EDAs cannot be Balanced and Stable

Tobias Friedrich
Hasso Plattner Institute
Potsdam, Germany

Timo Kötzing
Hasso Plattner Institute
Potsdam, Germany

Martin S. Krejca
Hasso Plattner Institute
Potsdam, Germany

ABSTRACT

Estimation of Distribution Algorithms (EDAs) work by iteratively updating a distribution over the search space with the help of samples from each iteration. Up to now, theoretical analyses of EDAs are scarce and present run time results for specific EDAs. We propose a *new framework* for EDAs that captures the idea of several known optimizers, including PBIL, UMDA, λ-MMAS$_{\text{IB}}$, cGA, and $(1, \lambda)$-EA.

Our focus is on analyzing two core features of EDAs: a *balanced* EDA is sensitive to signals in the fitness; a *stable* EDA remains uncommitted under a biasless fitness function. We prove that no EDA can be both balanced and stable.

The LeadingOnes function is a prime example where, at the beginning of the optimization, the fitness function shows no bias for many bits. Since many well-known EDAs are balanced and thus not stable, they are not well-suited to optimize LeadingOnes. We give a stable EDA which optimizes LeadingOnes within a time of $O(n \log n)$.

Categories and Subject Descriptors

F.2 [**Theory of Computation**]: Analysis of Algorithms and Problem Complexity

Keywords

Estimation of distribution algorithm; LeadingOnes; theory; run time analysis

1. INTRODUCTION

Estimation of Distribution Algorithms (EDAs, [15]) are search meta-heuristics that maintain a probability distribution of the solution space and iteratively update it according to samples from this distribution. This is in contrast to Evolutionary Algorithms (EAs), which employ an explicit set of potential solutions, called *population*, and update this set with variation operators such as mutation, recombination, and selection.

GECCO '16 July 20-24, 2016, Denver, CO, USA

ACM ISBN 978-1-4503-4206-3/16/07.

DOI: http://dx.doi.org/10.1145/2908812.2908895

Hauschild and Pelikan [11] give a nice survey of EDAs where they point out many successful applications of these algorithms to a wide range of problems, frequently yielding better results than any other competing algorithms. They also state advantages of EDAs that give an explanation to *why* they perform so well; one of these being *reduced memory requirements* of, what they call, *incremental* EDAs. Such EDAs only sample a small set of solutions each iteration, which is discarded afterward, whereas classical EAs always have to store their entire population. This can lead to drastic differences in the memory needed and has been shown by, e.g., Sastry, Goldberg, and Llorà [21], where they compare a simple genetic algorithm to the Compact Genetic Algorithm (cGA, [10]), which only samples 2 solutions each iteration.

In this paper, we consider the Boolean domain $\{0,1\}^n$ as search space. An *arbitrary* distribution over $\{0,1\}^n$ requires storing of 2^n different values, which is infeasible. To counteract this combinatorial explosion, many discrete EDAs – such as the Population-based Incremental Learning algorithm (PBIL, [2]), the Univariate Marginal Distribution Algorithm (UMDA, [16]), and the cGA – assume independence of the different bit positions, thus reducing the memory space needed for one distribution down to n values. Hauschild and Pelikan [11] call these kind of EDAs *univariate*. Mathematically speaking, such EDAs maintain a Poisson binomial distribution, i.e., a frequency vector $\boldsymbol{p} \in [0,1]^n$, which holds the probabilities to sample a 1 at each bit position (instead of an arbitrary distribution over $\{0,1\}^n$).

Because of their nice properties, many of the few results from run time analysis on EDAs in the Boolean domain have been made using univariate incremental EDAs as their model [3, 5, 7, 9, 13, 17]. However, most of these results focus only on one specific algorithm and not on basic properties of EDAs that are sufficient or necessary for optimization. Some results do not even mention that the algorithm analyzed is, in fact, an EDA. We therefore propose a general framework for univariate incremental EDAs, called the n-Bernoulli-λ-EDA, which subsumes many EDAs that have been analyzed.

A similar approach has already been done by Ollivier, Arnold, Auger, and Hansen by proposing the Information-Geometric Optimization method (IGO, [18]). IGO is a very general method defined for arbitrary search spaces, maximizing invariance properties of said space. Applying the IGO method to the Boolean domain results in a more general PBIL with weights, which is capable of subsuming all of the aforementioned EDAs. It does however not encapsulate *all* univariate incremental EDAs.

Another framework for evolutionary processes in general was introduced by Paixão et al. [19]. This very general framework subsumes all univariate incremental EDAs; however, due to its generality, it does not specifically focus on EDAs and is not well-suited for their analysis.

Other approaches have been examined by Shapiro [23], who analyzed EDAs that use a maximum-likelihood update, and by Corus, Dang, Eremeev, and Lehre [4], who proposed a class of EDAs that update their distributions only with information from the current samples, not with information from the current distribution. Again, these approaches do not capture all univariate incremental EDAs.

Our framework – the n-Bernoulli-λ-EDA – has the benefit of being very general with respect to EDAs of interest over the Boolean domain while being easy to analyze and even showing connections between existing EDAs. We hope that this framework leads to a more general analysis of EDAs, focusing on the properties needed to succeed in optimization instead of analyzing specific algorithms.

In Section 2 we introduce the framework and in Section 3 we show how the well-known EDAs PBIL, UMDA, cGA, λ-MMAS$_{\text{IB}}$ [24], and even $(1,\lambda)$-EA [22] fit very well into the framework.

Furthermore, we classify n-Bernoulli-λ-EDAs depending on whether they are *locally updating*, meaning that they decide, for each position and depending on the samples and their fitness, whether to increase or decrease the corresponding frequency (or to stay at the same frequency). The update is then performed according to a fixed update rule; in fact, this update rule is at the core of the EDA and is easily depicted as a graph. Figure 1 gives an overview of the graphs for all locally updating n-Bernoulli-λ-EDAs we discuss.

Section 4 introduces the two important terms *balanced* and *stable*. Both terms come into play when looking at an n-Bernoulli-λ-EDA that does not get any information for a certain bit position i. In other words, the fitness of a bit string does not depend on whether it has a 1 or a 0 in position i. Thus, there is no preference for whether to set bit i to 1 or to 0, so we might want any EDA to stay undecided (with a frequency of $1/2$). We call an n-Bernoulli-λ-EDA *stable* if the frequency $\boldsymbol{p}_i$ is concentrated around $1/2$ (in the limit). Furthermore, in such a scenario we do not want the frequency $\boldsymbol{p}_i$ to change (in expectation); we call this property of an n-Bernoulli-λ-EDA *balanced*. With these definitions, we want an n-Bernoulli-λ-EDA to be balanced *and* stable at the same time. However, as the main result of this paper, we prove that this is impossible for a vast class of n-Bernoulli-λ-EDAs by showing that frequencies of balanced n-Bernoulli-λ-EDAs that do not get any information tend to drift to their borders 0 or 1 (see Theorem 10).

All our example EDAs are balanced and not stable. In Section 5 we show how to easily adjust the cGA to a stable variant: the scGA (which is now not balanced). A test function frequently considered in the literature is LeadingOnes [5, 8]. It returns the number of leading 1s in a bit string, starting from the left. This problem is equivalent to finding a hidden permutation, and many traditional search-heuristics need time in $\Theta\big(n^2\big)$, as discussed by Afshani et al. [1]. Because of this problem structure, for a long time there is no relevant information regarding bits at the very end of the bit string and a stable n-Bernoulli-λ-EDA would be preferable. We show that the scGA optimizes LeadingOnes in $\mathrm{O}(n \log n)$, which is close to the best possible run time of $\Theta(n \log \log n)$ [1].

