The Univariate Marginal Distribution Algorithm Copes Well With Deception and Epistasis

Benjamin Doerr[1] and Martin S. Krejca[2]

[1] Laboratoire d'Informatique (LIX), CNRS, École Polytechnique, Institute Polytechnique de Paris, Palaiseau, France
doerr@lix.polytechnique.fr
[2] Hasso Plattner Institute, University of Potsdam, Potsdam, Germany
martin.krejca@hpi.de

Abstract In their recent work, Lehre and Nguyen (FOGA 2019) show that the univariate marginal distribution algorithm (UMDA) needs time exponential in the parent populations size to optimize the DeceivingLeadingBlocks (DLB) problem. They conclude from this result that univariate EDAs have difficulties with deception and epistasis.
In this work, we show that this negative finding is caused by an unfortunate choice of the parameters of the UMDA. When the population sizes are chosen large enough to prevent genetic drift, then the UMDA optimizes the DLB problem with high probability with at most $\lambda(\frac{n}{2} + 2e \ln n)$ fitness evaluations. Since an offspring population size λ of order $n \log n$ can prevent genetic drift, the UMDA can solve the DLB problem with $O(n^2 \log n)$ fitness evaluations. In contrast, for classic evolutionary algorithms no better run time guarantee than $O(n^3)$ is known, so our result rather suggests that the UMDA can cope well with deception and epistatis.
Together with the result of Lehre and Nguyen, our result for the first time rigorously proves that running EDAs in the regime with genetic drift can lead to drastic performance losses.

Keywords: Estimation-of-distribution algorithm · Univariate marginal distribution algorithm · Run time analysis · Epistasis · Theory

1 Introduction

Estimation-of-distribution algorithms (EDAs) are randomized search heuristics that evolve a probabilistic model of the search space in an iterative manner. Starting with the uniform distribution, an EDA takes samples from its current model and then adjusts it such that better solutions have a higher probability of being generated in the next iteration. This method of refinement leads to gradually better solutions and performs well on many practical problems, often outperforming competing approaches [21].

Theoretical analyses of EDAs also often suggest an advantage of EDAs when compared to evolutionary algorithms (EAs); for an in-depth survey of run time results for EDAs, please refer to the article by Krejca and Witt [16]. With respect

to simple unimodal functions, EDAs seem to be comparable to EAs. For example, Sudholt and Witt [22] proved that the two EDAs cGA and 2-MMAS$_{\text{ib}}$ have an expected run time of $\Theta(n \log n)$ on the standard theory benchmark function OneMax (assuming optimal parameter settings; n being the problem size), which is a run time that many EAs share. The same is true for the EDA UMDA, as shown by the results of Krejca and Witt [15], Lehre and Nguyen [17], and Witt [24]. For the benchmark function LeadingOnes, Dang and Lehre [4] proved an expected run time of $O(n^2)$ for the UMDA when setting the parameters right, which is, again, a common run time bound for EAs on this function. One result suggesting that EDAs can outperform EAs on unimodal function was given by Doerr and Krejca [9]. They proposed an EDA called sig-cGA, which has an expected run time of $O(n \log n)$ on both OneMax and LeadingOnes – a performance not known for any classic EA or EDA.

For the class of all linear functions, the EDAs perform slightly worse than EAs. The classical $(1+1)$ evolutionary algorithm optimizes all linear functions in time $O(n \log n)$ in expectation [12]. In contrast, the conjecture of Droste [11] that the cGA does not perform equally well on all linear functions was recently proven by Witt [23], who showed that the cGA has an $\Omega(n^2)$ expected run time on the binary value function. We note that the binary value function was found harder also for classical EAs. While the $(1+\lambda)$ evolutionary algorithm optimizes OneMax with $\Theta(n\lambda \log\log \lambda / \log \lambda)$ fitness evaluations, it takes $\Theta(n\lambda)$ fitness evaluations for the binary value functions [8].

For the bimodal Jump$_k$ benchmark function, which has a local optimum with a Hamming distance of k away from the global optimum, EDAs seem to drastically outperform EAs. Hasenöhrl and Sutton [13] recently proved that the cGA only has a run time of $\exp\big(O(k + \log n)\big)$. In contrast, common EAs have a run time of $\Theta(n^k)$ [12] (mutation-only) or $\Theta(n^{k-1})$ [3] (mutation and crossover), and only get to $O(n \log n + kn + 4^k)$ by using crossover in combination with diversity mechanisms like Island models [2]. Doerr [6] proved that the cGA even has an expected run time of $O(n \log n)$ on Jump$_k$ if $k < \frac{1}{20} \ln n$, meaning that the cGA is unfazed by the gap of k separating the local from the global optimum.

Another result in favor of EDAs was given by Chen et al. [1], who introduced the Substring function and proved that the UMDA optimizes it in polynomial time, whereas the $(1+1)$ evolutionary algorithm has an exponential run time, both with high probability. In the Substring function, only substrings of length αn, for $\alpha \in (0,1)$, of the global optimum are relevant to the fitness of a solution, and these subtrings provide a gradient to the optimum. In the process, the $(1+1)$ evolutionary algorithm loses bits that are not relevant anymore for following the gradient (but relevant for the optimum). The UMDA fixes its model for correct positions while it is following the gradient and thus does not lose these bits.

The first, and so far only, result to suggest that EDAs can be drastically worse than EAs was recently stated by Lehre and Nguyen [18] via the DeceivingLeadingBlocks function (DLB for short), which they introduce and which consists of blocks of size 2, each with a deceiving function value, that need to be solved sequentially. The authors prove that many common EAs optimize DLB within

$O(n^3)$ fitness evaluations in expectation, whereas the UMDA has a run time of $e^{\Omega(\mu)}$ (where μ is an algorithm-specific parameter that often is chosen as a small power of n) for a large regime of parameters. Only for extreme parameter values, the authors prove an expected run time of $O(n^3)$ also for the UMDA.

In this paper, we prove that the UMDA is, in fact, able to optimize DLB in time $O(n^2 \log n)$ with high probability if its parameters are chosen more carefully (Theorem 5). Note that our result is better than any of the run times proven in the paper by Lehre and Nguyen [18]. We achieve this run time by choosing the parameters of the UMDA such that its model is unlikely to degenerate during the run time (Lemma 6). Here by *degenerate* we mean that the sampling frequencies approach the boundary values 0 and 1 without that this is justified by the objective function. This leads to a probabilistic model that is strongly concentrated around a single search point. This effect is often called *genetic drift* [22]. While it appears natural to choose the parameters of an EDA as to prevent genetic drift, it also has been proven that genetic drift can lead to a complicated run time landscape and inferior performance (see Lengler et al. for the cGA [19]).

