evolutionary Algorithms," *Natural Computing*, vol. 3, no. 1, pp. 21–35, 2004.

[37] J. Jägersküpper and T. Storch, "When the plus strategy outperforms the comma strategy and when not," in *Proc. of FOCI '07*, IEEE, 2007, pp. 25–32.

[38] T. Jansen, K. A. De Jong, and I. Wegener, "On the choice of the offspring population size in evolutionary algorithms," *Evolutionary Computation*, vol. 13, pp. 413–440, 4 2005.

[39] T. Jansen and D. Sudholt, "Analysis of an asymmetric mutation operator," *Evolutionary Computation*, vol. 18, no. 1, pp. 1–26, 2010.

[40] T. Jansen and C. Zarges, "Performance analysis of randomised search heuristics operating with a fixed budget," *Theoretical Computer Science*, vol. 545, pp. 39–58, 2014.

[41] ——, "Example landscapes to support analysis of multimodal optimisation," in *Proc. of PPSN XIV*, Springer, 2016, pp. 792–802.

[42] D. Johannsen, "Random combinatorial structures and randomized search heuristics," PhD thesis, Universität des Saarlandes, Saarbrücken, Germany and the Max-Planck-Institut für Informatik, 2010.

[43] J. Lässig and D. Sudholt, "Analysis of speedups in parallel evolutionary algorithms for combinatorial optimization," in *Proc. of ISAAC 2011*, Springer, 2011, pp. 405–414.

[44] ——, "General upper bounds on the runtime of parallel evolutionary algorithms," *Evolutionary Computation*, vol. 22, no. 3, pp. 405–437, Nov. 2013.

[45] P. K. Lehre and C. Witt, "Black-box search by unbiased variation," *Algorithmica*, vol. 64, no. 4, pp. 623–642, 2012.

[46] ——, "Concentrated hitting times of randomized search heuristics with variable drift," in *Proc. of ISAAC '14*, https://arxiv.org/abs/1307.2559, 2014, pp. 686–697.

[47] G. Luque and E. Alba, *Parallel Genetic Algorithms–Theory and Real World Applications*. Springer, 2011.

[48] A. Mambrini, D. Sudholt, and X. Yao, "Homogeneous and heterogeneous island models for the set cover problem," in *Proc. of PPSN '12*, Springer, 2012, pp. 11–20.

[49] F. Neumann, J. Reichel, and M. Skutella, "Computing minimum cuts by randomized search heuristics," *Algorithmica*, vol. 59, no. 3, pp. 323–342, 2011.

[50] P. S. Oliveto, D. Sudholt, and C. Zarges, "On the benefits and risks of using fitness sharing for multimodal optimisation," *Theoretical Computer Science*, vol. 773, pp. 53–70, 2019.

[51] T. Paixão, J. Pérez Heredia, D. Sudholt, and B. Trubenová, "Towards a runtime comparison of natural and artificial evolution," *Algorithmica*, vol. 78, no. 2, pp. 681–713, 2017.

[52] J. E. Rowe and D. Sudholt, "The choice of the offspring population size in the (1,λ) evolutionary algorithm," *Theoretical Computer Science*, vol. 545, pp. 20–38, 2014.

[53] J. E. Rowe and M. D. Vose, "Unbiased black box search algorithms," in *Proc. of GECCO '11*, ACM, 2011, pp. 2035–2042.

[54] D. Sudholt, "Crossover is provably essential for the Ising model on trees," in *Proc. of GECCO '05*, ACM, 2005, pp. 1161–1167.

[55] ——, "Hybridizing evolutionary algorithms with variable-depth search to overcome local optima," *Algorithmica*, vol. 59, no. 3, pp. 343–368, 2011.

[56] A. M. Sutton and F. Neumann, "Runtime analysis of evolutionary algorithms on randomly constructed high-density satisfiable 3-cnf formulas," in *Proc. of PPSN '14*, Springer, 2014, pp. 942–951.

[57] O. Teytaud and S. Gelly, "General lower bounds for evolutionary algorithms," *Proc. of PPSN IX*, vol. 8623, pp. 21–31, 2006.

[58] D. Thierens and P. A. N. Bosman, "Hierarchical problem solving with the linkage tree genetic algorithm," in *Proc. of GECCO '13*, ACM, 2013, pp. 877–884.

[59] R. A. Watson, G. S. Hornby, and J. B. Pollack, "Modeling building-block interdependency," in *Proc. of PPSN V*, Springer, 1998, pp. 97–106.

[60] I. Wegener and C. Witt, "On the optimization of monotone polynomials by simple randomized search heuristics," *Combinatorics, Probability and Computing*, vol. 14, no. 1–2, pp. 225–247, 2005.

[61] C. Witt, "Worst-case and average-case approximations by simple randomized search heuristics," in *Proc. of STACS '05*, Springer, 2005, pp. 44–56.

[62] ——, "Runtime analysis of the (μ+1) EA on simple pseudo-Boolean functions," *Evolutionary Computation*, vol. 14, no. 1, pp. 65–86, 2006.

[63] C. Zarges, "Rigorous runtime analysis of inversely fitness proportional mutation rates," in *Proc. of PPSN X*, Springer, 2008, pp. 112–122.

[64] ——, "On the utility of the population size for inversely fitness proportional mutation rates," in *Proc. of FOGA 2009*, ACM, 2009, pp. 39–46.

Per Kristian Lehre is a Senior Lecturer at the University of Birmingham, UK.

He received MSc and PhD degrees in Computer Science from the Norwegian University of Science and Technology (NTNU). After finishing his PhD in 2006, he held postdoctorial positions in the School of Computer Science at the University of Birmingham and at the Technical University of Denmark. From 2011, he was a Lecturer in the School of Computer Science at the University of Nottingham, until 2017, when he returned to Birmingham.

Dr Lehre's research interests are in theoretical aspects of nature-inspired search heuristics, in particular, runtime analysis of population-based evolutionary algorithms. His research has won several best paper awards, including at GECCO (2013, 2010, 2009, 2006), ICSTW (2008), and ISAAC (2014). He is editorial board member of Evolutionary Computation, and associate editor of IEEE Transactions on Evolutionary Computation. He was the coordinator of the successful 2M euro EU-funded project SAGE which brought together the theory of evolutionary computation and population genetics.

Dirk Sudholt received his Master's and Ph.D. degrees in computer science, under the supervision of Prof. I. Wegener, from the Technische Universität Dortmund, Dortmund, Germany, in 2004 and 2008, respectively. He is a Senior Lecturer and Head of the Algorithms Group at the University of Sheffield, Sheffield, U.K. He has held Post-Doctoral positions at the International Computer Science Institute, Berkeley, CA, USA, and the University of Birmingham, Birmingham, U.K. He has more than 100 refereed publications. His current research interests include runtime analysis of randomized search heuristics, such as evolutionary algorithms and swarm intelligence. Dr. Sudholt was a recipient of the EU's Future and Emerging Technologies Scheme (SAGE Project) and eight best paper awards at GECCO and PPSN. He is an Editorial Board Member of Evolutionary Computation, Natural Computing and The Computer Journal.