2. PRELIMINARIES

We consider the optimization of pseudo-Boolean functions, i.e., functions $f\colon \{0,1\}^n \to \mathbb{R}$, which we call *fitness functions*.

Throughout the whole paper let n denote the dimension of the solution space $\{0,1\}^n$. For any bit string $\boldsymbol{x} \in \{0,1\}^n$, we call $f(\boldsymbol{x})$ the *fitness of* $\boldsymbol{x}$, and we denote the i-th bit of $\boldsymbol{x}$ by $\boldsymbol{x}_i$ $\big(i \in \{1,\dots,n\}\big)$.

2.1 Our EDA Framework

We present the n-Bernoulli-λ-EDA, an EDA inspired by evolutionary algorithms, that keeps a Poisson binomial distribution, i.e., the n-fold product of a Bernoulli distribution, and updates this distribution by sampling $\lambda \in \mathbb{N}^+$ offspring.

At any point in time t, the state of the algorithm is completely determined by its *frequency vector* $\boldsymbol{p}^{(t)} \in [0,1]^n$, whose components we call *frequencies*. That means that the probability to sample any individual $\boldsymbol{x} \in \{0,1\}^n$ is as follows (let i range from 1 to n if not stated otherwise):

$$\forall i\colon \Pr(\boldsymbol{x}_i = 1) = \boldsymbol{p}_i \wedge \Pr(\boldsymbol{x}_i = 0) = 1 - \boldsymbol{p}_i \ .$$

The initial frequency vector is given by $\boldsymbol{p}^{(0)} = (0.5)_{i=1}^n$. In each iteration, the algorithm samples λ offspring and updates each frequency according to its *update scheme* until an optimal solution is found.

The update scheme of an n-Bernoulli-λ-EDA is a function $\varphi\colon [0,1]^n \times \big(\{0,1\}^n \times \mathbb{R}\big)^\lambda \to [0,1]^n$ that takes the current frequency vector $\boldsymbol{p}^{(t)}$, an offspring population D of λ individuals sampled according to $\boldsymbol{p}^{(t)}$, and their respective fitness and yields the frequencies of $\boldsymbol{p}^{(t+1)}$ for the following iteration. Thus, determining the update scheme φ determines the n-Bernoulli-λ-EDA as seen in Algorithm 1.

We call an update scheme *local* if there are two functions,

- $\text{move}\colon \big(\{0,1\} \times \mathbb{R}\big)^\lambda \to \{\text{up}, \text{stay}, \text{down}\}$ and
- $\text{set}\colon [0,1] \to [0,1]$,

such that, for all i and for $|D| = \lambda$, abbreviating $v_i = \text{move}\big(\big(\boldsymbol{x}_i, f(\boldsymbol{x})\big)_{\boldsymbol{x} \in D}\big)$,

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

This means that the update of each frequency $\boldsymbol{p}_i$ is independent from the others, making only use of the value of $\boldsymbol{p}_i$ alone for an update. Thus, each update can only use local information. Note that increase and decrease are centrally symmetric around $(1/2, 1/2)$, i.e., an increase of $\boldsymbol{p}_i$ is the same as a decrease of $1 - \boldsymbol{p}_i$. We also call such an n-Bernoulli-λ-EDA *locally updating*.

We say that an n-Bernoulli-λ-EDA is ρ-*bounded* if, for all i, $|\boldsymbol{p}_i - \varphi(\boldsymbol{p}, \cdot)_i| \leq \rho$.

If, for all $t \in \mathbb{N}$, $\boldsymbol{p}^{(t)} \in [b, 1-b]^n$, with $b \in [0, 1/2)$, we say that the n-Bernoulli-λ-EDA has a *margin* of b, and we call b and $1 - b$ the (lower and upper) borders. If $b = 0$, we say that the margin is *trivial*. A nontrivial margin prevents the algorithm from getting trapped in a bit position.

The definition of the n-Bernoulli-λ-EDA does not explicitly make use of mutation since the framework can already handle mutation implicitly. A simple way to do so is the following, assuming independent mutation per bit. Let $\text{mutate}\colon [0,1] \to [0,1]$ denote a mutation operator, and let p_m denote the probability that a mutation takes place. Then $\text{mutate}^*\colon \boldsymbol{p}_i \mapsto p_\text{m} \cdot \text{mutate}(\boldsymbol{p}_i) + (1 - p_\text{m})\,\boldsymbol{p}_i$ describes the expectation of $\boldsymbol{p}_i$ after one step that may involve mutation. Composing mutate^* with an update scheme (componentwise) results in an n-Bernoulli-λ-EDA that makes use of mutation.

In general, by making use of random variables in the n-Bernoulli-λ-EDA's update scheme, even more sophisticated mutation operators are possible.

Algorithm 1: n-Bernoulli-λ-EDA with a given update scheme φ

```
t ← 0;
foreach i ∈ {1, ..., n} do p_i^(t) ← 1/2 ;
repeat
    D ← ∅;
    for j ∈ {1, ..., λ} do
        x ← offspring sampled with respect to p^(t);
        D ← D ∪ {x};
    p^(t+1) ← φ(p^(t), (x, f(x))_{x∈D});
    t ← t + 1;
until optimum in D;
```

2.2 LeadingOnes and Tools

Our fitness function of interest is the well-known LeadingOnes function, which counts the number of consecutive 1s in a bit string, starting from the left. More formally, $\text{LeadingOnes}\colon \{0,1\}^n \to \{0,\dots,n\}$ and, for all $\boldsymbol{x} \in \{0,1\}^n$, $\text{LeadingOnes}(\boldsymbol{x}) = \sum_{i=1}^{n} \prod_{j=1}^{i} \boldsymbol{x}_j$. We want to maximize LeadingOnes, hence the unique global optimum is the all-ones string 1^n.

We are also making use of drift theory, i.e., we can give expected hitting times and concentration bounds on Markovian processes if there is a bias, called the *drift*, toward a certain direction.

Theorem 1 (Multiplicative Drift [6])**.** *Let $(X_t)_{t\in\mathbb{N}}$ be random variables over $\mathbb{R}_0^+$, each with finite expectation, and let $T = \min\{t\colon X_t < 1\}$. Suppose there exists an $\varepsilon > 0$ such that, for all t, $\mathrm{E}(X_t - X_{t+1} \,|\, X_t, t < T) \geq \varepsilon X_t$.*

Then $\mathrm{E}(T \,|\, X_0) \leq \frac{1+\ln X_0}{\varepsilon}$.

Theorem 2 (Negative Drift [14])**.** *Let $(X_t)_{t\in\mathbb{N}}$ be random variables over $\mathbb{R}$, each with finite expectation, let $b > 0$, and let $T = \min\{t\colon X_t \geq b \,|\, X_0 \leq 0\}$. Suppose there are ρ, $0 < \rho < b$ and $\varepsilon < 0$ such that, for all t, (1.) $\mathrm{E}(X_{t+1} - X_t \,|\, X_t, t < T) \leq \varepsilon$, and (2.) $|X_{t+1} - X_t| < \rho$.*

Then, for all $t \in \mathbb{N}$, $\Pr(T \leq t) \leq t \exp\left(-\frac{b|\varepsilon|}{16\rho^2}\right)$.