In contrast to our setting, for their exponential lower bound, Lehre and Nguyen [18] use parameters that lead to genetic drift. Once the probabilistic model it is sufficiently degenerated, the progress of the UMDA is so slow that even to leave the local optima of DLB (that have a better search point in Hamming distance two only), the EDA takes time exponential in μ.

Since the UMDA shows a good performance in the (more natural) regime without genetic drift and was shown inferior only in the regime with genetic drift, we disagree with the statement of Lehre and Nguyen [18] that there are "inherent limitations of univariate EDAs against deception and epistasis".

In addition to the improved run time, we derive our result using only tools commonly used in the analysis of EDAs and EAs, whereas the proof of the polynomial run time of $O(n^3)$ for the UMDA with uncommon parameter settings [18] uses the level-based population method, which is an advanced tool that can be hard to use. We are thus optimistic that our analysis method can be useful also in other run time analyses of EDAs.

Last, we complement our theoretical result with an empirical comparison of the UMDA to two other evolutionary algorithms. The outcome of these experiments suggests that the UMDA outperforms the competing approaches while also having a smaller variance.

The remainder of this paper is structured as follows: in Section 2, we introduce our notation, formally define DLB and the UMDA, and we state the tools we use in our analysis. Section 3 contains our main result (Theorem 5) and discusses its proof informally before stating the different lemmas used to prove it. In Section 4, we discuss our empirical results. Last, we conclude this paper in Section 5.

2 Preliminaries

We are concerned with the run time analysis of algorithms optimizing pseudo-Boolean functions, that is, functions $f\colon \{0,1\}^n \to \mathbb{R}$, where $n \in \mathbb{N}$ denotes the dimension of the problem. Given a pseudo-Boolean function f and a bit string x, we refer to f as a *fitness function,* to x as an *individual,* and to $f(x)$ as the *fitness of* x.

For $n_1, n_2 \in \mathbb{N} \coloneqq \{0,1,2,\dots\}$, we define $[n_1..n_2] = [n_1, n_2] \cap \mathbb{N}$, and for an $n \in \mathbb{N}$, we define $[n] = [1..n]$. From now on, if not stated otherwise, the variable n always denotes the problem size. For a vector x of length n, we denote its component at index $i \in [n]$ by x_i and, for and index set $I \subseteq [n]$, we denote the subvector of length $|I|$ consisting only of the components at indices in I by x_I. Further, let $|x|_1$ denote the number of 1s of x and $|x|_0$ its number of 0s.

DeceivingLeadingBlocks. The pseudo-Boolean function DeceivingLeadingBlocks (abbreviated as DLB) was introduced by Lehre and Nguyen [18] as a deceptive version of the well known benchmark function LeadingOnes. In DLB, an individual x of length n is divided into blocks of equal size 2. Each block consists of a trap, where the fitness of each block is determined by the number of 0s (minus 1), except that a block of all 1s has the best fitness of 2. The overall fitness of x is then determined by the longest prefix of blocks with fitness 2 plus the fitness of the following block. Note that in order for the chunking of DLB to make sense, it needs to hold that 2 divides n. In the following, we always assume this implicitly.

We now provide a formal definition of DLB. To this end, we first introduce the function DeceivingBlock$\colon \{0,1\}^2 \to [0..2]$ (abbreviated as DB), which determines the fitness of a block (of size 2). For all $x \in \{0,1\}^2$, we have

$$\mathrm{DB}(x) = \begin{cases} 2 & \text{if } |x|_1 = 2, \\ |x|_0 - 1 & \text{else.} \end{cases}$$

Further, we define the function Prefix$\colon \{0,1\}^n \to [0..n]$, which determines the longest prefix of x with blocks of fitness 2. For a logic formula P, let $[P]$ be 1 if P is true and 0 otherwise. We define, for all $x \in \{0,1\}^n$,

$$\textsc{Prefix}(x) = \sum_{i=1}^{n/2} \big[\forall j \leq i\colon \mathrm{DB}(x_{\{2i-1,2i\}}) = 2\big].$$

DLB is now defined as follows for all $x \in \{0,1\}^n$:

$$\mathrm{DLB}(x) = \begin{cases} n & \text{if } \textsc{Prefix}(x) = n, \\ \sum_{i=1}^{\textsc{Prefix}(x)+1} \mathrm{DB}(x_{\{2i-1,2i\}}) & \text{else.} \end{cases}$$

The univariate marginal distribution algorithm. Our algorithm of interest is the UMDA ([20]; Algorithm 1) with parameters $\mu, \lambda \in \mathbb{N}^+$, $\mu \leq \lambda$. It maintains a vector p *(frequency vector)* of probabilities *(frequencies)* of length n as its probabilistic model. This vector is used to sample an individual $x \in \{0,1\}^n$, which we denote as $x \sim \text{sample}(p)$, such that, for all $y \in \{0,1\}^n$,

$$\Pr[x = y] = \prod_{\substack{i=1:\\ y_i=1}}^{n} p_i \prod_{\substack{i=1:\\ y_i=0}}^{n} (1 - p_i).$$

The UMDA updates this vector iteratively in the following way: first, λ individuals are sampled. Then, among these λ individuals, a subset of μ with the highest fitness is chosen (breaking ties uniformly at random), and, for each index $i \in [n]$, the frequency p_i is set to the relative number of 1s at position i among the μ best individuals. Last, if a frequency p_i is below $\frac{1}{n}$, it is increased to $\frac{1}{n}$, and, analogously, frequencies above $1 - \frac{1}{n}$ are set to $1 - \frac{1}{n}$. Capping into the interval $[\frac{1}{n}, 1 - \frac{1}{n}]$ circumvents frequencies from being stuck at the extremal values 0 or 1. Last, we denote the frequency vector of iteration $t \in \mathbb{N}$ with $p^{(t)}$.

Algorithm 1: The UMDA [20] with parameters μ and λ, $\mu \leq \lambda$, maximizing a fitness function $f\colon \{0,1\}^n \to \mathbb{R}$ with $n \geq 2$

$t \leftarrow 0$;
$p^{(t)} \leftarrow (\frac{1}{2})_{i \in [n]}$;
repeat ▷ *iteration* t
  for $i \in [\lambda]$ **do** $x^{(i)} \sim \text{sample}\big(p^{(t)}\big)$;
  let $y^{(1)}, \ldots, y^{(\mu)}$ denote the μ best individuals out of $x^{(1)}, \ldots, x^{(\lambda)}$ (breaking ties uniformly at random);
  for $i \in [n]$ **do** $p_i^{(t+1)} \leftarrow \frac{1}{\mu} \sum_{j=1}^{\mu} y_i^{(j)}$;
  restrict $p^{(t+1)}$ to the interval $[\frac{1}{n}, 1 - \frac{1}{n}]$;
  $t \leftarrow t + 1$;
until termination criterion met;

Run time analysis. When analyzing the run time of the UMDA optimizing a fitness function f, we are interested in the number T of fitness function evaluations until an optimum of f is sampled for the first time. Since the UMDA is a randomized algorithm, this run time T is a random variable. Note that the run time of the UMDA is at most λ times the number of iterations until an optimum is sampled for the first time, and it is at least $(\lambda - 1)$ times this number.

In the area of run time analysis of randomized search heuristics, it is common to give bounds for the expected value of the run time of the algorithm under investigation. This is uncritical when the run time is concentrated around its expectation, as often observed for classical evolutionary algorithms. For EDAs, it

has been argued, among others in [6], that it is preferable to give bounds that hold with high probability. This is what we shall aim at in this work as well. Of course, it would be even better to give estimates in a distributional sense, as argued for in [5], but this appears to be difficult for EDAs, among others, because of the very different behavior in the regimes with and without strong genetic drift.

Probabilistic tools. We use the following results in our analysis.

In order to prove statements on random variables that hold with high probability, we use the following commonly known Chernoff bound.

Theorem 1 (Chernoff bound [7, Theorem 10.5], [14]). *Let $k \in \mathbb{N}$, $\delta \in [0,1]$, and let X be the sum of k independent random variables, each taking values in $[0,1]$. Then*

$$\Pr\big[X \leq (1-\delta)\mathrm{E}[X]\big] \leq \exp\left(-\frac{\delta^2 \mathrm{E}[X]}{2}\right).$$

The next lemma tells us that, for a random X following a binomial law, the probability of exceeding $\mathrm{E}[X]$ is bounded from above by roughly the term with the highest probability.

Lemma 2 ([7, Eq. (10.62)]). *Let $k \in \mathbb{N}$, $p \in [0,1]$, $X \sim \mathrm{Bin}(k,p)$, and let $m \in \big[\mathrm{E}[X]+1..k\big]$. Then*

$$\Pr[X \geq m] \leq \frac{m(1-p)}{m-\mathrm{E}[X]} \cdot \Pr[X = m].$$

We use Lemma 2 for the converse case, that is, in order to bound the probability that a binomially distributed random variable is smaller than its expected value.

Corollary 3. *Let $k \in \mathbb{N}$, $p \in [0,1]$, $X \sim \mathrm{Bin}(k,p)$, and let $m \in \big[0..\mathrm{E}[X]-1\big]$. Then*

$$\Pr[X \leq m] \leq \frac{(k-m)p}{\mathrm{E}[X]-m} \cdot \Pr[X = m].$$

Proof. Let $\overline{X} \coloneqq k - X$, and let $\overline{m} \coloneqq k - m$. Note that $\overline{X} \sim \mathrm{Bin}(k, 1-p)$ with $\mathrm{E}[\overline{X}] = k - \mathrm{E}[X]$ and that $\overline{m} \in \big[\mathrm{E}[\overline{X}]+1..k\big]$. With Lemma 2, we compute

$$\Pr[X \leq m] = \Pr[\overline{X} \geq \overline{m}] \leq \frac{\overline{m}p}{\overline{m}-\mathrm{E}[\overline{X}]} \cdot \Pr[\overline{X} = \overline{m}] = \frac{(k-m)p}{\mathrm{E}[X]-m} \cdot \Pr[X = m],$$

which proves the claim. □

Last, the following theorem deals with a *neutral* bit in a fitness function f, that is, a position $i \in [n]$ such that bit values at i do not contribute to the fitness value at all. The theorem (from [10], extending a similar result [25, Theorem 4.5]) states that if the UMDA optimizes such an f, then the frequency at position i stays close to its initial value $\frac{1}{2}$ for $\Omega(\mu)$ iterations. We go more into detail about how this relates to DLB at the beginning of Section 3.

Theorem 4 ([10, Corollary 2]). *Consider the UMDA optimizing a fitness function f with a neutral bit $i \in [n]$. Then, for all $d > 0$ and all $t \in \mathbb{N}$, we have*

$$\Pr\left[\forall t' \in [0..t] \colon |p_i^{(t')} - \tfrac{1}{2}| < d\right] \geq 1 - 2\exp\left(-\frac{d^2\mu}{2t}\right).$$

3 Run Time Analysis

In the following, we prove that the UMDA optimizes DeceivingLeadingBlocks efficiently, which is the following theorem.

Theorem 5. *Let $c_\mu, c_\lambda \in (0,1)$ be constants chosen sufficiently large or small, respectively. Consider the UMDA optimizing* DeceivingLeadingBlocks *with $\mu \geq c_\mu n \ln n$ and $\mu/\lambda \leq c_\lambda$. Then the UMDA samples the optimum after $\lambda(\frac{n}{2} + 2e \ln n)$ fitness function evaluations with a probability of at least $1 - 9n^{-1}$.*

Before we present the proof, we sketch its main ideas and introduce important notation. We show that the frequencies of the UMDA are set to $1 - \frac{1}{n}$ block-wise from left to right with high probability. We formalize this concept by defining that a block $i \in [\frac{n}{2}]$ is *critical* (in iteration t) if and only if $p_{2i-1}^{(t)} + p_{2i}^{(t)} < 2 - \frac{2}{n}$ and, for each index $j \in [2i-2]$, the frequency $p_j^{(t+1)}$ is at $1 - \frac{1}{n}$. Intuitively, a critical block is the first block whose frequencies are not at their maximum value. We prove that a critical block is optimized within a single iteration with high probability if we assume that its frequencies are not below $(1-\varepsilon)/2$, for $\varepsilon \in (0,1)$ being a constant.