We say that an event E occurs *with high probability* if, for a constant $c > 0$, $\Pr(\overline{E}) = \mathrm{O}(n^{-c})$.

3. CLASSIFYING EXISTING EDAs

In this section we show how existing EDAs are subsumed by the n-Bernoulli-λ-EDA. To simplify the notation of all update schemes we present, we suppose that D is always ordered by fitness such that $\boldsymbol{x}^{(k)}$ is the k-th best individual (ties are broken such that each index is unique). This implies that $\boldsymbol{x}^{(1)}$ is a best individual.

Every n-Bernoulli-λ-EDA can be defined to have a nontrivial margin b by setting the respective frequency to the border value (whenever the border is crossed), just as introduced by Stützle and Hoos [24]. Since the choice of b is arbitrary, we only mention the margin for the following algorithms if they trivially follow from the update scheme. If the scheme allows for frequencies up to 0 or 1, we do not enforce a margin. We further assume that, if an update creates frequencies outside of the interval $[0,1]$, said frequency is set to either 0 or 1, whichever is closer.

PBIL. The Population-based Incremental Learning algorithm (PBIL) was introduced by Baluja [2] and has been, for example, theoretically analyzed by Hohfeld and Rudolph [12].

It is a ρ-bounded n-Bernoulli-λ-EDA with parameters ρ, the so-called *learning rate*, and μ, the so-called *population size*, with $\mu \leq \lambda$. The update scheme is, for all i,

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

This algorithm yields other well-known algorithms for some extreme cases of μ or ρ, as we will show next.

UMDA. The Univariate Marginal Distribution Algorithm (UMDA) was introduced by Mühlenbein and Paass [16]. Some theoretical analyses have been conducted by Chen et al. [3] and Dang and Lehre [5].

The UMDA is an n-Bernoulli-λ-EDA with parameter μ, the so-called *population size*, with $\mu \leq \lambda$. The update scheme is, for all i,

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

This is a special case of PBIL for $\rho = 1$. The update does not make any use of the frequencies anymore and sets them to the relative frequency of the μ best individuals.

λ-MMAS$_{\text{IB}}$/λ-AS$_{\text{IB}}$. The MAX-MIN Ant System algorithm (MMAS) was introduced by Stützle and Hoos [24]. We consider a version using λ ants with iteration-best update (IB), as introduced and analyzed by Neumann et al. [17]. It is a locally updating, ρ-bounded n-Bernoulli-λ-EDA with a nontrivial margin, where the parameter ρ is the so-called *evaporation factor*.

The *MAX-MIN* part of the name indicates a nontrivial margin but the update scheme does not; we refer to the algorithm with trivial margin as λ-AS$_{\text{IB}}$. Its update scheme is the same as that of PBIL for $\mu = 1$; in this case, the update scheme makes use of only one bit (one of a best individual) instead of arbitrarily many, which gives that λ-AS$_{\text{IB}}$ is a locally updating n-Bernoulli-λ-EDA whereas PBIL is not.

The update schemes of λ-MMAS$_{\text{IB}}$ and λ-AS$_{\text{IB}}$ and the two following algorithms can be seen in Table 1.

$(1,\lambda)$-EA. The $(1,\lambda)$-EA was introduced by Schwefel [22] and has been analyzed by Rowe and Sudholt [20].

It is a locally updating n-Bernoulli-λ-EDA with margin $1/n$, where λ is the *offspring population size*, just as in the definition of the n-Bernoulli-λ-EDA. However, the update

Algorithm	move	set
λ-AS$_{\text{IB}}$	$\text{move}\Big(\big(\boldsymbol{x}_i, f(\boldsymbol{x})\big)_{\boldsymbol{x}\in D}\Big) = \begin{cases}\text{up} & \text{if } \boldsymbol{x}_i^{(1)} = 1\,,\\ \text{down} & \text{if } \boldsymbol{x}_i^{(1)} = 0\,.\end{cases}$	$\text{set}(\boldsymbol{p}_i) = (1-\rho)\,\boldsymbol{p}_i + \rho$
$(1,\lambda)$-EA		$\text{set}(\boldsymbol{p}_i) = 1 - \frac{1}{n}$
cGA	$\text{move}\Big(\big(\boldsymbol{x}_i, f(\boldsymbol{x})\big)_{\boldsymbol{x}\in D}\Big) = \begin{cases}\text{up} & \text{if } \boldsymbol{x}_i^{(1)} > \boldsymbol{x}_i^{(2)},\\ \text{down} & \text{if } \boldsymbol{x}_i^{(1)} < \boldsymbol{x}_i^{(2)},\\ \text{stay} & \text{if } \boldsymbol{x}_i^{(1)} = \boldsymbol{x}_i^{(2)}.\end{cases}$	$\text{set}(\boldsymbol{p}_i) = \boldsymbol{p}_i + \rho$

Table 1: The three local update schemes in comparison. λ-AS$_{\text{IB}}$ and $(1,\lambda)$-EA only differ in the definition of their set function.

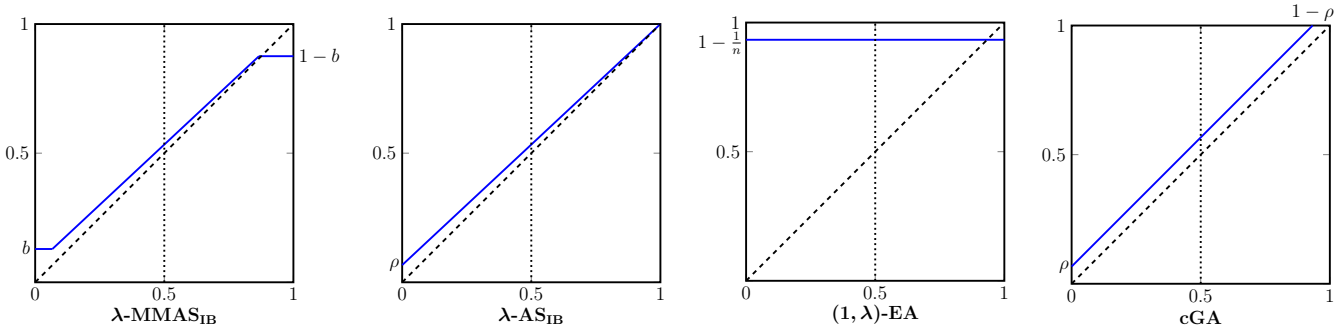

Figure 1: The set functions for all of the mentioned locally updating EDAs in comparison. The dashed line always represents the identity, and horizontal lines intersecting with the plot's borders represent margins.

does not make use of the frequencies at all. That is why the $(1,\lambda)$-EA always has a nontrivial margin.

Note that the update schemes of the $(1,\lambda)$-EA and λ-AS$_{\text{IB}}$, seen in Table 1, only differ in how their set function is defined. This similarity has also been noted and even analyzed by Neumann et al. [17]. In the n-Bernoulli-λ-EDA framework, this close connection can easily be seen.

cGA. The Compact Genetic Algorithm (cGA) was introduced by Harik et al. [10] and has been analyzed, e.g., in [7, 9].