In order to assure that the frequencies of each block are at least $(1-\varepsilon)/2$ until it becomes critical, we show that most of the frequencies right of the critical block are not impacted by the fitness function. We call such frequencies *neutral*. More formally, a frequency p_i is neutral in iteration t if and only if the probability to have a 1 at position i in each of the μ *selected* individuals equals $p_i^{(t)}$. Note that since we assume that $\mu = \Omega(n \log n)$, the impact of the genetic drift on neutral frequencies is low with high probability (Theorem 4).

We know which frequencies are neutral and which are not by the following key observation: consider a population of λ individuals of the UMDA during iteration t; only the first (leftmost) block that has strictly fewer than μ 11s is relevant for selection, since the fitness of individuals that do not have a 11 in this block cannot be changed by bits to the right anymore. We call this block *selection-relevant*. Note that this is a notion that depends on the random offspring population in iteration t, whereas the notion *critical* depends only on $p^{(t)}$.

The consequences of a selection-relevant block are as follows: if block $i \in [\frac{n}{2}]$ is selection-relevant, then all frequencies in blocks left of i are set to $1 - \frac{1}{n}$, since there are at least μ individuals with 11s. All blocks right of i have *no* impact on the selection process: if an individual has no 11 in block i, its fitness is already fully determined by all of its bits up to block i by the definition of DLB. If an individual has a 11 in block i, it is definitely chosen during selection, since there are fewer than μ such individuals and since its fitness is better than that of all of the other individuals that do not have a 11 in block i. Thus, its bits at positions in blocks right of i are irrelevant for selection. Overall, since the bits in blocks right of i do not matter, the frequencies right of block i get no signal from the fitness function and are thus neutral (Lemma 6).

Regarding block i itself, all of the individuals with 11s are chosen, since they have the best fitness. Nonetheless, individuals with a 00, 01, or 10 can also be chosen, where an individual with a 00 in block i is preferred, as a 00 has the second best fitness after a 11. Since the fitness for a 10 or 01 is the same, this does not impact the number of 1s at position i in expectation. However, if more 00s than 11s are sampled for block i, it can happen that the frequencies of block i are decreased. Since we assume that $\mu = \Omega(n \log n)$, the frequency is sufficiently high before the update and the frequencies of block i do not decrease by much with high probability (Lemma 8). Since, in the next iteration, block i is the critical block, it is then optimized within a single iteration (Lemma 9), and we do not need to worry about its frequencies decreasing again.

Neutral frequencies. We now prove that the frequencies right of the selection-relevant block do not decrease by too much within the first n iterations.

Lemma 6. *Let $\varepsilon \in (0,1)$ be a constant. Consider the UMDA with $\lambda \geq \mu \geq (16n/\varepsilon^2)\log n$ optimizing* DeceivingLeadingBlocks. *Let $t \leq n$ be the first iteration such that block $i \in [\frac{n}{2}]$ becomes selection-relevant for the first time. Then, with a probability of at least $1 - 2n^{-1}$, all frequencies at the positions $[2i+1..n]$ are at least $(1-\varepsilon)/2$ within the first t iterations.*

Proof. Let $j \in [2i+1..n]$ denote the index of a frequency right of block i. Note that by the assumption that t is the first iteration such that block i becomes selection-relevant it follows that, for all $t' \leq t$, the frequency $p_i^{(t')}$ is neutral, as we discussed above.

Since $p_j^{(t')}$ is neutral for all $t' \leq t$, by Theorem 4 with $d = \frac{\varepsilon}{2}$, we see that the probability that p_j leaves the interval $\big((1-\varepsilon)/2, (1+\varepsilon)/2\big)$ within the first $t \leq n$ iterations is at most $2\exp\big(-\varepsilon^2\mu/(8t)\big) \leq 2\exp\big(-\varepsilon^2\mu/(8n)\big) \leq 2n^{-2}$, where we used our bound on μ.

Applying a union bound over all $n - 2i \leq n$ neutral frequencies yields that at least one frequency leaves the interval $\big((1-\varepsilon)/2, (1+\varepsilon)/2\big)$ within the first t iterations with a probability of at most $2n^{-1}$, as desired. □

Update of the selection-relevant block. As mentioned at the beginning of the section, while frequencies right of the selection-relevant block do not drop

below $(1-\varepsilon)/2$ with high probability (by Lemma 6), the frequencies of the selection-relevant block can drop below $(1-\varepsilon)/2$, as the following example shows.

Example 7. Consider the UMDA with $\mu = 0.05\lambda \geq c \ln n$, for a sufficiently large constant c, optimizing DLB. Consider an iteration t and assume that block $i = \frac{n}{2} - 1 - o(n)$ is critical. Assume that the frequencies in blocks i and $i+1$ are all at $2/5$. Then the offspring population in iteration t has roughly $\big((2/5)^2/e\big)\lambda \approx 0.058\lambda > \mu$ individuals with at least $2i$ leading 1s in expectation. By Theorem 1, this also holds with high probability. Thus, the frequencies in block i are set to $1 - \frac{1}{n}$ with high probability.

The expected number of individuals with at least $2i+2$ leading 1s is roughly $\big((2/5)^4/e\big)\lambda \approx 0.0095\lambda$, and the expected number of individuals with $2i$ leading 1s followed by a 00 is roughly $\big((2/5)^2 \cdot (3/5)^2/e\big)\lambda \approx 0.02\lambda$. In total, we expect approximately $0.0295\lambda < \mu$ individuals with $2i$ leading 1s followed by either a 11 or a 00. Again, by Theorem 1, these numbers occur with high probability. Note that this implies that block $i+1$ is selection-relevant with high probability.

Consider block $i+1$. For selection, we choose all 0.0295λ individuals with $2i$ leading 1s followed by either a 11 or a 00 (which are sampled with high probability). For the remaining $\mu - 0.0295\lambda = 0.0205\lambda$ selected individuals with $2i$ leading 1s, we expect half of them, that is, 0.01025λ individuals to have a 10. Thus, with high probability, the frequency $p_{2i+1}^{(t+1)}$ is set to roughly $(0.0095 + 0.01025)\lambda/\mu = 0.395$, which is less than $0.4 = p_{2i+1}^{(t)}$. Thus, this frequency decreased.

The next lemma shows that such frequencies do not drop too low, however.