It is a locally updating, ρ-bounded n-Bernoulli-2-EDA with parameter ρ, where $1/\rho$ is the so-called *population size*, normally denoted as $K = 1/\rho$. We, however, denote its parameter with ρ to make this notation more consistent with the other algorithms' notations.

The cGA's update scheme can be seen in Table 1. It is important that the cGA is able to make no update at all if, for an index i, the two sampled bits are the same. In the other two cases it shifts $\boldsymbol{p}_i$ toward the direction of the fitter individual's bit value.

The set functions. An overview of the different set functions can be seen in Figure 1. It shows, for example, how a margin changes the update by cutting off values that are too low or too high, as seen in the plots for λ-MMAS$_{\text{IB}}$ and λ-AS$_{\text{IB}}$. Moreover, one easily sees how the update behaves by looking at the distance to the identity function. This distance depicts the change made by a single update. Once the identity is intersected or a border is hit, the respective frequency is stuck.

A set function on its own does not fully determine an n-Bernoulli-λ-EDA because a move function is missing. However, the set function denotes the *behavior* of an update, whereas the move function denotes the *probability* of an update.

Extensions. Note that our definition of an n-Bernoulli-λ-EDA does not cover storing the best-so-far solution but can easily be modified to do so. For such an n-Bernoulli-$(1+\lambda)$-EDA, one only has to adjust the update scheme to take $\lambda+1$ pairs of bit strings and fitnesses instead of λ.

4. BALANCED VS. STABLE

EDAs succeed in optimizing a fitness function f by noticing a bias in their samples, introduced by f, and then updating their distribution accordingly. However, it is interesting to see what happens if there is no bias at a certain position.

Definition 3. *Given an n-Bernoulli-λ-EDA A and a fitness function f, we say that a position i of A is* f-independent *if, for all $\boldsymbol{x}, \boldsymbol{y} \in \{0,1\}^n$ such that $\boldsymbol{x}$ and $\boldsymbol{y}$ only differ in position i, $f(\boldsymbol{x}) = f(\boldsymbol{y})$.*

Having an f-independent position i, we define the following two types of behavior that $\boldsymbol{p}_i$ might express.

Definition 4. *An n-Bernoulli-λ-EDA A is* balanced *if, for all f-independent positions i of A and for all $t \in \mathbb{N}$, the frequency $\boldsymbol{p}_i^{(t)}$ does not change, in expectation, after one update, i.e., $\mathrm{E}\big(\boldsymbol{p}_i^{(t+1)} \,|\, \boldsymbol{p}_i^{(t)}\big) = \boldsymbol{p}_i^{(t)}$.*

Definition 5. *An n-Bernoulli-λ-EDA A is* stable *if, for all f-independent positions i of A, the limit distribution of frequency $\boldsymbol{p}_i^{(t)}$, as $t \to \infty$, exists and is symmetric around $1/2$, taking its maximum at $1/2$, and is strictly monotonically decreasing from $1/2$ toward the borders.*

An f-independent position is completely meaningless when optimizing f and could be ignored. This definition is rather strict and can be relaxed such that the term may also be applied when a position is f-independent with respect to all individuals $\boldsymbol{x}$ of a current offspring population D and not all $\boldsymbol{x} \in \{0,1\}^n$. This happens, for example, in the case of the LeadingOnes function: early in the optimization process, bits towards the end of the bit string will be LeadingOnes-independent for all practical purposes, but will become meaningful later in the optimization.

Intuitively, it would be sensible if the frequency of an f-independent position would not stray too far from its last meaningful value. We will, however, show that this is not the case for a large class of n-Bernoulli-λ-EDAs.

We start off by giving examples of some n-Bernoulli-λ-EDAs that are balanced and some that are not.

Theorem 6. *PBIL, UMDA, λ-AS$_{IB}$, and the cGA are balanced, the $(1,\lambda)$-EA is not.*

Proof. Let position i be f-independent, and let $\boldsymbol{p}_i'$ be the value of $\boldsymbol{p}_i$ after an update. Let $\boldsymbol{x}^{(k)}$ denote the k-th best individual in the offspring population D of any of the n-Bernoulli-λ-EDAs listed below, as defined in the beginning of Section 3.

Since position i is f-independent, $\boldsymbol{x}_i^{(k)}$ is 1 with probability $\boldsymbol{p}_i$ and 0 with probability $1-\boldsymbol{p}_i$.

PBIL:

$$\begin{aligned}\mathrm{E}\big(\boldsymbol{p}_i' \,\big|\, \boldsymbol{p}_i\big) &= (1-\rho)\boldsymbol{p}_i + \rho\frac{\sum_{k=1}^{\mu}\mathrm{E}\big(\boldsymbol{x}_i^{(k)}\big)}{\mu} \\ &= (1-\rho)\boldsymbol{p}_i + \rho\frac{\sum_{k=1}^{\mu}\boldsymbol{p}_i}{\mu} = (1-\rho)\boldsymbol{p}_i + \rho\boldsymbol{p}_i = \boldsymbol{p}_i\ .\end{aligned}$$

UMDA/λ-AS$_{\mathrm{IB}}$: Both algorithms are balanced since they are special cases of PBIL.

cGA:

$$\begin{aligned}\mathrm{E}\big(\boldsymbol{p}_i' \,\big|\, \boldsymbol{p}_i\big) &= (\boldsymbol{p}_i-\rho)\boldsymbol{p}_i(1-\boldsymbol{p}_i) + \boldsymbol{p}_i\big(1-2\boldsymbol{p}_i(1-\boldsymbol{p}_i)\big) + \\ &\qquad (\boldsymbol{p}_i+\rho)\boldsymbol{p}_i(1-\boldsymbol{p}_i) \\ &= 2\boldsymbol{p}_i^2(1-\boldsymbol{p}_i) + \boldsymbol{p}_i - 2\boldsymbol{p}_i^2(1-\boldsymbol{p}_i) = \boldsymbol{p}_i\ .\end{aligned}$$

$(1,\lambda)$-EA:

$$\begin{aligned}\mathrm{E}\big(\boldsymbol{p}_i' \,\big|\, \boldsymbol{p}_i\big) &= \frac{1}{n}(1-\boldsymbol{p}_i) + \left(1-\frac{1}{n}\right)\boldsymbol{p}_i \\ &= \frac{1}{n} + \left(1-\frac{2}{n}\right)\boldsymbol{p}_i \overset{\text{if } \boldsymbol{p}_i \neq \frac{1}{2}}{\neq} \boldsymbol{p}_i\ .\end{aligned}$$

□

Margins. The property of an n-Bernoulli-λ-EDA being balanced highly depends on the algorithm's margin. If its set function does not asymptotically go toward the identity as the frequency goes toward the upper border, there exist frequencies such that the expected value of said frequencies after an update is closer to 1/2 than to the upper border because the increase gets cut off by the margin. If such frequencies can be reached, the respective n-Bernoulli-λ-EDA is not balanced.

Recall Figure 1. The set function of λ-AS$_{\mathrm{IB}}$ goes toward the identity as the frequency goes toward 1 (the upper border). Even if the set function were scaled such that its domain were $[b, 1-b]$ for a margin $b \in (0, 1/2)$, in the limit, a frequency going toward $1-b$ would reach the identity. Thus, even scaled variants of λ-AS$_{\mathrm{IB}}$'s set function result in balanced n-Bernoulli-λ-EDAs. This does *not* hold for the λ-MMAS$_{\mathrm{IB}}$, where an update close to the upper border $1-b$ results in the expected new frequency being closer to 1/2 because the increase gets cut off. This happens for the cGA as well when looking at frequencies in $(1-\rho, 1)$. However, since no frequency can ever take these values, they are not considered.