Lemma 8. *Let $\varepsilon, \delta \in (0,1)$ be constants, and let c be a sufficiently large constant. Consider the UMDA with $\lambda \geq \mu \geq c \ln n$ optimizing* DECEIVINGLEADINGBLOCKS. *Further, consider an iteration t such that block $i \in [2..\frac{n}{2}]$ is selection-relevant, and assume that its frequencies $p_{2i-1}^{(t)}$ and $p_{2i}^{(t)}$ are at least $(1-\varepsilon)/2$ when sampling the population. Then the frequencies $p_{2i-1}^{(t+1)}$ and $p_{2i}^{(t+1)}$ are at least $(1-\delta)(1-\varepsilon)^2/4$ with a probability of at least $1 - 4n^{-2}$.*

Proof. Let k denote the number of individuals with a prefix of at least $2i-2$ leading 1s. Since block i is selection-relevant, it follows that $k \geq \mu$. We consider a random variable X that follows a binomial law with k trials and with a success probability of $p_{2i-1}^{(t)} p_{2i}^{(t)} =: \widetilde{p} \geq (1-\varepsilon)^2/4$. We now bound the probability that at least $(1-\delta)\widetilde{p}\mu =: m$ have $2i$ leading 1s, that is, we bound $\Pr[X \geq m \mid X < \mu]$, where the condition follows from the definition of block i being selection-relevant.

Elementary calculations show that

$$\begin{aligned}\Pr[X \geq m \mid X < \mu] &= 1 - \Pr[X < m \mid X < \mu] = 1 - \frac{\Pr[X < m, X < \mu]}{\Pr[X < \mu]} \\ &= 1 - \frac{\Pr[X < m]}{\Pr[X < \mu]}. \qquad (1)\end{aligned}$$

To show a lower bound for (1), consider separately the two cases that $\mathrm{E}[X] < \mu$ and $\mathrm{E}[X] \geq \mu$.

Case 1: $\mathrm{E}[X] < \mu$**.** We first bound the numerator of the subtrahend in (1). Since $m/(1-\delta) = \widetilde{p}\mu \leq \widetilde{p}k = \mathrm{E}[X]$, we have $\Pr[X < m] \leq \Pr\left[X < (1-\delta)\mathrm{E}[X]\right]$. By Theorem 1, by $\mathrm{E}[X] \geq \widetilde{p}\mu$, and by our assumption that $\mu \geq c \ln n$, choosing c sufficiently large, we have

$$\Pr\left[X < (1-\delta)\mathrm{E}[X]\right] \leq \exp\left(-\frac{\delta^2 \mathrm{E}[X]}{2}\right) \leq \exp\left(-\frac{\delta^2 \widetilde{p}\mu}{2}\right) \leq n^{-2}.$$

For bounding the denominator, we note that $\widetilde{p} \leq 1 - \frac{1}{n}$ and use the fact that a binomially distributed random variable with a success probability of at most $1-\frac{1}{n}$ is below its expectation with a probability of at least $\frac{1}{4}$ [7, Lemma 10.20 (b)]. This yields

$$\Pr[X < \mu] \geq \Pr\left[X < \mathrm{E}[X]\right] \geq \frac{1}{4}.$$

Combining these bounds, we obtain $\Pr[X \geq m | X < \mu] \geq 1 - 4n^{-2}$ for this case.

Case 2: $\mathrm{E}[X] \geq \mu > m$**.** We bound the subtrahend from (1) from above. By basic estimations and by Corollary 3, we see that

$$\frac{\Pr[X < m]}{\Pr[X < \mu]} \leq \frac{\Pr[X \leq m-1]}{\Pr[X = \mu - 1]} \leq \frac{(k-m+1)\widetilde{p}}{\mathrm{E}[X] - m + 1} \cdot \frac{\Pr[X = m-1]}{\Pr[X = \mu - 1]}. \tag{2}$$

We bound the first factor of (2) as follows:

$$\begin{aligned}\frac{(k-m+1)\widetilde{p}}{\mathrm{E}[X]-m+1} &\leq \frac{\mathrm{E}[X]}{\mathrm{E}[X]-m} = 1 + \frac{m}{\mathrm{E}[X]-m} \leq 1 + \frac{m}{\mu - m} \leq 1 + \frac{m}{\frac{m}{\widetilde{p}} - m} \\ &= 1 + \frac{\widetilde{p}}{1-\widetilde{p}} \leq 1 + n - 1 = n\ ,\end{aligned}$$

where the last inequality uses that $\widetilde{p} \leq (1-\frac{1}{n})^2 \leq 1 - \frac{1}{n}$.

For the second factor of (2), we compute

$$\begin{aligned}\frac{\Pr[X = m-1]}{\Pr[X = \mu - 1]} &= \frac{\binom{k}{m-1}\widetilde{p}^{m-1}(1-\widetilde{p})^{k-m+1}}{\binom{k}{\mu-1}\widetilde{p}^{\mu-1}(1-\widetilde{p})^{k-\mu+1}} \\ &= \frac{(\mu-1)!(k-\mu+1)!}{(m-1)!(k-m+1)!} \cdot \left(\frac{1-\widetilde{p}}{\widetilde{p}}\right)^{\mu-m}. \end{aligned} \tag{3}$$

Since $\widetilde{p} \geq (1-\varepsilon)^2/4$, we see that $(1-\widetilde{p})/\widetilde{p} \leq 4/(1-\varepsilon)^2$.

For the first factor of (3), let $p^* := (1-\delta)\widetilde{p}$, thus $\mu p^* = m$. Noting that, for all $a, b \in \mathbb{R}$ with $a < b$, the function $j \mapsto (a+j)(b-j)$ is maximal for $j = (b-a)/2$, we first bound

$$\begin{aligned}\frac{(\mu-1)!}{(m-1)!} &= \prod_{j=0}^{\mu-m-1} (\mu - 1 - j) \leq \prod_{j=0}^{\lfloor(\mu-m-1)/2\rfloor} \big((\mu-1-j)(m+j)\big) \\ &\leq \left(\frac{\mu+m}{2}\right)^{\mu-m} \leq \left(\frac{\mu}{2}(1+p^*)\right)^{\mu-m}.\end{aligned}$$

Substituting this into the first factor of (3), we bound

$$\frac{(\mu-1)!(k-\mu+1)!}{(m-1)!(k-m+1)!} \leq \Big(\frac{\mu}{2}(1+p^*)\Big)^{\mu-m} \cdot \frac{(k-\mu+1)!}{(k-m+1)!}$$
$$= \frac{\big(\frac{\mu}{2}(1+p^*)\big)^{\mu-m}}{\prod_{j=0}^{\mu-m-1}(k-m+1-j)} = \prod_{j=0}^{\mu-m-1} \frac{\mu(1+p^*)}{2(k-m+1-j)}.$$