All in all, if one conditions on the frequencies being far (depending on the specific set function) away from the borders, the corresponding n-Bernoulli-λ-EDAs can be viewed as balanced. For example, if we condition for the $(1,\lambda)$-EA on not having reached the borders yet, i.e., $\boldsymbol{p}_i = 1/2$, it turns out to be balanced as well, since $1/n + (1-2/n)/2 = 1/2$.

We now show that λ-AS$_{\mathrm{IB}}$, $(1,\lambda)$-EA, and cGA are *not* stable. Both λ-AS$_{\mathrm{IB}}$ and cGA are even horribly unstable in the sense that their variance goes *exponentially* fast to the maximum value of 1/4 (Corollaries 8 and 9)! That is, the standard deviation approaches 1/2, which is maximal, since all frequencies start at 1/2.

Showing that the $(1,\lambda)$-EA is unstable is trivial as, in the limit, the frequency of an f-independent position can only take the values $1/n$ and $1-1/n$. The proofs for λ-AS$_{\mathrm{IB}}$ and cGA follow the same pattern, which we formulate as the following lemma.

Lemma 7. *Given an f-independent position i of a balanced n-Bernoulli-λ-EDA A, assume that, for all $t \in \mathbb{N}$, $\mathrm{Var}\big(\boldsymbol{p}_i^{(t+1)} \mid \boldsymbol{p}_i^{(t)}\big) = -a\big(\boldsymbol{p}_i^{(t)}\big)^2 + b\boldsymbol{p}_i^{(t)} + c$ with $0 < a < 1$, and let $q = -\frac{1}{4}a + \frac{1}{2}b + c$.*

Then, for all $t \in \mathbb{N}$, $\mathrm{Var}\big(\boldsymbol{p}_i^{(t)}\big) = \frac{q}{a} - \frac{q}{a}(1-a)^t$.

Proof. We prove Lemma 7 by induction. For the base case $t=0$, $\mathrm{Var}\big(\boldsymbol{p}_i^{(0)}\big) = q/a - q/a = 0$ follows from the initialization $\boldsymbol{p}_i^{(0)} = 1/2$.

For the induction step, we make use of the law of total variance, i.e.,

$$\begin{aligned}\mathrm{Var}\Big(\boldsymbol{p}_i^{(t+1)}\Big) = \mathrm{E}\Big(\mathrm{Var}\Big(\boldsymbol{p}_i^{(t+1)}\,\Big|\,\boldsymbol{p}_i^{(t)}\Big)\Big) + \\ \mathrm{Var}\Big(\mathrm{E}\Big(\boldsymbol{p}_i^{(t+1)}\,\Big|\,\boldsymbol{p}_i^{(t)}\Big)\Big)\,.\end{aligned}$$

Since A is balanced, we have $\mathrm{E}\big(\boldsymbol{p}_i^{(t+1)} \,|\, \boldsymbol{p}_i^{(t)}\big) = \boldsymbol{p}_i^{(t)}$ and thus

$$\mathrm{Var}\Big(\mathrm{E}\Big(\boldsymbol{p}_i^{(t+1)}\,\Big|\,\boldsymbol{p}_i^{(t)}\Big)\Big) = \mathrm{Var}\Big(\boldsymbol{p}_i^{(t)}\Big)\ .$$

For the following transformations please note that because of $\big(\boldsymbol{p}_i^{(t)} - 1/2\big)^2 = \big(\boldsymbol{p}_i^{(t)}\big)^2 - \boldsymbol{p}_i^{(t)} + 1/4$ we have $\boldsymbol{p}_i^{(t)}\big(1-\boldsymbol{p}_i^{(t)}\big) = -\big(\boldsymbol{p}_i^{(t)} - 1/2\big)^2 + 1/4$. Further note that $\mathrm{E}\big(\boldsymbol{p}_i^{(t)}\big) = 1/2$ because A is balanced and $\boldsymbol{p}_i^{(0)} = 1/2$.

Making use of our assumption regarding $\mathrm{Var}\big(\boldsymbol{p}_i^{(t+1)} \,|\, \boldsymbol{p}_i^{(t)}\big)$ and the induction hypothesis, we get

$$\begin{aligned}\mathrm{Var}\Big(\boldsymbol{p}_i^{(t+1)}\Big) &= \mathrm{E}\Big(-a\big(\boldsymbol{p}_i^{(t)}\big)^2 + b\boldsymbol{p}^{(t)} + c\Big) + \mathrm{Var}\Big(\boldsymbol{p}_i^{(t)}\Big) \\ &= a\mathrm{E}\Big(\boldsymbol{p}_i^{(t)}\big(1-\boldsymbol{p}_i^{(t)}\big)\Big) + (b-a)\mathrm{E}\Big(\boldsymbol{p}_i^{(t)}\Big) + c + \mathrm{Var}\Big(\boldsymbol{p}_i^{(t)}\Big) \\ &= a\mathrm{E}\Big(-\Big(\boldsymbol{p}_i^{(t)} - \frac{1}{2}\Big)^2 + \frac{1}{4}\Big) + \mathrm{Var}\Big(\boldsymbol{p}_i^{(t)}\Big) - \frac{a}{2} + \frac{b}{2} + c \\ &= (1-a)\mathrm{Var}\Big(\boldsymbol{p}_i^{(t)}\Big) - \frac{a}{4} + \frac{b}{2} + c \\ &= (1-a)\Big(\frac{q}{a} - \frac{q}{a}(1-a)^t\Big) + q\end{aligned}$$

$$= \frac{q}{a} - \frac{q}{a}(1-a)^t - q + q(1-a)^t + q$$
$$= \frac{q}{a} - \frac{q}{a}\big((1-a)^t - a(1-a)^t\big) = \frac{q}{a} - \frac{q}{a}(1-a)^{t+1} \ . \qquad \square$$

Lemma 7 is interesting because it says that the variance of an f-independent position's frequency adhering to the restrictions of the theorem is q/a in the limit. Thus, if it turns out to be $1/4$, such a frequency is expected to have reached one of the borders 0 or 1 in the limit, i.e., it would drift away from $1/2$ toward the borders, although the frequency does not move in expectation. That is the same behavior as with the gambler's ruin.

We now show that this is exactly what happens for λ-AS$_{\text{IB}}$ and the cGA.

Corollary 8. *Let i be an f-independent position of λ-AS$_{IB}$. Then, for all $t \in \mathbb{N}$, $\operatorname{Var}\big(\boldsymbol{p}_i^{(t)}\big) = \frac{1}{4} - \frac{(1-\rho^2)^t}{4}$.*

Proof. We know from Theorem 6 that λ-AS$_{\text{IB}}$ is balanced. Since we want to use Lemma 7, we have to calculate the variance of $\boldsymbol{p}_i^{(t+1)}$ conditioned on $\boldsymbol{p}_i^{(t)}$.

$$\operatorname{Var}\Big(\boldsymbol{p}_i^{(t+1)} \,\Big|\, \boldsymbol{p}_i^{(t)}\Big) = \operatorname{E}\Bigg(\Big(\boldsymbol{p}_i^{(t+1)} - \operatorname{E}\Big(\boldsymbol{p}_i^{(t+1)} \,\Big|\, \boldsymbol{p}_i^{(t)}\Big)\Big)^2 \,\Big|\, \boldsymbol{p}_i^{(t)}\Bigg)$$
$$= \operatorname{E}\Bigg(\Big(\boldsymbol{p}_i^{(t+1)} - \boldsymbol{p}_i^{(t)}\Big)^2 \,\Big|\, \boldsymbol{p}_i^{(t)}\Bigg) \ ,$$

since λ-AS$_{\text{IB}}$ is balanced. To enhance the readability, let $p = \boldsymbol{p}_i^{(t)}$.

$$\operatorname{Var}\Big(\boldsymbol{p}_i^{(t+1)} \,\Big|\, \boldsymbol{p}_i^{(t)}\Big) = \operatorname{E}\Bigg(\Big(\boldsymbol{p}_i^{(t+1)} - \boldsymbol{p}_i^{(t)}\Big)^2 \,\Big|\, \boldsymbol{p}_i^{(t)}\Bigg)$$
$$= \big((1-\rho)p + \rho - p\big)^2 \cdot p + \big((1-\rho)p - p\big)^2 \cdot (1-p)$$
$$= \rho^2(1-p)^2 p + \rho^2 p^2 (1-p) = \rho^2 p(1-p)(1-p+p)$$
$$= -\rho^2 p^2 + \rho^2 p \ ,$$

which is in the form as we need it for Lemma 7. Applying the lemma with $a = \rho^2$, $b = \rho^2$, and $c = 0$ finishes the proof. $\square$

For the cGA the result looks nearly identically, the difference being a factor of 2 in a term in the numerator.

Corollary 9. *Let i be an f-independent position of the cGA. Then, for all $t \in \mathbb{N}$, $\operatorname{Var}\big(\boldsymbol{p}_i^{(t)}\big) = \frac{1}{4} - \frac{(1-2\rho^2)^t}{4}$.*

Proof. This proof works the same way as the one before. We also use the same notation.

$$\operatorname{Var}\Big(\boldsymbol{p}_i^{(t+1)} \,\Big|\, \boldsymbol{p}_i^{(t)}\Big) = \operatorname{E}\Bigg(\Big(\boldsymbol{p}_i^{(t+1)} - \boldsymbol{p}_i^{(t)}\Big)^2 \,\Big|\, \boldsymbol{p}_i^{(t)}\Bigg)$$
$$= (p + \rho - p)^2 \cdot p(1-p) + (p-p)^2 \cdot \big(1 - 2p(1-p)\big) +$$
$$(p - \rho - p)^2 \cdot p(1-p)$$
$$= 2\rho^2 p(1-p) = -2\rho^2 p^2 + 2\rho^2 p \ .$$

Applying Lemma 7 with $a = 2\rho^2$, $b = 2\rho^2$, and $c = 0$ yields, again, the result. $\square$

Seeing the results for λ-AS$_{\text{IB}}$ and the cGA, it would be interesting to know whether all f-independent positions of balanced n-Bernoulli-λ-EDAs with a trivial margin behave in the same way, that is, their respective frequency has a variance of $1/4$ in the limit.

We can answer this question for a big class of balanced n-Bernoulli-λ-EDAs with a trivial margin.

Theorem 10. *Let i be an f-independent position of a balanced n-Bernoulli-λ-EDA A with a trivial margin and 0 and 1 being the only fixed points of its update scheme. Assume that, for all $t \in \mathbb{N}$, $\operatorname{Var}\big(\boldsymbol{p}_i^{(t+1)} \mid \boldsymbol{p}_i^{(t)}\big) = -a\big(\boldsymbol{p}_i^{(t)}\big)^2 + b\boldsymbol{p}_i^{(t)} + c$, with $0 < a < 1$. Then $\operatorname{Var}\big(\boldsymbol{p}_i^{(t+1)} \mid \boldsymbol{p}_i^{(t)}\big) = a\boldsymbol{p}_i^{(t)}\big(1 - \boldsymbol{p}_i^{(t)}\big)$, and hence $\lim_{t\to\infty} \operatorname{Var}\big(\boldsymbol{p}_i^{(t)}\big) = \frac{1}{4}$.*

Proof. Let B denote the event $\boldsymbol{p}_i^{(t+1)} < \boldsymbol{p}_i^{(t)}$, let C denote the event $\boldsymbol{p}_i^{(t+1)} > \boldsymbol{p}_i^{(t)}$, and let $\varepsilon = \boldsymbol{p}_i^{(t+1)} - \boldsymbol{p}_i^{(t)}$. Since A is balanced, we have

$$\Big|\operatorname{E}\Big(\varepsilon \,\Big|\, \boldsymbol{p}_i^{(t)}, B\Big)\Big| \Pr(B) = \Big|\operatorname{E}\Big(\varepsilon \,\Big|\, \boldsymbol{p}_i^{(t)}, C\Big)\Big| \Pr(C) \ .$$

Looking at a decreasing sequence of frequencies going toward the lower border 0, i.e., for all $t \in \mathbb{N}$, $\boldsymbol{p}_i^{(t+1)} < \boldsymbol{p}_i^{(t)}$, we see that the left side of the equation goes toward 0. Hence, the right must too. That means that the conditional variance for such a sequence must be 0 in the limit. Using our assumption of this variance as a quadratic function, we see that $c = 0$.

We now argue the same way as above, but we use an increasing sequence of frequencies. Thus, for these frequencies approaching 1, the conditioned variance is again 0. Solving the quadratic equation, with $c = 0$, for a, we get $a = b$, which results in $\operatorname{Var}\big(\boldsymbol{p}_i^{(t+1)} \mid \boldsymbol{p}_i^{(t)}\big) = a\boldsymbol{p}_i^{(t)}\big(1 - \boldsymbol{p}_i^{(t)}\big)$.

Applying now Lemma 7 with $a = b$ and $c = 0$ yields the limit variance of $1/4$. $\square$

Of course Theorem 10 only tells us the variance of an f-independent frequency in the limit, whereas Corollaries 8 and 9 additionally give a rate of convergence for λ-AS$_{\text{IB}}$ and cGA, respectively.

Looking at the $(1-c\rho^2)^t$ term for both algorithms, we can bound $(1-c\rho^2)^t \le \mathrm{e}^{-2c\rho^2 t}$ and, thus, the variance. Assuming $t = \omega\big(1/\rho^2\big)$, lets the exponential term go toward 0, hence, after that many steps, an f-independent $\boldsymbol{p}_i$ is arbitrarily close to the borders 0 or 1.

Since λ-AS$_{\text{IB}}$ cannot reach the borders within finite time, being close to the borders just means that it takes a long time to get away from them. For the cGA, however, it means that an f-independent position's frequency (in the relaxed sense) could be stuck on either 0 or 1, making it impossible to change this bit position, and thus optimization is likely to fail.

5. SOLVING LeadingOnes EFFICIENTLY

Considering f-independent positions is of particular interest when analyzing an n-Bernoulli-λ-EDA optimizing LeadingOnes. If the maximum number of leading 1s over all individuals in D is j, all positions $i > j+1$ are LeadingOnes-independent (in a relaxed sense) in that iteration because the respective bits $\boldsymbol{x}_i$ of each individual in D do not contribute to the fitness.