By noting that $k\widetilde{p} = \mathrm{E}[X] \geq \mu$, we bound the above estimate further:

$$\prod_{j=0}^{\mu-m-1} \frac{\mu(1+p^*)}{2(k-m+1-j)} \leq \prod_{j=0}^{\mu-m-1} \frac{\mu(1+p^*)}{2(\mu/\widetilde{p}-m+1-j)}$$
$$\leq \left(\frac{\mu(1+p^*)}{2\mu(1/\widetilde{p}-1)+2}\right)^{\mu-m} \leq \left(\frac{\widetilde{p}(1+p^*)}{2(1-\widetilde{p})}\right)^{\mu-m}.$$

Substituting both bounds into (3) and recalling that $m = \mu p^*$, we obtain

$$\frac{\Pr[X=m-1]}{\Pr[X=\mu-1]} \leq \left(\frac{1+p^*}{2}\right)^{\mu(1-p^*)} = \exp\left(-\mu(1-p^*)\ln\left(\frac{2}{1+p^*}\right)\right).$$

Finally, substituting this back into our bound of (2), using our assumption that $\mu \geq c\ln n$ and noting that p^* is constant, choosing c sufficiently large, we obtain

$$\frac{\Pr[X<m]}{\Pr[X<\mu]} \leq n\exp\left(-\mu(1-p^*)\ln\left(\frac{2}{1+p^*}\right)\right) \leq n^{-2}.$$

Concluding the proof. In both cases, we see that the number of 11s in block i is at least $m = (1-\delta)\widetilde{p}\mu \geq \big((1-\delta)(1-\varepsilon)^2/4\big)\mu$ with a probability of at least $1-4n^{-2}$. Since each 11 contributes to the new values of p_{2i-1} and p_{2i}, after the update, both frequencies are at least $(1-\delta)(1-\varepsilon)^2/4$, as we claimed. □

Optimizing the critical block. Our next lemma considers the critical block $i \in [\frac{n}{2}]$ of an iteration t. It shows that, with high probability, for all $j \in [2i]$, we have that $p_j^{(t+1)} = 1-\frac{1}{n}$. Informally, this means that (i) all frequencies left of the critical block remain at $1-\frac{1}{n}$, and (ii) the frequencies of the critical block are increased to $1-\frac{1}{n}$.

Lemma 9. *Let $\delta, \varepsilon, \zeta \in (0,1)$ be constants and let $q = (1-\delta)^2(1-\varepsilon)^4/16$. Consider the UMDA optimizing* DeceivingLeadingBlocks *with $\lambda \geq (4/\zeta^2)\ln n$ and $\mu/\lambda \leq (1-\zeta)q/e$, and consider an iteration t such that block $i \in [\frac{n}{2}]$ is critical and that $p_{2i-1}^{(t)}$ and $p_{2i}^{(t)}$ are at least $\sqrt{q}$. Then, with a probability of at least $1-n^{-2}$, at least μ offspring are generated with at least $2i$ leading 1s. In other words, the selection-relevant block of iteration t is at a position in $[i+1..\frac{n}{2}]$.*

Proof. Let X denote the number of individuals that have at least $2i$ leading 1s. Since block i is critical, each frequency at a position $j \in [2i-2]$ is at $1-\frac{1}{n}$. Thus, the probability that all of these frequencies sample a 1 for a single individual is $(1-\frac{1}{n})^{2i-2} \geq (1-\frac{1}{n})^{n-1} \geq 1/e$. Further, since the frequencies $p_{2i-1}^{(t)}$ and $p_{2i}^{(t)}$ are at least $\sqrt{q}$, the probability to sample a 11 at these positions is at least q. Hence, we have $\mathrm{E}[X] \geq q\lambda/e$.

We now apply Theorem 1 to show that it is unlikely that fewer than μ individuals from the current iteration have fewer than $2i$ leading 1s. Using our bounds on μ and λ, we compute

$$\Pr[X < \mu] \leq \Pr\left[X \leq (1-\zeta)\frac{q}{e}\lambda\right] \leq \Pr\left[X \leq (1-\zeta)\mathrm{E}[X]\right] \leq e^{-\frac{\zeta^2\lambda}{2}} \leq n^{-2}.$$

Thus, with a probability of at least $1-n^{-2}$, at least μ individuals have at least $2i$ leading 1s. This concludes the proof.

The run time of the UMDA on DLB. We now prove our main result.

Proof (of Theorem 5). We prove that the UMDA samples the optimum after $\frac{n}{2} + 2e\ln n$ iterations with a probability of at least $1-9n^{-1}$. Since it samples λ individuals each iteration, the theorem follows.

Due to Lemma 6 and $\mu \geq c_\mu n \log n$, for an $\varepsilon \in (0,1)$, within the first n iterations, with a probability of at least $1-2n^{-1}$, no frequency drops below $(1-\varepsilon)/2$ while its block has not been selection-relevant yet.

By Lemma 8, for another constant $\delta \in (0,1)$, with a probability of at least $1-4n^{-2}$, once a block becomes selection-relevant, its frequencies do not drop below $(1-\delta)(1-\varepsilon)^2/4$ for the next iteration. By a union bound, this does not fail for n consecutive times with a probability of at least $1-4n^{-1}$. Note that a selection-relevant block becomes critical in the next iteration.

Consider a critical block $i \in [\frac{n}{2}]$. By Lemma 9, choosing c_λ sufficiently large, with a probability of at least $1-n^{-2}$, all frequencies at positions in $[2i]$ are immediately set to $1-\frac{1}{n}$ in the next iteration, and the selection-relevant block has an index of at least $i+1$, thus, moving to the right. Applying a union bound for the first n iterations of the UMDA and noting that each frequency belongs to a selection-relevant block at most once shows that all frequencies are at $1-\frac{1}{n}$ after the first $\frac{n}{2}$ iterations, since each block contains two frequencies, and stay there for at least $\frac{n}{2}$ additional iterations with a probability of at least $1-2n^{-1}$.