We now look at the cGA optimizing LeadingOnes. We call positions j with $\boldsymbol{p}_j = 1$ *solved* and all other positions *unsolved*. The cGA can easily solve the leftmost *(first)* unsolved position j because an individual sampled with a 1 at position j always has a higher fitness than an individual having a 0. So $\boldsymbol{p}_j$ cannot decrease, whereas the frequencies

of all the other unsolved positions can. We say that an algorithm *efficiently* optimizes LeadingOnes if the expected run time of an algorithm is in $o(n^2)$.

We first look at the expected time needed for the first unsolved position to be solved, assuming its frequency is not too low.

Lemma 11. *Consider the cGA optimizing* LeadingOnes *and that i is the first unsolved position with $\boldsymbol{p}_i$ being $c = \Omega(1)$. $\boldsymbol{p}_i$ then reaches 1 within an expected number of $O(\rho^{-1} \log \rho^{-1})$ steps.*

Proof. Since i is the first unsolved position, $\boldsymbol{p}_i$ cannot decrease and the probability of making an increase is $2\boldsymbol{p}_i(1-\boldsymbol{p}_i)$; else it does not move.

We look at the drift of the potential $X_t = 1-\boldsymbol{p}_i$, i.e., how far $\boldsymbol{p}_i$ is away from the goal 1. The drift is thus

$$\mathrm{E}(X_t - X_{t+1} \mid X_t) = 2\boldsymbol{p}_i(1-\boldsymbol{p}_i)\rho = 2(1-X_t)X_t\rho \geq X_t\rho \ .$$

Because we look at the first hitting time T of $\boldsymbol{p}_i$ going below ρ, we have to scale the space by $1/\rho$. Note that this does not change the relative drift. The initial potential X_0 is thus $(1-c)/\rho = O(\rho^{-1})$, and by Theorem 1 we get

$$\mathrm{E}(T \mid X_0) \leq \frac{1}{\rho}\left(1 + \ln \frac{1-c}{\rho}\right) = O\left(\frac{1}{\rho} \log \frac{1}{\rho}\right) . \qquad \square$$

Now that we took a closer look at the cGA, we want to give a general result for optimizing LeadingOnes with an n-Bernoulli-λ-EDA.

Theorem 12. *Consider an n-Bernoulli-λ-EDA with a trivial margin optimizing* LeadingOnes*. Let q be a polynomial and let $0 < \ell < 1/2$ be a real possibly dependent on n.*

If, for each unsolved position i, the frequency $\boldsymbol{p}_i$ drops below ℓ within $O(nq(n))$ rounds only with probability at most $n^{-(\varepsilon+1)}$ for any constant $\varepsilon > 0$, and if, for each first *unsolved position j, $\boldsymbol{p}_j \geq \ell$ reaches 1 within $O(q(n))$ rounds in expectation, then, with probability at least $1 - n^{-\varepsilon}$, the algorithm succeeds after an expected time of $O(nq(n))$.*

Proof. First, since each frequency only drops below ℓ within $O(nq(n))$ rounds with probability at most $n^{-(\varepsilon+1)}$, at least one of the n frequencies does so during the same number of rounds with probability at most $n^{-\varepsilon}$, by union bound. Thus, with probability at least $1 - n^{-\varepsilon}$, all frequencies will be at least at ℓ for $O(nq(n))$ rounds.

By induction and linearity of expectation, all frequencies reach 1 within an expected time of $O(nq(n))$. $\square$

Theorem 12 shows us that an n-Bernoulli-λ-EDA can optimize LeadingOnes in $O(n \log n)$ if the time needed for each frequency to reach 1 is in $O(\log n)$, and if the frequencies of yet unsolved positions do not drop too low with high probability.

Because the cGA is not stable, it is unlikely to solve LeadingOnes efficiently. We hence propose to change the set function of the cGA such that the algorithm becomes stable and such that each first unsolved position's frequency reaches 1 within $O(\log n)$ rounds. We call this variant *scGA* for *stable cGA* with parameters a and d.

$$\mathrm{set}(\boldsymbol{p}_i) = \begin{cases} \boldsymbol{p}_i + \rho + a\,, & \text{if } \boldsymbol{p}_i < 1/2\,; \\ \boldsymbol{p}_i + \rho\,, & \text{if } 1/2 \leq \boldsymbol{p}_i < d\,; \\ 1\,, & \text{else.} \end{cases}$$

a is a bias that makes an f-independent $\boldsymbol{p}_i$ concentrate around $1/2$. It does so because the update toward $1/2$ will always be larger by a than the update to a border.

The parameter d helps in reaching 1 or 0 faster since it sets a $\boldsymbol{p}_i$ directly to that value once d or $1-d$ is reached. We do so, because the scGA is stable, hence, it is unlikely that $1-d$ is reached accidentally, which would imply that $\boldsymbol{p}_i$ wrongly fixated to 0.

If $d = \Theta(1) \in (1/2, 1)$, we can re-use the proof of Lemma 11 and see that, for the scGA, for each first unsolved position i, $\boldsymbol{p}_i = \Omega(1)$ reaches 1 in $O(\rho^{-1})$ steps because we have to scale the search space only by the constant $1/(1-d)$. Note that a does not influence the proof.

We now show that frequencies of f-independent positions of the scGA, with certain parameters a and d, concentrate around $1/2$.

Lemma 13. *Consider an f-independent position i of the scGA with $d^2(1-d)a/(\rho+a)^2 \geq \ln n^{16c}$, for $c = \Theta(1)$, $\rho + a = o(1)$, and $d = \Theta(1) \in (1/2, 1)$. After $n^{c'}$, $c' < c$, rounds of the algorithm, $\boldsymbol{p}_i$ will reach either d or $1-d$ only with probability at most $2n^{c'-c}$.*

Proof. We focus on $\boldsymbol{p}_i \in [1/2, 1]$ and show that it reaches d within any polynomial number of rounds only with polynomially low probability. Because the scGA is locally updating, the argumentation for $\boldsymbol{p}_i \in [0, 1/2]$ follows analogously.

This proof uses Theorem 2. Hence, we are interested in the drift of $\boldsymbol{p}_i$. Therefore, let $\boldsymbol{p}_i'$ denote $\boldsymbol{p}_i$ after an update. Since we want to show that d is only reached with low probability, we condition on the event that $\boldsymbol{p}_i < d - \rho$ without denoting this explicitly.

$$\begin{aligned} \mathrm{E}(\boldsymbol{p}_i' - \boldsymbol{p}_i \mid \boldsymbol{p}_i) &= (\boldsymbol{p}_i + \rho - \boldsymbol{p}_i)\boldsymbol{p}_i(1-\boldsymbol{p}_i) + \\ &\quad (\boldsymbol{p}_i - \boldsymbol{p}_i)\big(1 - 2\boldsymbol{p}_i(1-\boldsymbol{p}_i)\big) + \\ &\quad (\boldsymbol{p}_i - \rho - a - \boldsymbol{p}_i)\boldsymbol{p}_i(1-\boldsymbol{p}_i) \\ &= \rho\boldsymbol{p}_i(1-\boldsymbol{p}_i) - (\rho+a)\boldsymbol{p}_i(1-\boldsymbol{p}_i) \\ &= -a\boldsymbol{p}_i(1-\boldsymbol{p}_i) \leq -d(1-d)a \ . \end{aligned}$$

Since we have $|\boldsymbol{p}_i' - \boldsymbol{p}_i| \leq \rho + a = o(1) < \Theta(1) = d$, we can now use Theorem 2 to bound the hitting time T of $\boldsymbol{p}_i$ reaching d within t steps.

$$\Pr(T \leq t) \leq t\mathrm{e}^{-\frac{d|-d(1-d)a|}{16(\rho+a)^2}} \leq t\mathrm{e}^{-\frac{d^2(1-d)a}{16(\rho+a)^2}} \leq t\mathrm{e}^{-c\ln n} = \frac{t}{n^c} \ .$$

Hence, if $t \leq n^{c'}$, the probability of $\boldsymbol{p}_i$ reaching d for any $d = \Theta(1) \in (1/2, 1)$ within t steps is at most $n^{c'-c}$. $\square$

We can now conclude that the scGA is able to optimize LeadingOnes in $O(n \log n)$ as we show in the following corollary.

Corollary 14. *With probability polynomially close to 1, the scGA with $\rho = \Theta(1/\log n)$, $a = O(\rho) > 0$, and $d = \Theta(1) \in (1/2, 1)$ optimizes* LeadingOnes *in $O(n \log n)$ in expectation.*

Proof. We want to use Theorem 12, so we make sure to fulfill the requirements.

As discussed beforehand, since d is a constant, the expected time needed for each first unsolved position's frequency, starting from a constant value, to reach 1 is in $O(\rho^{-1})$, which is in $\Theta(\log n)$ since we assume $\rho = \Theta(1/\log n)$.

We then use Lemma 13. The scGA has no LeadingOnes-independent positions, but we pessimistically assume each unsolved position except the first to be LeadingOnes-independent. We can do so because the frequency of an unsolved position contributing to the fitness cannot decrease. So the probability from Lemma 13 of such a frequency not reaching $1-d$ is an upper bound for the actual probability. We now apply the lemma.

$\rho + a = \mathrm{o}(1)$ holds by assumption and so does $d = \Theta(1) \in (1/2, 1)$. Because $a = \mathrm{O}(\rho)$, we get $d^2(1-d)a/(\rho+a)^2 = \Omega(\rho^{-1}) \geq \ln n^{16\cdot(2+2\varepsilon)}$, $\varepsilon = \Theta(1)$, for sufficiently large values of ρ and a. Therefore the probability of each unsolved position's frequency to reach $1-d$ within $n^{1+\varepsilon} = \omega(n \log n)$ steps is at most $n^{-(\varepsilon+1)}$.

We can now use Theorem 12, which completes the proof. □

Interestingly, the bias a plays a far less significant role than ρ. Basically any small bias toward $1/2$ suffices, whereas ρ has to be in $\Theta(1/\log n)$. If ρ were any larger, there were a decent chance of an f-independent position's frequency reaching $1-d$, which would mean that such a frequency actually reached 0. On the other hand, if ρ were any smaller, it would take too long for each first unsolved position's frequency to reach 1 within $\mathrm{O}(\log n)$ steps.

Acknowledgments

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 618091 (SAGE).

References

[1] P. Afshani, M. Agrawal, B. Doerr, C. Doerr, K. Larsen, and K. Mehlhorn. The query complexity of finding a hidden permutation. In *Space-Efficient Data Structures, Streams, and Algorithms*, pp. 1–11. 2013.

[2] S. Baluja. Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning. Technical report, 1994.

[3] T. Chen, P. Lehre, K. Tang, and X. Yao. When is an estimation of distribution algorithm better than an evolutionary algorithm? In *Proc. of IEEE CEC '09*, pp. 1470–1477, 2009.

[4] D. Corus, D.-C. Dang, A. V. Eremeev, and P. K. Lehre. *Proc. of PPSN XIII*, pp. 912–921. 2014.

[5] D.-C. Dang and P. K. Lehre. Simplified runtime analysis of estimation of distribution algorithms. In *Proc. of GECCO '15*, pp. 513–518, 2015.

[6] B. Doerr, D. Johannsen, and C. Winzen. Multiplicative drift analysis. *Algorithmica*, 64: 673–697, 2012.

[7] S. Droste. A rigorous analysis of the compact genetic algorithm for linear functions. *Natural Computing*, 5: 257–283, 2006.

[8] S. Droste, T. Jansen, and I. Wegener. On the analysis of the (1+1) evolutionary algorithm. *Theor. Comput. Sci.*, 276:51–81, 2002.

[9] T. Friedrich, T. Kötzing, M. S. Krejca, and A. M. Sutton. The benefit of recombination in noisy evolutionary search. In *Proc. of ISAAC '15*, pp. 140–150, 2015.

[10] G. Harik, F. G. Lobo, and D. E. Goldberg. The compact genetic algorithm. In *IEEE Trans. Evol. Comput.*, pp. 523–528, 1998.

[11] M. Hauschild and M. Pelikan. An introduction and survey of estimation of distribution algorithms. *Swarm and Evolutionary Computation*, 1:111–128, 2011.

[12] M. Hohfeld and G. Rudolph. Towards a theory of population-based incremental learning. In *Proc. of CEC '97*, pp. 1–5, 1997.

[13] T. Jansen, K. A. De Jong, and I. Wegener. On the choice of the offspring population size in evolutionary algorithms. *Evol. Comput.*, 13:413–440, 2005.

[14] T. Kötzing. Concentration of first hitting times under additive drift. *Algorithmica*, pp. 1–17, 2015.

[15] P. Larraanaga and J. A. Lozano. *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*. 2001.

[16] H. Mühlenbein and G. Paass. From recombination of genes to the estimation of distributions I. binary parameters. In *Proc. of PPSN IV*, pp. 178–187, 1996.

[17] F. Neumann, D. Sudholt, and C. Witt. A few ants are enough: ACO with iteration-best update. In *Proc. of GECCO '10*, pp. 63–70, 2010.

[18] Y. Ollivier, L. Arnold, A. Auger, and N. Hansen. Information-geometric optimization algorithms: A unifying picture via invariance principles. *Journal of Machine Learning Research*, 2016+. Accepted, available at http://arxiv.org/abs/1106.3708.

[19] T. Paixão, G. Badkobeh, N. Barton, D. Çörüş, D.-C. Dang, T. Friedrich, P. K. Lehre, D. Sudholt, A. M. Sutton, and B. Trubenová. Toward a unifying framework for evolutionary processes. *Journal of Theoretical Biology*, pp. 28–43, 2015.

[20] J. E. Rowe and D. Sudholt. The choice of the offspring population size in the (1,λ) EA. In *Proc. of GECCO '12*, pp. 1349–1356, 2012.

[21] K. Sastry, D. E. Goldberg, and X. Llorà. Towards billion bit optimization via parallel estimation of distribution algorithm. In *Proc. of GECCO '07*, pp. 577–584, 2007.

[22] H.-P. P. Schwefel. *Evolution and Optimum Seeking: The Sixth Generation*. 1993.

[23] J. L. Shapiro. Diversity loss in general estimation of distribution algorithms. In *Proc. of PPSN IX*, pp. 92–101, 2006.

[24] T. Stützle and H. H. Hoos. Max-min ant system. *Future Gener. Comput. Syst.*, 16:889–914, 2000.