Consequently, after the first $\frac{n}{2}$ iterations, the optimum is sampled in each iteration with a probability of $(1-\frac{1}{n})^n \geq 1/(2e)$. Thus, after $2e\ln n$ additional iterations, the optimum is sampled with a probability of at least $1-\big(1-1/(2e)\big)^{2e\ln n} \geq 1-n^{-1}$.

Overall, by applying a union bound over all failure probabilities above, the UMDA needs at most $n+\ln n$ iterations to sample the optimum with a probability of at least $1-9n^{-1}$. □

4 Experiments

In their paper, Lehre and Nguyen [18] analyze (among others) the $(1+1)$ EA and the (μ, λ) GA on DLB. For an optimal choice of parameters, they prove an expected run time of $O(n^3)$ for all considered algorithms.

Since these are only provide upper bounds, it is not clear how well the algorithms actually perform against the UMDA, which has a run time in the order of $n^2 \ln n$ (Theorem 5) for optimal parameters. Thus, we provide some empirical results in Figure 1 on how well these algorithms compare against each other.

We see that, for increasing n, the UMDA seems to perform best. Further, the run time behavior of the $(1+1)$ EA and the (μ, λ) GA is very similar. These findings indicate that there is a strict difference in run time between the UMDA and the other two algorithms.

Another interesting aspect of Figure 1 is the variance of the different algorithms. The $(1+1)$ EA and the (μ, λ) GA have a visible variance that does not seems to reduce. In contrast, the UMDA is strongly concentrated around its empirical mean. This behavior is supported by our analysis in Section 3, which proves a run time that holds with high probability.

Overall, the UMDA seems to be outperforming the competing approaches.

5 Conclusion

We conducted a rigorous run time analysis of the UMDA on the DecievingLeadingBlocks function. In particular, it shows that the algorithm with the right parameter choice finds the optimum within $O(n^2 \log n)$ fitness evaluations with high probability (Theorem 5). This result shows that the lower bound by Lehre and Nguyen [18], which is exponential in μ, is not due to the UMDA being ill-suited for coping with epistasis and deception, but rather due to an unfortunate choice of the algorithm's parameters. For several EAs, Lehre and Nguyen [18] showed a run time bound of $O(n^3)$ on DeceivingLeadingBlocks and we believe that this is tight, which is further supported by our experiments in Section 4. In this light, our result suggests that the UMDA can handle epistasis and deception even better than many evolutionary algorithms.

Our run time analysis holds for parameter regimes that prevent genetic drift. When comparing our run time with the one shown in [18], we obtain a strong suggestion for running EDAs in regimes of low genetic drift. In contrast to the work of Lengler, Sudholt, and Witt [19] that indicates moderate performance losses due to genetic drift, here we obtain the first fully rigorous proof of such a performance loss, and in addition one that is close to exponential in n (the $\exp(-\Omega(\mu))$ lower bound of [18] holds for μ up to $o(n)$).

On the technical side, our result indicates that the regime of low genetic drift admits relatively simple and natural analyses of run times of EDAs, in contrast, e.g., to the level-based methods previously used in comparable analyses, e.g., in [4,18].

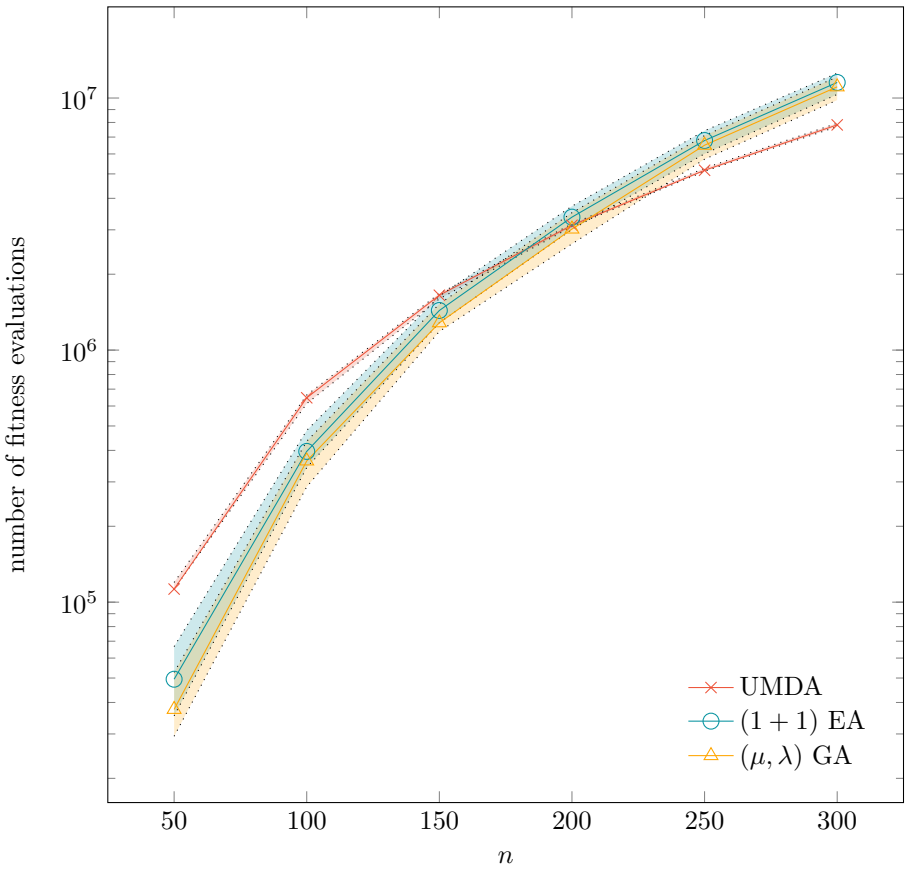

Figure 1. This figure shows the number of fitness evaluations (note the logarithmic scale) until the optimum is sampled for the first time. Depicted are three different algorithms and for various values of n (from 50 to 300 in steps of 50), optimizing DLB. For each value of n, 50 independent runs were started per algorithm. The results of these runs are depicted above. The lines depict the median of the 50 runs of an algorithm, and the shaded areas denote the center 50 %.
The UMDA uses $\mu = 3n \ln n$ and $\lambda = 12\mu$. The (μ, λ) GA uses $\mu = \ln n$, $\lambda = 9\mu$, uniform crossover, and has a crossover probability of $1/2$.

We conjecture that our result can be generalized to a version of the DeceivingLeadingBlocks function with a block size of $k \leq n$.

Acknowledgments

This work was supported by COST Action CA15140 and by a public grant as part of the Investissements d'avenir project, reference ANR-11-LABX-0056-LMH, LabEx LMH, in a joint call with Gaspard Monge Program for optimization, operations research and their interactions with data sciences.

References

1. Chen, T., Lehre, P.K., Tang, K., Yao, X.: When is an estimation of distribution algorithm better than an evolutionary algorithm? In: Proc. of CEC '09. pp. 1470–1477 (2009). https://doi.org/10.1109/CEC.2009.4983116
2. Dang, D., Friedrich, T., Kötzing, T., Krejca, M.S., Lehre, P.K., Oliveto, P.S., Sudholt, D., Sutton, A.M.: Escaping local optima with diversity mechanisms and crossover. In: Proc. of GECCO '16. pp. 645–652 (2016). https://doi.org/10.1145/2908812.2908956
3. Dang, D., Friedrich, T., Kötzing, T., Krejca, M.S., Lehre, P.K., Oliveto, P.S., Sudholt, D., Sutton, A.M.: Escaping local optima using crossover with emergent diversity. IEEE Transactions on Evolutionary Computation **22**(3), 484–497 (2018). https://doi.org/10.1109/TEVC.2017.2724201
4. Dang, D., Lehre, P.K.: Simplified runtime analysis of estimation of distribution algorithms. In: Proc. of GECCO '15. pp. 513–518 (2015). https://doi.org/10.1145/2739480.2754814
5. Doerr, B.: Analyzing randomized search heuristics via stochastic domination. Theoretical Computer Science **773**, 115–137 (2019). https://doi.org/10.1016/j.tcs.2018.09.024
6. Doerr, B.: A tight runtime analysis for the cGA on jump functions: EDAs can cross fitness valleys at no extra cost. In: Proc. of GECCO '19. pp. 1488–1496 (2019). https://doi.org/10.1145/3321707.3321747
7. Doerr, B.: Probabilistic tools for the analysis of randomized optimization heuristics. In: Theory of Evolutionary Computation: Recent Developments in Discrete Optimization, pp. 1–87. Springer (2020). https://doi.org/10.1007/978-3-030-29414-4_1, also available at `https://arxiv.org/abs/1801.06733`
8. Doerr, B., Künnemann, M.: Optimizing linear functions with the $(1+\lambda)$ evolutionary algorithm - Different asymptotic runtimes for different instances. Theoretical Computer Science **561**, 3–23 (2015). https://doi.org/10.1016/j.tcs.2014.03.015
9. Doerr, B., Krejca, M.S.: Significance-based estimation-of-distribution algorithms. In: Proc. of GECCO '18. pp. 1483–1490 (2018). https://doi.org/10.1145/3205455.3205553
10. Doerr, B., Zheng, W.: Sharp bounds for genetic drift in EDAs. CoRR **abs/1910.14389** (2019), `https://arxiv.org/abs/1910.14389`
11. Droste, S.: A rigorous analysis of the compact genetic algorithm for linear functions. Natural Computing **5**(3), 257–283 (2006). https://doi.org/10.1007/s11047-006-9001-0
12. Droste, S., Jansen, T., Wegener, I.: On the analysis of the (1+1) evolutionary algorithm. Theoretical Computer Science **276**(1-2), 51–81 (2002). https://doi.org/10.1016/S0304-3975(01)00182-7

13. Hasenöhrl, V., Sutton, A.M.: On the runtime dynamics of the compact genetic algorithm on jump functions. In: Proc. of GECCO '18. pp. 967–974 (2018). https://doi.org/10.1145/3205455.3205608
14. Hoeffding, W.: Probability inequalities for sums of bounded random variables. Journal of the American Statistical Association **58**(301), 13–30 (1963). https://doi.org/10.2307/2282952
15. Krejca, M.S., Witt, C.: Lower bounds on the run time of the univariate marginal distribution algorithm on OneMax. In: Proc. of FOGA '17. pp. 65–79 (2017). https://doi.org/10.1145/3040718.3040724
16. Krejca, M.S., Witt, C.: Theory of estimation-of-distribution algorithms. In: Theory of Evolutionary Computation: Recent Developments in Discrete Optimization, pp. 405–442. Springer (2020). https://doi.org/10.1007/978-3-030-29414-4_1, also available at `http://arxiv.org/abs/1806.05392`
17. Lehre, P.K., Nguyen, P.T.H.: Improved runtime bounds for the univariate marginal distribution algorithm via anti-concentration. In: Proc. of GECCO '17. pp. 1383–1390 (2017). https://doi.org/10.1145/3071178.3071317
18. Lehre, P.K., Nguyen, P.T.H.: On the limitations of the univariate marginal distribution algorithm to deception and where bivariate EDAs might help. In: Proc. of FOGA '19. pp. 154–168 (2019). https://doi.org/10.1145/3299904.3340316
19. Lengler, J., Sudholt, D., Witt, C.: Medium step sizes are harmful for the compact genetic algorithm. In: Proc. of GECCO '18. pp. 1499–1506 (2018). https://doi.org/10.1145/3205455.3205576
20. Mühlenbein, H., Paaß, G.: From recombination of genes to the estimation of distributions I. Binary parameters. In: Proc. of PPSN '96. pp. 178–187 (1996). https://doi.org/10.1007/3-540-61723-X_982
21. Pelikan, M., Hauschild, M., Lobo, F.G.: Estimation of distribution algorithms. In: Springer Handbook of Computational Intelligence, pp. 899–928. Springer (2015). https://doi.org/10.1007/978-3-662-43505-2_45
22. Sudholt, D., Witt, C.: On the choice of the update strength in estimation-of-distribution algorithms and ant colony optimization. Algorithmica **81**(4), 1450–1489 (2019). https://doi.org/10.1007/s00453-018-0480-z
23. Witt, C.: Domino convergence: why one should hill-climb on linear functions. In: Proc. of GECCO '18. pp. 1539–1546 (2018). https://doi.org/10.1145/3205455.3205581
24. Witt, C.: Upper bounds on the running time of the univariate marginal distribution algorithm on OneMax. Algorithmica **81**(2), 632–667 (2019). https://doi.org/10.1007/s00453-018-0463-0
25. Zheng, W., Yang, G., Doerr, B.: Working principles of binary differential evolution. In: Proc. of GECCO '18. pp. 1103–1110 (2018). https://doi.org/10.1145/3205455.3205623